Journal of Interconnection Networks
Vol. 10, Nos. 1 & 2 (2009) 59–91
© World Scientific Publishing Company

World Scientific
www.worldscientific.com

TOWARDS IMMUNIZING MANET'S SOURCE ROUTING PROTOCOLS AGAINST PACKET DROPPERS

DJAMEL DJENOURI
CERIST Research Center, Rue des Freres Aissou, Ben Aknoun,
Bp 143, Algiers, 16030, Algeria
ddjenouri@mail.cerist.dz
http://http://djenouri.googlepages.com/

OTHMANE MAHMOUDI
CERIST Research Center, Rue des Freres Aissou, Ben Aknoun,
Bp 143, Algiers, 16030, Algeria
o.mahmoudi@gmail.com

MOHAMED BOUAMAMA
CERIST Research Center, Rue des Freres Aissou, Ben Aknoun,
Bp 143, Algiers, 16030, Algeria
m.bouamama@gmail.com

Received 7 July 2008
Accepted 3 November 2009

This paper deals with security of routing protocols of Mobile Ad hoc Networks (MANETs), and proposes a solution to immunize such protocols against packet dropping misbehavior. Most of the current secure protocols are vulnerable to packet dropping misbehavior, which can be exploited by selfish nodes and malicious ones as well. For example, simply by dropping RREQ (Route Request) packets a selfish node can exclude itself from routes and thereby avoid receiving data packets to forward. On the other hand, a malicious node can drop RERR (Route Error) packets to keep the use of failed routes, possibly resulting in a denial of service. To mitigate this vulnerability we propose a hybrid solution that secures routing protocols against the dropping of both directed and broadcast control packets, in which a different approach is adapted for each kind of packets. Dealing with control packets represents the main contribution in this manuscript, as all the current proposals in the context of selfish nodes only consider data packets. Our solution can be integrated with any source routing protocol. In this work it was implemented with one of the most secure protocols, namely ENDAIR. The resulting new extended secure protocol was assessed and analyzed through an extensive simulation study.

Keywords: Mobile ad hoc network; security; routing.

1. Introduction

Most of the current solutions aiming at securing MANET against selfish misbehavior or packet dropping attack (Byzantine black hole attack) focus on data packets, and are not directly applicable to control packets owing to many differences between these two kinds of packets. First, the number of control packets is too low compared with data packets, making the solutions that rely on a long experience before making a judgment (such as the Bayesian-based methods [1,2]) inappropriate, and thus a more realistic threshold definition is required. Second, some control packets are broadcast messages (e.g. RREQ), which might require a different monitoring approach compared to directed packets. In this paper we focus on all these issues, and propose a comprehensive hybrid solution to monitor the forwarding of control packets, judge the monitored nodes, and isolate the detected misbehaving nodes. The solution is hybrid as for each step it includes different approaches, and the one to be employed depends on the kind of the packet, i.e directed and broadcast. Regarding monitoring the two-hop ACK approach [3] is used for directed packets, and a promiscuous mode based approach (watchdog-like) is proposed for the broadcast ones. For judgment a redemption method is proposed, allowing nodes that are observed to forward packets to be redeemed. Finally, a witness-based isolation method is employed to isolate the suspicious nodes, in which a different algorithm is used for each kind of packets. The proposed solution is highly abstracted and hence can be integrated with any source routing protocol, notably the secure ones. It allows to immunize the routing protocol against packet dropping misbehavior. In our implementation ENDAIRA [4] was chosen, which is highly secure but suffers from packet dropping vulnerability. The remainder of this paper is organized as follows: The next section introduces some protocols from which our solution is derived. The third section describes our solution and its integration with ENDAIRA [4], followed by some discussions and analysis in section four, and a simulation evaluation study in section five. Section six sketches related work, while section seven concludes the paper and summarizes some perspectives.

2. Background

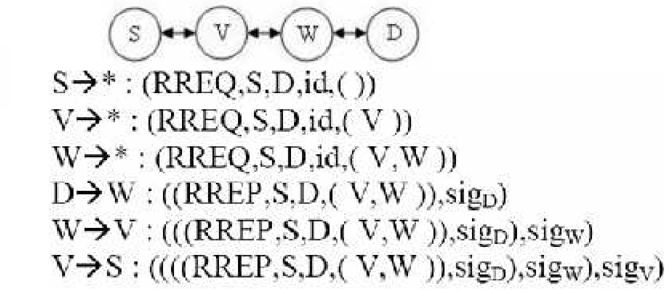
In the following some concepts used in the rest of the paper are introduced.

2.1. *DSR*

DSR (Dynamic Source Routing) [5] is a well-known reactive routing protocol of MANET, based on the source route approach. The principal of this approach is that the whole route is chosen by the source, and is put within each packet to be sent. Every node keeps in its cache (routing table) the source routes learned. When it needs to send a packet it first checks in its cache for the existence of such a route. If no entry to the appropriate destination is available in the cache, then the node launches a route discovery by broadcasting a request (RREQ) packet through the network. When receiving the (RREQ) a node seeks a route in its cache for the RREQ's destination, finding such a route results in sending a route reply (RREP) packet to the source. This is known as replying from cache optimization, aiming at limiting the long propagation of requests. However, if no appropriate route exists in its cache then the intermediate node adds its address to the RREQ and continues broadcasting it. In addition to route discovery procedure, DSR includes route maintenance. When some node detects a route failure, it sends a route error (RERR) packet to the source that is using this link, then this one repeats the route discovery process. Despite the insecurity of the this protocol, as it does not guaranty any security service, it has been largely used in literature for building new secure routing protocols. Almost all the current secure protocols just add some security procedures to DSR [6]. An example of such procedures will be illustrated hereafter with ENDAIRA.

2.2. *ENDAIRA*

This protocol has been inspired from ARIADAN [7], a DSR-based secure protocol. The main idea of ENDAIRA's first version [8] that distinguishes it from ERIADAN is to sign RREP packets instead of RREQs. In addition, the source node that initiates the route discovery adds to the RREQ a random ID assigned to the request. When receiving the RREQ the destination node produces a RREP that includes, additionally to the items defined by DSR, a digital signature on these items. Each intermediate node receiving the RREP checks that its ID appears in the source route, and that the ones of the previous and the next nodes correspond to neighboring nodes. Moreover, it verifies the signatures of the previous nodes and appends its signature to the previous ones (signs the packet and the signatures), then sends the packet back towards the source. When reaching the source, the route will not be accepted until performing all the verifications (the same as the intermediary). Also, an intermediary that finds any verification to be failed ignores the RREP and stops its forwarding. The example of figure 1 illustrates the operations of ENDAIRA in a scenario where node S is seeking a route towards D. Note that '$*$' means that the message is for broadcast, and $(msg, sign_x)$ denotes the signature of the item msg by node x.



S→* : (RREQ,S,D,id,( ))
V→* : (RREQ,S,D,id,(V))
W→* : (RREQ,S,D,id,(V,W))
D→W : ((RREP,S,D,(V,W)),sig_D)
W→V : (((RREP,S,D,(V,W)),sig_D),sig_W)
V→S : ((((RREP,S,D,(V,W)),sig_D),sig_W),sig_V)

Fig. 1. Basic ENDAIRA.

It has been shown that this version is vulnerable to some attacks [4]. An intermediary that already received from a destination a valid but expired RREP (one of the downstream links has been broken after the discovery) could use it to reply to a route request launched by the same source (if it launches again a route discovery either after some link failure or when the route lifetime expires). In addition to this vulnerability, spoofing the source address in RREQ is possible, as there is no authentication of such kind of packets. These two vulnerabilities can be exploited to make sophisticated attacks resulting in the discovery of invalid routes (DoS attacks) [4]. To counter these attacks the authors proposed some enhancement to the basic ENDAIRA, by suggesting to apply digital signatures to both RREQ and RREP, to encrypt the random ID, and to reuse it in the RREP [4]. This version is the most secure one. Nonetheless, it is vulnerable to the packet dropping misbehavior, since no monitoring of the packet forwarding was provided. That is, a selfish intermediate node can simply drop RREQ or RREP to exclude itself from routes, and a malicious node can drop either RRERs to prevent route maintenance and thus causing a DoS, or data packets to cause harms to the communicating end-point nodes. This problem has been treated in the context of selfish misbehavior in MANET, resulting in many solutions preventing the dropping of *data packets* that can be integrated with secure routing protocols. In the following we present two solutions that will be use later.

2.3. *Watchdog*

Watchdog (WD) [9] is the first solution that treats the problem of nodes misbehavior on packet forwarding in MANET. It is a basic technique upon which many further solutions rely. The main idea is to monitor neighbors using the promiscuous mode. Suppose node S sends packets to D through a route including (possibly amongst others) respectively three intermediate nodes: A, B, and C. When A transmits a packet to B to forward to C, A can check whether B forwards each packet by analyzing packets it overhears in the promiscuous mode during a given timeout. If A overhears a packet it is monitoring during the fixed timeout then it validates its forwarding, otherwise it raises a rating regarding B and then will consider that B is misbehaving as soon as the rate exceeds a given threshold. This monitoring procedure is generalized for each pair of hops in the source route. The approach is able to detect misbehaving nodes in many cases, and requires no overhead as long as nodes cooperate. It allows to monitor all packets regardless whether they are directed or broadcast. Nonetheless, WD fails in case of collisions, collusions, and power control employment as illustrated hereafter.

After a collision at C, B can circumvent retransmitting the packet without being detected by A. B can also circumvent WD by partially dropping packets, viz. at a lower rate than the configured accusation threshold. WD fails when two successive nodes collude to conceal the misbehavior of each other. That is, B can collude with C by not reporting to A when C misbehaves. Further, WD technique may cause false detections when the configured threshold fails,[a] and especially when the monitored node uses the power control technique to preserve its power. When C is closer to B than A and B transmits packets using a controlled power according to the distance separating it from C, A cannot overhear B's forwarding and may accuse it wrongly. Note that the power control technique has been used by many routing protocols proposed after WD in the field of power-aware routing [10,11].

Although WD does not punish the detected misbehaving nodes, it was used by almost all the recent solutions that treat this problem (misbehaving punishment), notably at their monitoring components that monitor the forwarding of *data* packets.

[a] the number of packets lost due to mobility and channel condition exceeds the configured threshold.

2.4. *Two-hop ACK*

To mitigate the problem related to the power control technique usage of WD, a monitoring approach based on a new kind of feedbacks called *two-hop ACK* has been proposed [3]. In the context of three aligned nodes, A, B, and C, such that A monitors B's forwarding to C, node C acknowledges packets sent from A by sending the latter *via B* a special ACK that travels two hops. Node B could, however, escape from the monitor without being detected by simply sending A a *falsified* two-hop ACK. To avoid this vulnerability, the authors use an asymmetric cryptography based strategy, where A generates a random number and encrypts it using C's public key, then A validates B's forwarding if and only if it receives later the random number it generated within the two-hop ACK. Otherwise, it notices a packet dropping for B after a timeout. This random number received at A is encrypted by C with A's public key. C does so (encrypts the random number with A's public key) after decrypting the number with its private key. This way, B cannot falsify a valid two hop ACK unless it gets or breaks C's private key.

Since the validation at A is related to C's reception and not only to B's forwarding, the solution is independent of the power control usage, and thus solves WD's problems related to this issue. Unlike WD, the two-hop ACK ensures that after a collision at C, B cannot escape from retransmitting the packet without being detected. Nodes are therefore not required to receive all packets sent in their neighborhood and can turn-off their radios, which is power efficient. This advantages over WD make this solution superior in terms of efficiency in monitoring than all the other watchdog-based solutions.

The major drawback of this solution is its important communication overhead, since a two-hop ACK is required for each data packet on each couple of hops. For a route of H hops, $2(H-1)$ ACK packets will be engendered for each data packet. Therefore, this solution that in fact was proposed for data packets cannot be generalized for monitoring broadcast packets. If we try to do so and requires that all two hop neighbors acknowledges broadcast packets, then the complexity for a broadcast packet in a network with a degree of connectivity C would be $2C^2 \times (H-1)$, which is irrational. Two optimizations have been proposed to this solution [12]; the use of MAC ACKs and the random asking. First, 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 employment of the MAC ACK reduces the number of two-hop ACK packets as much as half compared with a separate transmission on each hop. In the second optimization a monitor node does not continuously ask ACKs but it does so randomly with a coefficient that depends on the behavior of the monitored node. The coefficient is decreased when the monitored node behave cooperatively such that to give it more trust and ask less ACKs. The analysis and simulation study [12] demonstrate a considerable reduction of the overhead.

3. Solution Description

In the following a new solution to secure routing protocols against packet dropping is presented. We focus on routing control packets, as the solutions proposed to data packets cannot be directly applicable to control packets as shown later. Control packets can be divided into two kinds; directed packets and broadcast packets. The first kind consists of packet intended for one recipient, while the second consists of packets sent to all neighboring nodes. With the latter the approach of WD is efficient, while the two-hop ACK is efficient with the first kind. We propose a hybrid solution that is watchdog-like when monitoring broadcast packets, and two-hop ACK based when monitoring directed packets. We also deal with post-detection issues and propose solutions to approve and isolate suspicious nodes. First, the security and communication assumptions of the environment in which the solution operates are given.

3.1. *Assumptions*

- A public key infrastructure (PKI) certificate authority along with an access control scheme are implemented, in charge of authentically distributing public keys while limiting the certificates (IDs and keys) a node can authentically get to one, i.e. none can get more than one authentic certificate (valid ID and key) at a given time, which prevents Sybil attack. Although PKI and access control in ad hoc networks are problematic and open research issues, some solutions have been proposed and can be used. Dealing with these problems is definitely out of the scope of our work. Note that the same keys can be employed for other security purposes at other layers, and thus are not specific overhead of our solution.
- The only considered misbehavior in this work is packet dropping. However, we take into account all the possible vulnerabilities of our solution that can be exploited by a misbehaving node, and prevent it from circumventing the solution after dropping packets, and from launching another attack on the solution as well.
- An attacker can spoof IP addresses, falsify packets (as long as there is no authentication on them), and launch a distributed denial of service attack by compromising other nodes. However, we suppose that it cannot get more than one authentic certificate, and therefore cannot authentically use more than one ID. This must be ensured by the assumed access control mechanism together with the PKI.
- Communication links between each couple of nodes are FIFO (First-In First Out) and bidirectional, enabling communication in the two directions. If nodes use heterogenous hardware then the link can operate if the nodes are within the minimum power range of the two transceivers.
- Nodes are mobile and links are not always reliable, i.e packet collisions and losses are possible.
- Nodes are neighborhood-aware, i.e. each node has the list of all nodes in its communication range (neighbors).
- Source routing is used, and promiscuous mode is enabled, which ensures that all packets overheard by a node (packets sent in the same communication channel but not destined to the node) can captured and treated by its routing protocol.

3.2. *Monitoring directed packets*

The approach suggested for monitoring the forwarding of directed routing control packets (RREP and RERR) is based on the two-hop ACK presented before, wich applies for this kind of packets. Remember that this solution needs to be implemented with a source routing protocol, and employs asymmetric cryptography. The MAC ACKs piggybacking optimization is also employed, but not the random asking optimization (random two-hop-ACK). The reason of that is to obtain a maximum sample of control packets, whose number is much lower than data packets. As soon as the monitor node detects that the number of packets dropped by the monitored node exceeds a defined threshold, it considers the latter as misbehaving node and proceeds to its isolation. The setting of this threshold and its impact will be discussed later.

3.3. *Monitoring broadcast packets*

For RREQs packets (which are broadcast) we propose a watchdog-like approach, with some modifications to make it suitable for control packets. Each node monitors every RREQ it forwards or launches 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 every neighbor, except the node from which it received the RREQ if the node is not the source. If no 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 the configured threshold number of packets it judges B as misbehaving, and then tries to isolate it as illustrated later.

3.4. *Redeeming benign nodes*

To overcome false detections that may occur due to channel conditions, namely node mobility and packet collisions, we propose a redemption strategy for both kinds of packets. The aim is to allow a well-behaving node improving its reputation and tolerance threshold after it has been observed to drop packets owing to mobility or collisions. This can be achieved by decreasing the number of packets considered dropped each time it is perceived to correctly forward packets. The pace of decreasing is not inevitably fixed to one, but its value should be lower than one to prevent nodes from abusing this redemption. That is, forwarding one packet does not decrease the number of packets considered dropped by one. If the pace is m/n (such that $m, n \in \mathbb{N}, m < n$), then forwarding n packets results in decreasing the number by m. More investigations into this parameter (redemption pace) will be illustrated in the next section.

3.5. *Isolation*

After judging a node as misbehaving, the detector attempts to isolate it. Isolating a misbehaving node means: i) do not route packets through it, to avoid losing them, and ii) do not

Algorithm 1 Algorithm executed by node i, describing the Network layer component of the Monitor

Notations:

• R_{Key}: encrypting R with Key, • R^{key}: decrypting R with Key, • P_X: the public key of node X, • S_X: the secrete key of node X. $P(X)$: Probability of asking an ACK for monitoring node X

Directed Packets

Before sending a packet (RREP or RERR) to node X (X either the next hop or the destination and i is either the source or a forwarding node):

```
if (X $\neq$ D's destination) then
    Decide whether to require a two-hop ACK with probability $P(X)$
    if (Two-hop ACK required) then
        R = generate an even random number
        add(R,X) to the buffer Wait2HopsACK
    else
        R = generate an odd random number
    end if
    Y = X's successor in the source route
    append $(R_{P_Y}, i)$ to D's header
end if
send D to X
```

When receive a packet D from the MAC protocol sent by X:

```
if (X $\neq$ D's source) then
    remove the random number generated by X's predecessor from the header along with the corresponding node address
end if
send the packet to the network layer protocol
```

When receive a two-hop ACK packet TwoHopsACK from the MAC layer component

```
$R' = TwoHopsACK.Rand^{S_I}$
if $(R', TwoHopsACK.sender) \in Wait2HopsACK$ then
    remove $(R', TwoHopsACK.sender)$ from Wait2HopsACK
    $nbrcontrolfwd_X = nbrcontrolfwd_X + 1$
    if ($nbrcontrolfwd_X = n_{pace}$) then
        $nbrcontroldrp_X = nbrcontroldrp_X - m_{pace}$
    end if
    $P(X) = P_{trust}$
end if
```

When a timeout of a Wait2HopsACK entry (R,X) is expired

```
$nbrcontroldrp_X = nbrcontroldrp_X + 1$
$P(X) = 1$
if ($nbrcontroldrp_X > threshold$) then
    Put X in the suspects' set
    Inform the isolator to launch WREQ against X for directed packet
end if
```

Broadcast Packets

When receive RREQ from X

```
for each neighbor j except X do
    Set a timer
    Set BroadcastFlag[j] to false
end for
```

When a timeout of RREQ regarding X expired

```
if BroadcastFlag[X]=false then
    $nbrcontroldrp_X = nbrcontroldrp_X + 1$
    if ($nbrcontroldrp_X > threshold$) then
        Put X in the suspects' set
        Inform the isolator to launch WREQ against X for broadcast packet
    end if
else
    $nbrcontrolfwd_X = nbrcontrolfwd_X + 1$
    if ($nbrcontrolfwd_X = n_{pace}$) then
        $nbrcontroldrp_X = nbrcontroldrp_X - m_{pace}$
    end if
end if
```

forward packets for it to punish it. A node A that judges some other node B as misbehaving should not punish it unilaterally, but must ensure that this will be done by all the nodes. When A unilaterally punishes B, the other ones may consider A as misbehaving when they

Algorithm 2 Algorithm executed by node i, describing the MAC component of the Monitor

Directed Packets

```
When receive a packet D sent by X
  if (D.MACHeader.TwoHopsACK == true) then
    $R = D.MACHeader.Rand^{S_I}$
    if ($R$ mod $2 = 0$) then
      construct an ACK packet ACKpack
      ACKpack.TwoHopsACK = true
      $R' = R_{P_{D.MACHeader.TwoHopsSrc}}$
      ACKpack.Rand= $R'$
      ACKpack.SecondDest=D.MACHeader.TwoHopsSrc
      the other fields have to be filled out by the MAC protocol
      send the ACK to X
    else
      send an ordinary ACK if the packet requires an ACK
    end if
  else
    send an ordinary ACK if the packet requires an ACK
  end if
  pass the packet up to the network component after doing the handling required by the MAC protocol (moving the MAC header, frames defragmentation,..etc)
When receive packet D from the network layer
  Do the operations required by the MAC protocol (fragmentation, making the MAC header,..etc)
  if (D's source $\neq$ i) then
    D.MACHeader.TwoHopsACK = true
    D.MACHeader.Rand= the random number generated and encrypted by the network component
    D.MACHeader.TwoHopsSrc = i's predecessor
  else
    D.TwoHopsACK = false
  end if
  forward the packet
When receive an ACK packet ACKpack
  if (ACKpack.TwoHopsACK ==true) then
    construct a two-hop ACK packet TwoHopsACK
    TwoHopsACK.Rand= ACKpack.Rand
    TwoHopsACK.dest= ACKpack.SecondDest
    TwoHopsACK.sender=I
    send two-hop ACK to ACKpack.SecondDest
  end if
  do the handling required by the MAC protocol
When receive a two-hop ACK packet TwoHopsACK
  pass the packet up to the network layer component
```

Broadcast Packets

```
When overhear a RREQ or RREP packet from X regarding a packet being monitored
  BroadcastFlag[X]=true
```

realize that it does not forward packets for B. In social life a person that accuses another must show proof, and a possible way to prove the accusation is to get witnesses against the accused person.

Similarly, to isolate a detected node we suggest the use of a testimony-based protocol. Upon a detection, the detector informs nodes in its neighborhood about the dropper (the accused), and asks for witnesses by broadcasting a WREQ (Witness REQuest) packet. It also puts the detected node ID in a special set called *the suspects' set*. Each node receiving the WREQ investigates the issue according to the type of the packet perceived to be dropped as follows:

1) Directed packets: The receiver of WREQ immediately sends a *signed* WREP (Witness REPly) packet to the accuser if its suspects' set includes the accused node (denoted by B). Otherwise, if it is a neighbor of B but has not enough experience with it then it asks its successor (of B) whether it has received packets forwarded from it by sending an

ACREQ (ACcusation REQuest) packet (using a route not including B). But first, and 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. This is ensured by verifying whether such a packet has been recently overheard, using the promiscuous mode. The node also should check whether B has sent the accuser node an ACK *just after* overhearing the data, to ensure that the former has really received the packet and that the latter is not attempting a DoS attack on C by reporting false accusations to A (impressing A). If B's successor has not recently received any packet *forwarded* from B, it sends a *signed* ACREP (ACcusation REPly) packet to the investigator, then the latter testifies for the accusation and sends the accuser a signed WREP (Witness REPly) packet.

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 a 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 first, it must make sure that the accuser node has indeed recently sent out a RREQ, by looking up in its buffer.

As soon as the detector collects k validation messages 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 aims at mitigating wrong accusations caused by false testimonies [2]. Each node receiving such a valid accusation isolates the guilty. Otherwise, if the detector fails to collect k testimonies then it does not punish the detected node, but keeps it in the suspects' set and can avoid sending its own packets through it. The parameter k will be investigated in the simulation study.

The new solution can be implemented with any source routing protocol. We chose ENDAIRA, the securest of the current literature to our knowledge, to which the solution is integrated resulting in a secure and packet-dropping immune routing protocol. In the following the different approaches used by our solution are analyzed, then evaluated through a comparative simulation study.

4. Analysis and Discusion

4.1. *Monitoring*

Broadcast (RREQ) packets are sent to all neighboring nodes, thus using the full power. Therefore, the WD's problem related to the power control does not exist for this kind of packets, which allows to use WD to monitor them. In contrast to broadcast packets, a different approach was used to monitor directed packet that enabled the employment of power control. In the following we prove the correctness of this new approach.

First, the directed packets monitoring is modeled using Petri nets. After that, some linear algebra concepts are applied to the model to proof the protocol's correctness. For basic definitions related to Petri nets the reader can refer to [13].

Algorithm 3 Algorithm executed by a node i, describing the isolation approach

```
When receive notification from the monitor to isolate node X for type t (directed or broadcast) of packets
  WREQ.t=t
  Broadcast WREQ against X
When receive a WREQ sent by X against j:
  if (WREQ.t = directed packet) then
    if ($j \in$ the suspects' set or $nbrcontroldrp_j \approx threshold$) then
      send a direct signed WREP to X
    else
      if (j is a neighbor of i) then
        if (a packet from X to j was overheard as well as the ACK) then
          send ACREQ toward j's successor using a route that does not include j
        end if
      end if
    end if
  else {The WREQ is about a broadcast packet}
    if (j is a neighbor and neither RREQ nor RREP has been received from j) then
      send a direct signed WREP to X
    end if
  end if
When receive a ACREQ sent by Y against j where X is the previous hop: {Investigator}
  if (no packet has been recently forwarded from j including X as the previous hop) then
    send Y a ACREP
  end if
When receive a ACREP regarding X accusation: {Investigator}
  send X a signed undirect WREP
When receive a WREP sent by X against j:
  if (WREP.type = direct) then
    $nbrdirectwit[j] = nbrdirectwit[j] + 1$
  else
    $nbrundirectwit[j] = nbrundirectwit[j] + 1$
  end if
  $WitnessesSet[j] = WitnessesSet[j] \bigcup (X, Singature_X)$
  if ($nbrdirectwit[j] + nbrundirectwit[j] = k$ and $nbrdirectwit[j] > 0$) then
    Construct AC using WitnessesSet[j]
    broadcast AC to isolate j
  end if
```

4.1.1. *Notations and definitions*

In the rest of this section the following notations are used:

- $\langle P, T, I, O, M_0 \rangle$: a 5-tuple representing a Petri net, where; P and T are respectively the sets of places and transitions, I and O are respectively the Input and Output functions (matrices), and M_0 is the initial marking
- M^t: denotes the transpose matrix of matrix M
- $|x|$: the absolute value of x
- $\lceil x \rceil$: the upper integer part of x defined by:
  $\forall x \in \mathbb{R}, \lceil x \rceil = m \in \mathbb{N} : m \geq x$ and $m - 1 < x$.
- $\|S\|$: the cardinality of the set S, i.e the number of S's elements
- $M(t > M'$: t is enabled from the marking M, its firing leads to the marking M', t may be either a single transition or a sequence of transitions, i.e $t \in T^*$
- $R(\langle Net, M \rangle)$: the set of marking reachable from the marking M in the Petri net $Net\langle P, T, I, O \rangle$, formally speaking:
  $R(\langle Net, M \rangle) = \{M_R \in \mathbb{N}^{\|P\|} : \exists (t_1, \cdots, t_n) \in T^n, M(t_1 \cdots t_n > M_R\}$

In the following, we define some concepts used later

Definition 4.1. (sink state)
A marking M is called a sink state iff it enables no transition, i.e it fulfils:
$\forall T_i \in T$, $\exists p \in P$ such that $M(p) < I(p, T_i)$ [13].

Definition 4.2. (Host state)
A host state M_h is a marking reachable from any marking M reachable from the initial marking M_0, formally speaking:
M_h is a host state iff $\forall M \in R(\langle Net, M_0\rangle)$, $M_h \in R(\langle Net, M\rangle)$ [13].

Definition 4.3. A norm for a marking M_a is an application v from the markings set to $\mathbb{N}$, such that $\forall M \in R(\langle Net, M_0\rangle)$, v fulfils the following conditions:
i) $v(M) = 0 \Leftrightarrow M = M_a$
ii) $v(M) \neq 0 \Rightarrow \exists M' \in R(\langle Net, M\rangle)$ such that $v(M') < v(M)$ [13].

4.1.2. *The model*

As seen previously, in the proposed approach each node monitors the next hop forwarding of each packet it sends. When a node A (which may be either the source our an intermediate node) sends packets to B, it monitors B's forwarding to C. This concept was generalized all along the path from the source to the destination. To prove that the protocol does what it has to do (detects the packets dropped), we have just to prove that when the monitoring node A sends n packets to B, and if B drops m out of these n packets then A validates **exactly** n-m forwarding, and thereby detects the m packets dropping. In this proof, channels are assumed to be reliable, i.e. all packets sent will be correctly received at the recipient, and the sole cause of packets dropping is nodes misbehavior. the protocol is modeled by the following Petri net: $Net_0 = \langle P, T, I, O, M_0\rangle$
P=$\{P_0$, P_1,..........,$P_{10}\}$
P_0: buffer of packets to send by A
P_1: buffer of packets dropped at B
P_2: buffer of packets that are being monitored by A, i.e tokens in this place represents entries in the A's Wait2HopsACK buffer
P_3: buffer of packets whose forwarding has been validated, i.e tokens in this place represents entries removed from the A's Wait2HopsACK buffer
P_4: buffer of valid two-hop ACKs sent (forwarded) from B to A, which have not been treated by A
P_5: buffer of packets sent from A to B, not already received by B
P_6: buffer of packets received by B, not already treated
P_7: buffer of packets forwarded by B to C, not already received by C
P_8: buffer of packets received by C, not already treated
P_9: buffer of a two-hop ACKs packets sent by C, not already received by B
P_{10}: buffer of two-hop ACKs packets received by B

T={T_0, T_1,..........,T_8}
T_0: node A sends a packet to B
T_1: B receives a packet from A
T_2: A validates a packet forwarding and removes the appropriate entry from its Wait2HopsACK buffer
T_3: B forwards a packet
T_4: B drops a packet
T_5: C receives a packet from B
T_6: C sends a two-hop ACKs packet
T_7: B receives a two-hop ACKs packet
T_8: B forwards a two-hop ACKs packet

M$_0$ is represented by the following vector: $[n\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0]^t$.

Figure 2 illustrates this Petri net. I and O could be easily deduced from this graph.

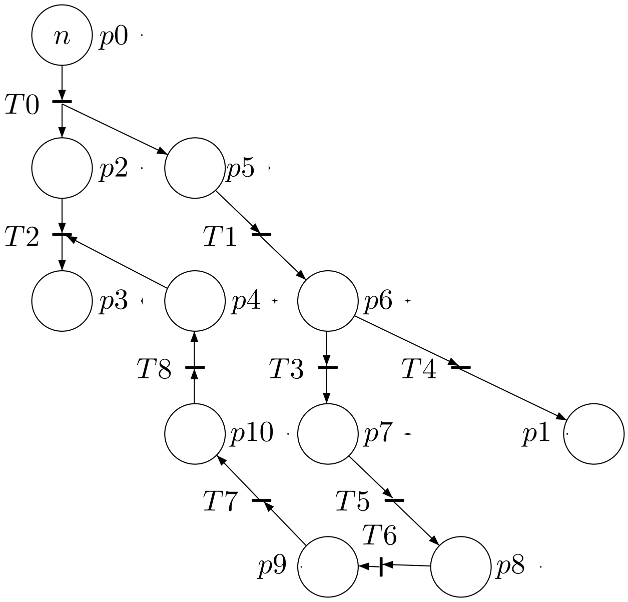


Fig. 2. The initial petri net.

4.2. *Proof overview*

Theorem 4.1. Let n be the number of packets sent by A and m the number of packets dropped by B. Our protocol ensures the following:
$\forall n \in \mathbb{N}, \forall m \in \mathbb{N}, m \leq n$, A validates exactly n-m packets forwarding.

Since each packet is monitored by A and is associated with a timeout, supposed to be great enough to the required time for receiving the two-hop ACKs related to the packet's forwarding, if the packet's forwarding is not validated (the packet has not been removed from the buffer up to a timeout), then it will be supposed dropped and will cause the B's rating increase. Validating exactly n-m packets is equivalent to detect m B's dropping. Consequently, this theorem show the protocol correctness. We bring the problem to our Petri net model, and propose the following lemma:

Lemma 4.1. $\forall n \in \mathbb{N}, M_f = \left[0\; m\; m\; n-m\; 0\; 0\; 0\; 0\; 0\; 0\; 0\right]^t, m \leq n$*, is a sink host state to* $(Net0, M_0)$.

If so, we realize that our system terminates at M_f. In other words, whatever the sequence of transitions fired, M_f will be reached (host state property), and no transition will be enabled from this marking (sink state property).
The semantic behind this, according to our model, is that when B drops m packets out of n, A will *inevitably* validate *exactly* n-m B's forwarding. This because M_f reaching means: the number of packets dropped by B (tokens in p_1) is m, the number of packets whose forwarding is validated by A (tokens in p_3) is n-m, and no other validation will take place since M_f is a sink state.
We have just to prove this lemma to conclude the previous theorem.

4.2.1. *Net reduction*

The initial Petri net can be reduced using the *place substitution* [13]. Places $p_5...p_{10}$ are *substitutable*, the result of their substitutions is the net $Net_r\langle P_r, T_r, I_r, O_r, M_{r_0}\rangle$ illustrated in figure 3.

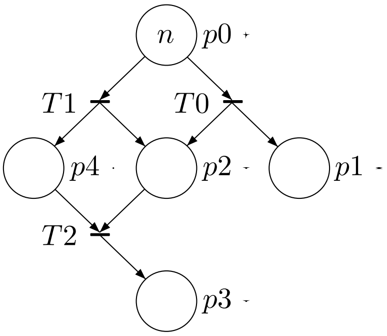


Fig. 3. The reduced petri net.

Although this reduction causes loss of details presented in the previous model, it helps reducing its size and facilitates our proof. Since the host state and the sink state properties are reducible by substitution [13], the lemma 4.1 is reducible to the following

Lemma 4.2. $\forall n \in \mathbb{N}, M_f = \left[0\; m\; m\; n-m\; 0\right]^t, m \leq n$*, is a sink host state to* (Net_r, M_{r_0}).

It is obvious that M_f is a sink state, since it enables no transition. To prove that it is a host state we use the following theorem, which exploits a linear algebra concept to built a sufficient condition regarding the host state existence.

Theorem 4.2. (host state using linear algebra)
If a marked Petri net admits a norm for a marking M, then M is a host state of this net [13].

4.2.2. *The norm*

We propose the following application that we will use in the next lemma:

$V : (R\langle Net_r, M_{r_0}\rangle \subset \mathbb{N}^5) \rightarrow \mathbb{N}$
$V(M) = X_0 + 3X_4 + \lceil |\log(\frac{X_0+\max(X_1,X_2)+1}{\min(X_1,X_2)+1})| \rceil + \lceil |\log(\frac{X_2+\min(n-X_1,X_3)+1}{X_0+X_2+\max(n-X_1,X_3)+1})| \rceil$
Such that M = $\begin{bmatrix} X_0 & X_1 & X_2 & X_3 & X_4 \end{bmatrix}^t$

To prove lemma 4.2, we use the following one:

Lemma 4.3. *V is a norm for M_f in Net_r*

We can prove this lemma by proving that V fulfills the two conditions of definition 4.3, see appendix.
From this lemma we realize the correctness of lemma 4.2, lemma 4.1, and theorem 4.1 □.
This proof relies on the assumption that no packet can be unintentional lost, and the only cause of this loss is intentional dropping. This shows the correctness of our solution in an ideal world, but in practice it is almost impossible to ensure this condition since packet loss due to channel conditions and nodes mobility in general MANETs is always possible, variant, and unpredictable. Consequentially, the problem is of high complexity and it is very difficult if not impossible to propose a deterministic and provable solution when considering this constraint, and false positives are always possible. This problem exists with our watchdog-like monitoring for broadcast packet, and all the current solution as well. Further, neither our solution nor any other in the context of packet dropper detection deal with collusive dropping. Collusive dropping is one of the complex misbehavior with which all the current solutions (including ours) fail. To explain this behavior consider the scenario of three successive nodes A, B, and C in this order, when B and C collude and B conceals the dropping of C. As a result no current monitoring solution enables A to detect this dropping, as they all rely on hop-by-hop distributed monitoring. It is very difficult to detect such a misbehavior owing to lack of central points.

4.3. *Isolation*

The aim of the isolation strategy is to punish misbehaving nodes, while reducing false positives of the monitor and mitigating rumors. Rumors refers to a possible DoS attack on our solution where the attacker claims another benign node as misbehaving, which is possible if the isolation is performed immediately upon one detection by one monitor. In our witness-based isolation 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 node unilaterally, because this can result in false detections against it. Upon the detection of a misbehaving node 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 suspect, or those that are monitoring it and whose misbehavior 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 the 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 merely do not testify. The following example illustrates and analyzes the investigation: Assume three aligned nodes, A, B and C, and another node D in the power range of A and B. 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, 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[b] 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 the 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.

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 accusations 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 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 unintentional false detections (false positives), since k nodes can collude to accuse maliciously (respectively wrongly) any node. This issue related to k will be investigated in the next section. Once the accuser collects

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

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.

5. Simulation Assessment

5.1. *Simulation setup*

Using an extended version of GloMoSim [14] (that includes our implementation) we simulated two kinds of packet dropping: RREQ dropping, which represents the selfish misbehavior and allows to evaluate our solution for broadcast packets, and RERR dropping that represents a malicious node behavior aiming a DoS attack, allowing to evaluate the solution with respect to directed packets.

We simulated a network of 50 nodes during 30 minutes simulation time, moving in an area of $1500 \times 1000m^2$ according to the random-waypoint model [14]. Each node has a power range of $250m$. We changed the node average speed (both the minimum and the maximum speed) from $0m/s$ to $4m/s$, i.e. for each speed configuration we fixed both the minimum and the maximum values to the appropriate value, to avoid uncontrollable change of the real average value [15]. For each value of mobility (speed), measurements for three different configurations of misbehavior were taken: i) low misbehavior rate with 5 misbehaving nodes, ii) medium rate with 12 misbehaving nodes, iii) and finally high rate in which 20 nodes misbehave. For each configuration five seeds were used, resulting in no less than 2000 scenarios. The curves presented hereafter represent the averaged values for these configurations. Note that each scenario causing considerable divergence from the reported average value was eliminated and replaced by an execution with another seed.

The simulation study is divided into two steps. The first one establishes the best values of the intrinsic parameters of our protocol. Afterwards, using these values we compare our protocol vs. DSR and the basic ENDAIRA, regarding both the efficiency and the cost. Table 1 lists the most important parameters of our simulation.

Parameter	**Value**
Number of nodes	50
Terrain	$1500 * 1000m^2$
Average speed(min=max)	$0m/s$ to $4m/s$ $pace = 0.5$
Pause-time	$0s$ to $8s$, $pace = 1$
Propagation model	free-space
MAC protocol	IEEE 802.11
Misbehavior	continuous dropping of RREQ and RRER
Traffic	CBR sessions (512byte/sec)

5.2. *Metric of comparison*

5.2.1. *True isolation rate*

The true isolation rate (TIR), or true positives, represents the efficiency on packet droppers *isolation*. It is the average rate of true isolations of nodes computed as follows:

$$TIR = \sum_{i=1, m_i \neq 0}^{n} \frac{trueisol_i / m_i}{k} \tag{5.1}$$

$trueisol_i$: is the true isolation of node i, i.e the number of misbehaving nodes monitored and detected by node i, then isolated in the network.
m_i: the number of misbehaving nodes monitored by node i.
n: the number of nodes.
k: the number of nodes that have monitored misbehaving nodes (whose $m_i \neq 0$).

5.2.2. *False isolation rate*

This metric (FIR) is very similar to the previous one. It is the average rate of false isolations, given by the following formula:

$$FIR = \sum_{i=1, m'_i \neq 0}^{n} \frac{falseisol_i / m'_i}{k'} \tag{5.2}$$

Where $falseisol_i$ is the false isolations of node i, viz the number of well-behaving nodes monitored and wrongly detected by node i and isolated, m'_i is the number of well-behaving nodes monitored by node i, and finally k' is the number of nodes that have monitored well-behaving nodes (whose $m'_i \neq 0$).

These metrics are involved in the two steps of the simulation. However, the next ones are used in the second step, as they illustrate the cost engendered by our solution.

5.2.3. *End-to-end delay*

This metric is defined as the average time separating the sending of a data packet from a source node and its arrival to the corresponding destination. Formally speaking:

$$delay = \frac{1}{nbRec} \sum_{i \in Rec} \sum_{j \in pr_i} \frac{delay_j}{nbpr_i} \tag{5.3}$$

Rec: is the set of *destination* nodes that received data packets. Nodes that did not receive any data packet are eliminated
$nbRec$: is the number of receiver nodes (‖ Rec ‖)
pr_i: is the set of packets received by node i as the final destination. Packets that did not arrive to their destination are eliminated.
$nbpr_i$: is the number of packets received (‖ pr_i ‖)
$delay_j$: is the transfer delay of packet j, such that:
$delay_j$ = packet j arrival time to its destination - packet j sending time by the source.

5.2.4. *Energy*

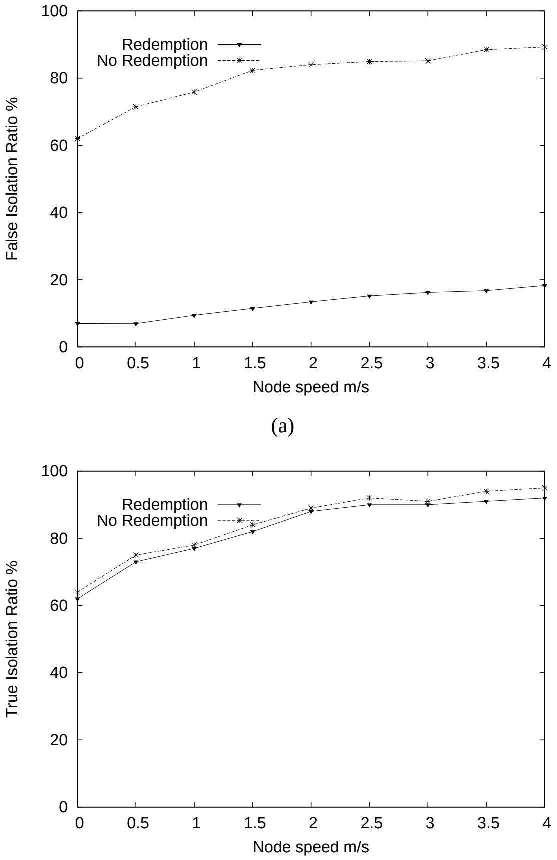
The average consumed power is defined as

$$average\_power = \sum_{i=1}^{n} \frac{PC_i}{n} \tag{5.4}$$

Such that PC_i is the power consumed by node i during the simulation, computed in GloMoSim using the NCR Wavelan radio model [14].

5.3. ***Simulation results***

5.3.1. *Best intrinsic parameter values*

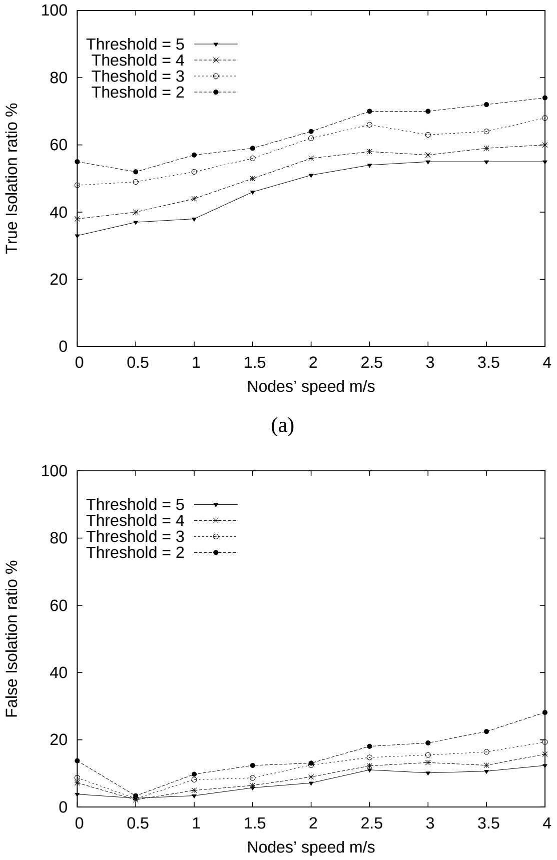


(a)

(b)

Fig. 4. Redemption Mechanism.

First, we investigate our redemption mechanism by simulating RREQ dropping in scenarios of our implemented protocol (enforced ENDAIRA) with and without redemption. Figures 4(a) and 4(b), representing respectively the false and true isolation ratios, show how the redemption approach hugely reduces the false isolation while keeping the true isolation close to the non-redemption version. It is clearly illustrated that the latter has unacceptable values of false isolations. In both figures we explain the increase of isolation ratios (false and true) with the mobility by the fact that it causes more link breakage, which engenders more RREQ retransmissions, thus more packets to monitor. Note that the redemption version is less affected by mobility with respect to false isolations compared to the other one. The increase of true isolations, however, reflects a good performance. Note that the redemption version in this step (RREQ dropping investigation) was performed by setting the redemption pace to 0.2, which gave in our scenarios the best performances (the highest true positives and the lowest false positives) compared to the other values.



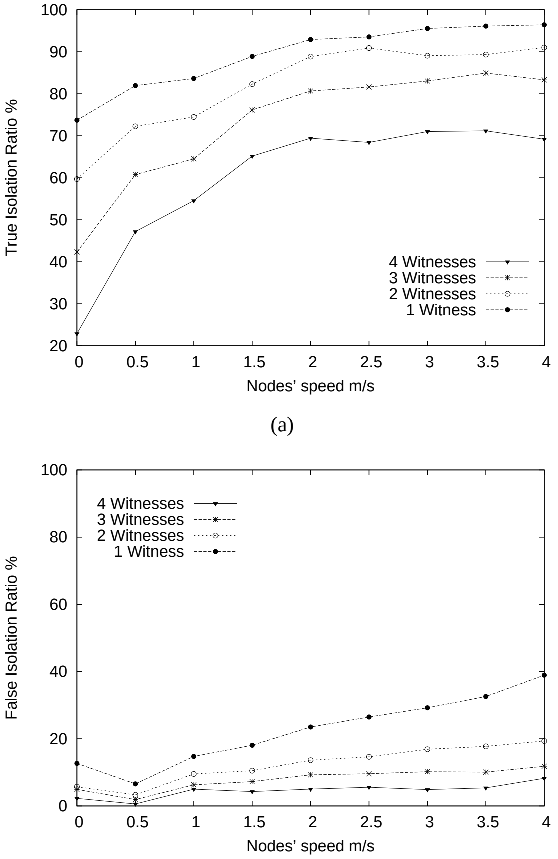
(a)

(b)

Fig. 5. Number of Packets Threshold (Selfish Behavior).

Now we try to find out the best values of the parameters for RREQ, namely the threshold number of packets upon which the node is accused and the number of witnesses, then we will deal with RERR packets.

From Figures 5(a) and 5(b) we realize that fixing this threshold to three strikes a balance between true and false isolations. The version representing this value has a very tolerable true detection, too close to the one representing the threshold value of two. The latter has high false isolations (up to more than 20%), while the others (threshold four and five) have too low true detections (lower than 50%). Note that the false isolations are largely affected by mobility. This impact will be reduced by fixing the other parameters, as it will be illustrated later.



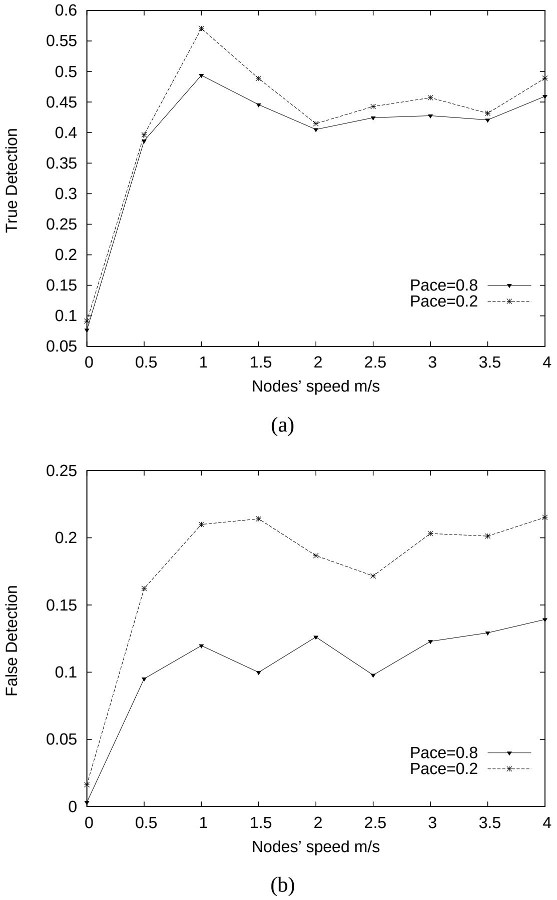
(a)

(b)

Fig. 6. Number of Witnesses (Selfish Behavior).

Now the previous threshold is fixed to 3, and try to find the best value for the number of witnesses. As depicted in figure 6(a) the version with two witnesses has good true isolation values, just a bit below the version with one witness and clearly above the others. On the other side, we remark in figure 6(b) that the values of the one-witness version is high and largely affected by mobility, contrary to the other ones. Overall, the version with two witnesses is the one that can be considered as well-balanced between true and false accusations. We carried out investigations in the same way to figure out the best value of the redemption pace. As reported earlier, this parameter was set to 0.2 ($2/5$). That is, for each five RREQ packets forwarded two dropping are forgotten.

Now we investigate the malicious RERR packet dropping. We can see in figures 7(a) and 7(b) that true positives of the versions with the redemption pace 0.8 and 0.2 are very close to each other, while false positives of the first version is clearly better than the other. The values go up with mobility for the same reason cited before. After fixing the value of this parameter to 0.8 we investigated the tolerance threshold. As depicted in figure 8(a), fixing the tolerance threshold to one gives much better true isolation rate compared to the other values. On the other hand, the difference regarding false isolations between all the versions is minor (figure 8(b)). Further, they are less affected by mobility. Therefor, this parameter is fixed to one.



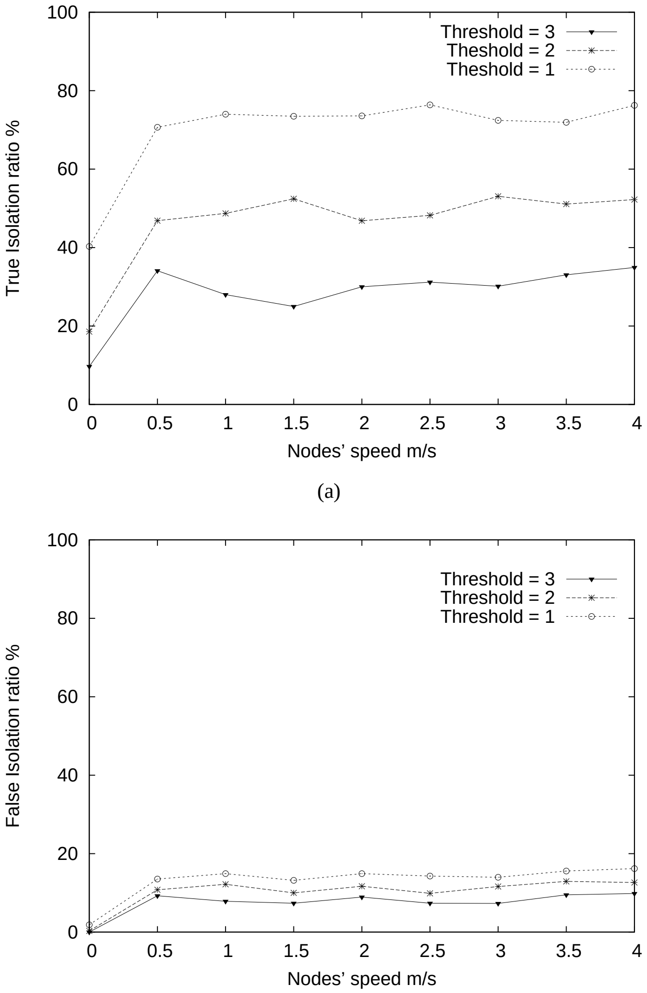
(a)

(b)

Fig. 7. Redemption Pace (Malicious Behavior).

Regarding the number of witnesses we can see that the version with three witnesses has low true isolations (figure 9(a)), while the version with one witness shows relatively high false isolations (figure 9(b)). We thus opt for two witnesses.

Table 2 illustrates the best values of the parameters for both directed (RREQ) and broad-



(a)

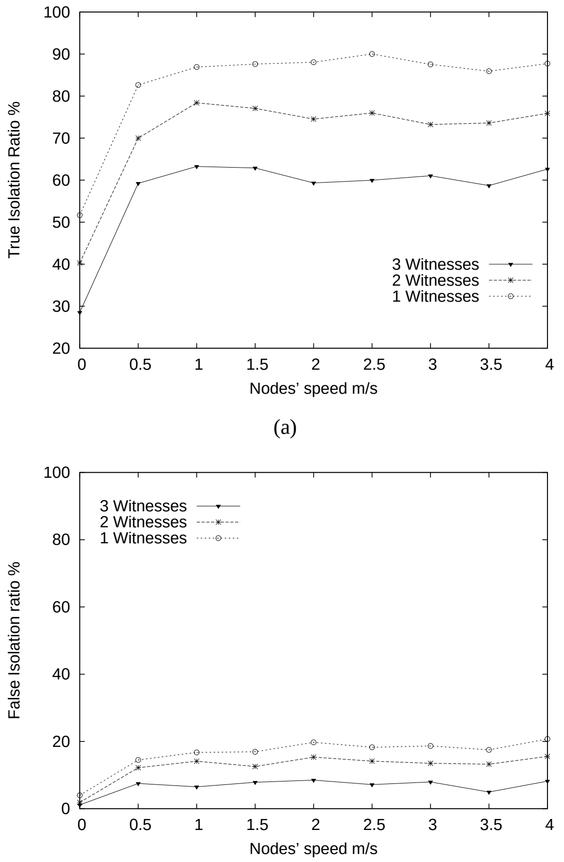
(b)

Fig. 8. Number of Packets Threshold (Malicious Behavior).

cast (RERR) packets. Investigations on RREP indicated that the best values for this kind of packets are the same as the ones of RERR.

Table 2. Best parameters' values.

	Parameter	Value
Broadcast packets	Tolerance threshold	3
	Number of witnesses	2
	Redemption pace	0.2
Directed packets	Tolerance threshold	1
	Number of witnesses	2
	Redemption pace	0.8



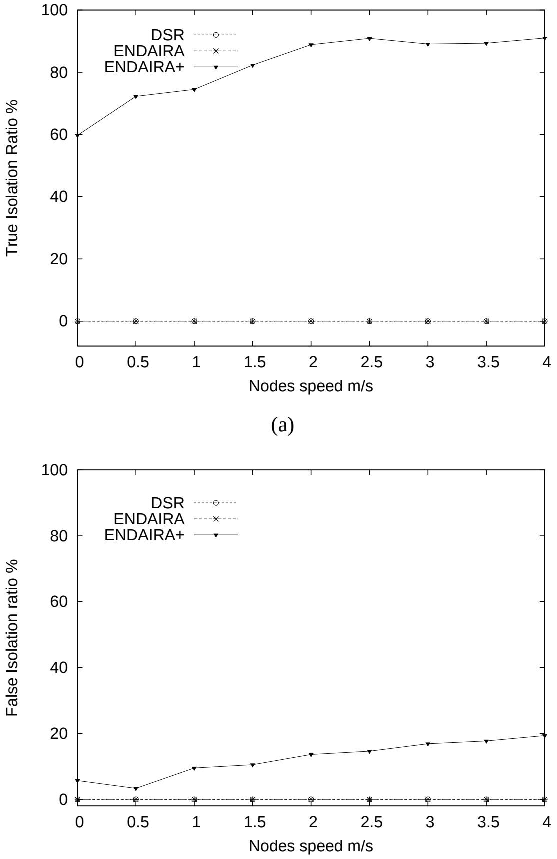
(a)

(b)

Fig. 9. Number of Witnesses (Malicious Behavior).

5.4. *Comparison*

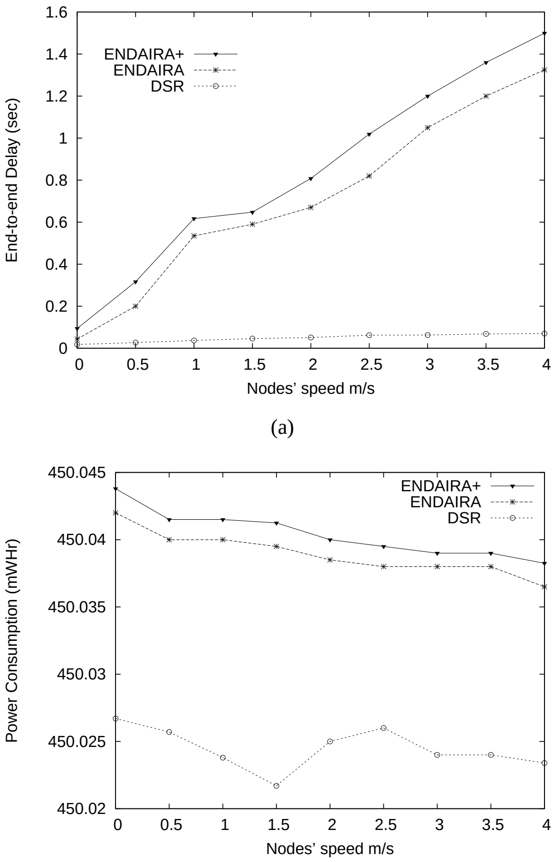
Now the previous parameters are set to their best values, and let us compare our protocol (denoted by ENDAIRA+) with the basic ENDAIRA and DSR in scenarios where misbehaving nodes drop RREQ packets. ENDAIRA+ clearly outperforms both DSR and ENDAIRA with respect to packet dropper isolation, since the latter protocols (DSR and ENDAIRA) simply do not detect such a misbehavior. Figure 10(a) shows how our protocol has high true isolations, especially when mobility goes up. On the other hand, figure 10(b) shows that the false isolation rate has been considerably reduced when fixing optimally the parameters, and more importantly that the protocol becomes less affected by mobility. The cost of this misbehavior detection is a small rise in both delay and power consumption, compared with ENDAIRA. For the delay, presented in figure 11(a), the big difference between DSR and the other secure protocols is basically due to cryptographic primitives (digital signatures computation on packets) used by the two secure protocols. The increase with mobility can be argued by the fact that mobility causes the launching of more route discoveries, thus more latency due to cryptography computation before sending data packets. The most important issue here is the minor difference between ENDAIRA and our protocol. This difference is due to the monitoring procedures. Finally, we observe almost the same differences regarding energy (figure 11(b)). The difference between the protocols is due to the overhead. The small difference between ENDAIRA and our protocol indicates that the cost of control packets added by the latter (overhead) is minor. The only difference between this figure and the previous one is the reduction of power consumption (especially for the secure protocols) with mobility. This is mainly due to the increase of the number of lost packets as mobility goes up. This reduction was not observed when measuring the delay because the lost packets are not used for the computation of this metric.



(a)

(b)

Fig. 10. True and False Isolation of Protocols.



(a)

(b)

Fig. 11. End-to-end Delay and Power Consumption of Protocols.

6. Related Work

Many secure routing protodcols have been recently proposed for MANET, aiming at preventing the establishment of falsified routes. SAR [7] is a general proposal that can be implemented with a reactive routing protocol. It defines the trust level that should be associated with each node, and ensures that a node is prevented from handling a RREQ (Route Request) unless it provides the required level. This way, data packets will be sent only through trusted nodes (with respect to the defined level). SAODV [7] is an implementation of SAR on AODV. One of the difficulties of this approach is the definition of the trust level. Further, assuming that nodes showing the required trust level are genuine is not always correct. SRP [16] is another secure routing protocol, based on DSR [17]. It prevents spoofing attacks, but is vulnerable to the wormhole attack [8] that also exists in ARAN [18]. ARIADNE [19] is another DSR-based protocol that overcomes this attack. There are different implementations of ARIADNE; the first one is based on TESLA, the second uses MACs (Message Authentication Codes), and the most sophisticated uses digital signatures. However, it has been illustrated that this protocol is vulnerable to some fabrication attacks causing the construction of nonexistent routes [8]. To mitigate this attack, ENDAIRA [8] has been proposed, then improved in [4]. This protocol is very similar to the last version of ARIADNE. As illustrated previously, the idea of the latest version of ARIADAN is simply to sign both RREP (route reply) and RREQ (route request) packets instead of merely RREQ ones. Despite of their advantages all these secure routing protocols do not handle packet dropping misbehavior, and hence are vulnerable to black hole attack and selfish behavior.

Watchdog (WD) [9], presented in section 2, is the first solution dealing with the packet dropping problem. Remember that this basic solution has the advantage of not causing any communication overhead as long as nodes behave well, and to be applicable both to data and control packets, but it is inappropriate when using the power control technique that is employed by some new power-aware routing protocols following the proposal of WD [11]. Furthermore, WD does not deal with the isolation step. When a misbehaving node is detected packets will be sent around it, but no measures will be taken against it. Therefore, WD by itself does not prevent nodes from misbehaving.

Many sophisticated solutions that deal with the post-detection issues rely on the WD approach for monitoring, such as CORE [5] and CONFIDANT [1]. These solutions define some reputation and punishment strategies. Since they are watchdog-based, they inherit all its monitoring drawbacks. Moreover, they require periodic exchange of reputation information, which is costly and unnecessary so long as nodes are benign. Another monitoring approach is the employment of a kind of ACK packets known as two-hop ACK [3]. An interesting optimization of this approach is the random asking strategy [12] that has been used in [2]. All these ACK-based solutions focus on data packets, and are not directly applicable to control packets, especially the broadcast ones. The main contribution of our paper is the dealing with control packets and proposing a hybrid solution that considers both directed and broadcast packets.

7. Conclusion and Perspectives

In this paper a general solution to secure routing protocols against packet dropping misbehavior in mobile ad hoc networks has been proposed. We focused on routing control packets, which have not received attention in literature compared to data packets. We adapt some approaches in a hybrid solution that applies to both directed and broadcast packets. The solution allows to monitor, detect, and isolate the droppers, and can be integrated with any source routing protocol. The solution has benn implemented with the secure routing protocol ENDAIRA, and made a comprehensive simulation study to first fix the crucial parameters of our solution to optimal values, and then to compared it with the basic protocols. Usually, the MANET nodes' mobility causes degradation in efficiency of protocols, but in our case we remarked that it helps improving the true positives of our solution. Nonetheless, we also remarked that it degrades false positives. After setting the parameters to the optimal values the latter metric becomes less affected by the increase of mobility, which renders the protocol adaptable to mobility. Compared with ENDAIRA, the cost of our protocol is small in both latency and power consumption. However, there is an important difference between the two secure protocols (ours and the basic ENDAIRA) and DSR. This difference is basically caused by the employment of digital signatures, which indeed are robust but costly. Implementing our solution with another lighter secure routing protocol could represent a perspective to this work. Proposing a solution for self setting the parameters according to the network configuration and conditions also represents a potential perspective.

Appendix A. Proof of Lemma 4.3

Appendix A.1. *The first condition*

$$\mathbf{V(M) = 0 \Leftrightarrow M = M_f} \tag{A.1}$$

remember that $M_f = \begin{bmatrix} 0 \; m \; m \; n-m \; 0 \end{bmatrix}^t, m \leq n$

i) $\underline{M = M_f \Rightarrow V(M) = 0}$: this implication is obvious, we need just to compute $V(M_f)$, to find that it equals to 0.

ii) $\underline{V(M) = 0 \Rightarrow M = M_f}$
Since V is the sum of four terms in $\mathbb{N}$, V(M)=0 $\Leftrightarrow \forall i \in 0, 1, 2, 3 \; Term_i = 0$

$$V(M) = 0 \Rightarrow \begin{cases} Term_0 = 0 \Leftrightarrow X_0 = 0 & \& \\ Term_1 = 0 \Leftrightarrow X_4 = 0 & \& \\ Term_2 = 0 \Leftrightarrow \log(\frac{X_0+\max(X_1,X_2)+1}{\min(X_1,X_2)+1}) = 0 & \& \\ Term_3 = 0 \Leftrightarrow \log(\frac{X_2+\min(n-X_1,X_3)+1}{X_0+X_2+\max(n-X_1,X_3)+1}) = 0 \end{cases}$$

$$V(M) = 0 \Rightarrow \begin{cases} X_0 = 0 & \& \\ X_4 = 0 & \& \\ \frac{\max(X_1,X_2)+1}{\min(X_1,X_2)+1} = 1 \Leftrightarrow \max(X_1, X_2) + 1 = \min(X_1, X_2) + 1 & \& \\ \frac{X_2+\min(n-X_1,X_3)+1}{X_2+\max(n-X_1,X_3)+1} = 1 \Leftrightarrow X_2 + \min(n - X_1, X_3) + 1 = X_2+ \\ \max(n - X_1, X_3) + 1 \end{cases}$$

$$V(M) = 0 \Rightarrow \begin{cases} X_0 = 0 & \& \\ X_4 = 0 & \& \\ \max(X_1, X_2) = \min(X_1, X_2) \Leftrightarrow X_1 = X_2 & \& \\ min(n - X_1, X_3) = \max(n - X_1, X_3) \Leftrightarrow X_3 = n - X_1 \end{cases}$$

V(M)=0 $\Rightarrow$ M= $\begin{bmatrix} 0 \; m \; m \; n-m \; 0 \end{bmatrix}^t, m \leq n \Rightarrow M = M_f \square$

Appendix A.2. *The second condition*

$$\mathbf{V(M) \neq 0 \Rightarrow \exists M' \in R(\langle Net, M_0 \rangle) : V(M') < V(M)} \tag{A.3}$$

case 1: $\mathbf{X_0 \neq 0}$
In this case T_0 is enable, we will prove that it leads to a marking M' holding the condition. Let us consider $M = \begin{bmatrix} X_0 \; X_1 \; X_2 \; X_3 \; X_4 \end{bmatrix}^t \in R(\langle Net_r, M_{0_r} \rangle)$, and $M(T_0 > M_{T0}$, then:
$M_{T0} = \begin{bmatrix} X_0 - 1 \; X_1 + 1 \; X_2 + 1 \; X_3 \; X_4 \end{bmatrix}^t$
$V(M) \neq 0$ is verified since $X_0 \neq 0$, we have to verify that $V(M_{T0}) < V(M)$

$V(M_{T0}) = \underbrace{X_0 - 1}_{\overline{Term_0}} + \underbrace{3X_4}_{\overline{Term_1}} + \overline{Term_2} + \overline{Term_3}$

We have: $\overline{Term_0} = Term_0 - 1 \Rightarrow \overline{Term_0} < Term_0$, and $\overline{Term_1} = Term_1$

Now, we try to prove that $\overline{Term_2} \leq Term_2$ and $\overline{Term_3} \leq Term_3$ to realize that $V(M_{T0}) < V(M)$

$\overline{Term_2} = \lceil |\log(\frac{X_0-1+\max(X_1+1,X_2+1)+1}{\min(X_1+1,X_2+1)+1})|\rceil = \lceil |\log(\frac{X_0-1+\max(X_1,X_2)+1+1}{\min(X_1,X_2)+2})|\rceil = \lceil |\log(\frac{X_0+\max(X_1,X_2)+1}{\min(X_1,X_2)+2})|\rceil$

Since: $1 \leq \frac{X_0+\max(X_1,X_2)+1}{\min(X_1,X_2)+2} \leq \frac{X_0+\max(X_1,X_2)+1}{\min(X_1,X_2)+1}$, we realize that $\overline{Term_2} \leq Term_2$, because $\log$ is an increasing function in the interval $[1, +\infty[$ as well as the absolute value and the upper integer part functions.

It remains the last terms $Term_3, \overline{Term_3}$.

$\overline{Term_3} = \lceil |\log(\frac{X_2+1+\min(n-X_1-1,X_3)+1}{X_0+X_2+\max(n-X_1-1,X_3)+1})|\rceil$

In one hand, we have:

$\min(n - X_1 - 1, X_3) \geq \min(n - X_1, X_3) - 1 \Rightarrow$

$$X_2 + 1 + \min(n - X_1 - 1, X_3) \geq X_2 + \min(n - X_1, X_3) \tag{A.4}$$

On other hand:

$$X_0 + X_2 + \max(n - X_1, X_3) + 1 \geq X_0 + X_2 + \max(n - X_1 - 1, X_3) + 1 \tag{A.5}$$

From A.4 and A.5 it results: $\frac{X_2+1+\min(n-X_1-1,X_3)+1}{X_0+X_2+\max(n-X_1-1,X_3)+1} \geq \frac{X_2+\min(n-X_1,X_3)+1}{X_0+X_2+\max(n-X_1,X_3)+1}$

Since: $\frac{X_2+1+\min(n-X_1-1,X_3)+1}{X_0+X_2+\max(n-X_1-1,X_3)+1} \leq 1$, and $|log(x)|$ is a decreasing function on $[0, 1]$, we realize that $\overline{Term_3} \leq Term_3$:

We have proved that: $\overline{Term_2} \leq Term_2$ and $\overline{Term_3} \leq Term_3$. Moreover, we have $\overline{Term_0} < Term_0$ and $\overline{Term_1} = Term_1$. Consequently, $V(M_{T0}) < V(M)$ then the condition cond2 is fulfilled, (we have just to take $M' = M_{T0}$)

case 2: $\mathbf{X_0 = 0}$

in this case both T_0 and T_1 are disable.

case 2.1: when $\mathbf{T_2}$ is enable $(X_2 \geq 1, X_4 \geq 1)$

As in case 1, $V(M) \neq 0$ is verified. We will perform in the same way and try to prove that $V(M_{T2}) \neq V(M)$, such that $M(T_2 > M_{T2}$, $M_{T2} = \begin{bmatrix} 0 \; X_1 \; X_2 - 1 \; X_3 + 1 \; X_4 - 1 \end{bmatrix}^t$

$V(M_{T2}) = 3(X_4 - 1) + \lceil |\log(\frac{\max(X_1,X_2-1)+1}{\min(X_1,X_2-1)+1})|\rceil + \lceil |\log(\frac{X_2-1+\min(n-X_1,X_3+1)+1}{X_2-1+\max(n-X_1,X_3+1)+1})|\rceil$

$$= \underbrace{-1}_{\overline{Term_0}} + \underbrace{3X_4}_{\overline{Term_1}} + \underbrace{\lceil |\log(\frac{\max(X_1, X_2 - 1) + 1}{\min(X_1, X_2 - 1) + 1})|\rceil - 1}_{\overline{Term_2}} +$$

$$\underbrace{\lceil |\log(\frac{X_2 - 1 + \min(n - X_1, X_3 + 1) + 1}{+X_2 - 1 + \max(n - X_1, X_3 + 1) + 1})|\rceil - 1}_{\overline{Term_3}}$$

As in case 1, we have $\overline{Term_0} < Term_0$ (since $Term_0 = 0$), and $\overline{Term_1} = Term_1$. We have just to prove that $\overline{Term_2} \leq Term_2$ and $\overline{Term_3} \leq Term_3$

$\overline{\mathbf{Term_2}} \leq \mathbf{Term_2}$:
Depending on values of X_1, we can distinguish two cases; $X_1 = 0$ and $X_1 \geq 1$. Note that $X_2 \geq 1$ in the two cases (as $\mathbf{T_2}$ is enable). In the following we prove that the above inequality is held in the two cases:
When $\mathbf{X_1} = \mathbf{0}$: by replacing X_1 with 0 in the expressions of the two terms we get: $\overline{Term_2} = \lceil |\log(X_2)| \rceil$ and $Term_2 = \lceil |\log(X_2+1)| \rceil$. It is obvious that $\lceil |\log(X_2)| \rceil \leq \lceil |\log(X_2+1)| \rceil$, as $X_2 \geq 1$.
When $\mathbf{X_1} \geq \mathbf{1}$: using the property: $\log(a/b) = \log(b) - \log(b)$, we will have:
$Term_2 = \lceil |\log(\max(X_1, X_2)+1) - \log(\min(X_1, X_2)+1)| \rceil$
$\overline{Term_2} = \lceil |\log(\max(X_1, X_2-1)+1) - \log(\min(X_1, X_2-1)+1)| \rceil$
$\max(X_1, X_2-1) \leq \max(X_1, X_2) \Rightarrow \max(X_1, X_2-1)+1 \leq \max(X_1, X_2)+1 \Rightarrow$

$$\log(\max(X_1, X_2-1)+1) \leq \log(\max(X_1, X_2)+1) \tag{A.6}$$

The last implication is justified by the fact that $\max(X_1, X_2-1)+1 \geq 1$ and the function $\log$ is increasing in the interval $[1, +\infty]$.
$\min(X_1, X_2-1) \geq \min(X_1, X_2) - 1 \Rightarrow \min(X_1, X_2-1)+1 \geq \min(X_1, X_2) \Rightarrow \log(\min(X_1, X_2-1)+1) \geq \log(\min(X_1, X_2)) \Rightarrow$

$$\log(\min(X_1, X_2-1)+1)+1 \geq \log(\min(X_1, X_2)+1)+1 \tag{A.7}$$

As $\log(x)+1 \geq log(x+1)$ when $x \geq 1$, $\log(\min(X_1, X_2))+1 \geq \log(\min(X_1, X_2)+1)$, since $X_1, X_2 \geq 1$, thus
A.7 $\Rightarrow \log(\min(X_1, X_2-1)+1)+1 \geq \log(\min(X_1, X_2)+1) \Rightarrow$

$$-\log(\min(X_1, X_2-1)+1) \leq -\log(\min(X_1, X_2)+1)+1 \tag{A.8}$$

A.6 and A.8 $\Rightarrow$ $\log(\max(X_1, X_2-1)+1) - \log(\min(X_1, X_2-1)+1) \leq \log(\max(X_1, X_2)+1) - \log(\min(X_1, X_2)+1)+1$
$\Rightarrow \lceil |\log(\max(X_1, X_2-1)+1) - \log(\min(X_1, X_2-1)+1)| \rceil \leq \lceil |\log(\max(X_1, X_2)+1) - \log(\min(X_1, X_2)+1)| \rceil + 1$
$\Rightarrow \lceil |\log(\max(X_1, X_2-1)+1) - \log(\min(X_1, X_2-1)+1)| \rceil - 1 \leq \lceil |\log(\max(X_1, X_2)+1) - \log(\min(X_1, X_2)+1)| \rceil \Rightarrow \overline{Term_2} \leq Term_2$
In the same way, $\overline{Term_3} \leq Term_3$ can be proved.
As in case 1, the condition A.3 is fulfilled (the existence of M' is justified by $M' = M_{T2}$).

case 2.2: when $\mathbf{T_3}$ is disable
We will gradually prove that the unique form of the marking reachable from M_{0_r} and representing such a case in the same time is M_f. Let M be The markin representing case 2.2, $M = \begin{bmatrix} 0 & X_1 & X_2 & X_3 & X_4 \end{bmatrix}^t \in R(\langle Net_r, M_{0_r} \rangle)$

i) $\mathbf{X_4} = \mathbf{0}$
T_3 is disable $\Rightarrow$ ($X_2 = 0$ or $X_4 = 0$).
We will prove: ($X_2 = 0 \Rightarrow X_4 = 0$), to conclude that $X_4 = 0$ in this case (case 2.2),

because when $X_2 \neq 0$, X_4 must be 0 to disable T_3.
We prove this using the reductio ad absurdum, assume $X_2 = 0$ and $X_4 \neq 0$, that is $M = \begin{bmatrix} 0 \; X_1 \; 0 \; X_3 \; X_4 \end{bmatrix}^t \in R(\langle Net_r, M_{0_r} \rangle), X_4 \neq 0$
According to theorem 1, $\forall v \in \{{C_r}^T Y = 0\}, (M - M_{0_r}) \perp v$

$$ {C_r}^T Y = 0 \Leftrightarrow \begin{bmatrix} -1 & -1 & 0 \\ 1 & 0 & 0 \\ 1 & 1 & -1 \\ 0 & 0 & 1 \\ 0 & 1 & -1 \end{bmatrix}^t \times \begin{bmatrix} Y_0 \\ Y_1 \\ Y_2 \\ Y_3 \\ Y_4 \end{bmatrix} = 0 $$

$$ \Leftrightarrow \begin{bmatrix} -1 \, 1 & 1 \, 0 & 0 \\ -1 \, 0 & 1 \, 0 & 1 \\ 0 \, 0 & -1 \, 1 & -1 \end{bmatrix} \times \begin{bmatrix} Y_0 \\ Y_1 \\ Y_2 \\ Y_3 \\ Y_4 \\ \\ \end{bmatrix} = 0 $$

$$ \Leftrightarrow \left\{ \begin{array}{l} -Y_0 + Y_1 + Y_2 = 0 \; \& \\ -Y_0 + Y_2 + Y_4 = 0 \; \& \\ -Y_2 + Y_3 - Y_4 = 0 \end{array} \right\}. \text{ (sys1)} $$

$(M - M_{0_r}) \perp Y \Leftrightarrow (M - M_{0_r})^t Y = 0 \Leftrightarrow$

$$ -nY_0 + X_1Y_1 + X_3Y_3 + X_4Y_4 + = 0 \tag{A.9} $$

a) when $X_3 \neq n$, we remark that $\Omega_0 = \begin{bmatrix} 1 \; 0 \; 1 \; 1 \; 0 \end{bmatrix}^t \in \{{C_r}^t Y = 0\}$ (it is a solution to sys1)
Ω_0 fulfills A.9 $\Rightarrow X_3 = n$, which represents a contradiction
b) when $X_3 = n$, we remark that $\Omega_1 = \begin{bmatrix} 2 \; 1 \; 1 \; 2 \; 1 \end{bmatrix}^t \in \{{C_r}^t Y = 0\}$ (it is a solution to sys1)
Ω_1 fulfills A.9 $\Rightarrow X_1 + X_4 = 0 \Rightarrow X_1 = -X_4 \Rightarrow X_1 < 0$ (since $X_4 \neq 0$) which represents a contradiction.
Whatever the values of X_3 and n, ($X_4 \neq 0$ and $X_2 = 0$) leads to contradictions, thereby: $X_2 = 0 \Rightarrow X_4 = 0$, hence, $X_4 = 0$.
$M = \begin{bmatrix} 0 \; X_1 \; X_2 \; X_3 \; 0 \end{bmatrix}^t$

ii) $\mathbf{X_1 = X_2}$
$(M - M_{0_r}) \perp Y \Leftrightarrow (M - M_{0_r})^t Y = 0 \Leftrightarrow$

$$ -nY_0 + X_1Y_1 + X_2Y_2 + X_3Y_3 = 0 \tag{A.10} $$

$\Omega_2 = \begin{bmatrix} 0 \; 1 \; -1 \; 0 \; 1 \end{bmatrix}^t \in \{{C_r}^t Y = 0\}$ (it is a solution to sys1)
Ω_2 fulfills A.10 $\Rightarrow X_1 = X_2$
M= $\begin{bmatrix} 0 \; X \; X \; X_3 \; 0 \end{bmatrix}^t$

iii) $\mathbf{X_3 = n - X}$

$$(M - M_{0_r}) \perp Y \Leftrightarrow (M - M_{0_r})^t Y = 0 \Leftrightarrow$$
$$-nY_0 + XY_1 + XY_2 + X_3Y_3 = 0 \qquad \text{(A.11)}$$

$\Omega_3 = \begin{bmatrix} 1\ 0\ 1\ 1\ 0 \end{bmatrix}^t \in \{{C_r}^t Y = 0\}$ (it is a solution to sys1)
Ω_3 fulfills A.11 $\Rightarrow X_3 = n - X$
We realize that M= $\begin{bmatrix} 0\ X\ X\ n-X\ 0 \end{bmatrix}^t$ in this case (case 2.2)
Therefore, $V(M) = 0$, which means that $V(M) \neq 0$ is false. Consequently, the condition A.3 is satisfied □

References

1. S. Buchegger and J.-Y. Le-Boudec, "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.
2. D. Djenouri and N. Badache, "Struggling against selfishness and black hole attacks in manets," *Wirless Communications and Mobile Computing (WCMC), Wiley & sons Publisher*, vol. 8, no. 6, pp. 689–704, 2008.
3. D. Djenouri and N. Badache, "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.
4. G. Acs, L. Buttyan, and I. Vajda, "Provably secure on-demand source routing in mobile ad hoc networks," *IEEE Transactions on Mobile Computing*, vol. 5, no. 11, pp. 1533–1546, 2006.
5. P. Michiardi and R. Molva, "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.
6. D. Djenouri, L. Khalladi, and N. Badache, "A survey of security issues in mobile ad hoc and sensor networks," *IEEE Communications Surveys*, vol. 7, no. 4, pp. 2–28, 2005.
7. S. Yi, P. Naldurg, and R. Kravets, "Security-aware ad hoc routing for wireless networks," in *The ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHOC01)*, Long Beach, CA, October 2001.
8. L. Buttyan and I. Vajda, "Towards provable security for ad hoc routing protocols," in *The ACM Workshop on Security in Ad Hoc and Sensor Networks SASN04*, Washington DC, October 2004.
9. S. Marti, T. Giuli, K. Lai, and M. Baker, "Mitigating routing misbehavior in mobile ad hoc networks," in *ACM Mobile Computing and Networking, MOBICOM 2000*, Boston, MA, USA, 2000, pp. 255–65.
10. S. Doshi and T. Brown, "Minimum energy routing schemes for a wireless ad hoc network," in *The 21st IEEE Annual Joint Conference on Computer Communications and NetworkingINFOCOM'02*, New York, USA, 2002.
11. D. Djenouri and N. Badache, "New power-aware routing for mobile ad hoc networks," *The International Journal of Ad Hoc and Ubiquitous Computing (Inderscience Publisher)*, vol. 1, no. 3, pp. 126–136, 2006.
12. D. Djenouri, N. Ouali, A. Mahmoudi, and N. Badache, "Random feedbacks for selfish nodes detection in mobile ad hoc networks," in *The 5th IEEE International Workshop on IP Operations and Management, IPOM'05*, ser. LNCS, no. 3751. Barcelona, Spain: Springer-Verlag GmbH, October 2005, pp. 68–75.
13. G.W.BRAMS, *Rseau de petri: Thorie et pratique*. Edition masson, 1983.

14. X. Zeng, R. Bagrodia, and M. Gerla, "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.
15. J. Yoon, M. Liu, and B. Noble, "Random waypoint considered harmful," in *The 22nd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM)*, San Francisco, CA, USA, 2003, pp. 1312–1321.
16. P. Papadimitratos and Z. Haas, "Secure routing for mobile ad hoc networks," in *The SCS Communication Networks and Distributed Systems Modeling and Simulation Conference CNDS02*, San Antonio, Texas, 2002.
17. B. David and A. David, "Dynamic source routing in ad hoc wireless networks," in *Mobile Computing*, imielinskiand and korth ed. Kluwer Academic, 1996, vol. 353, pp. 153–181.
18. B. Dahill, B. N. Levine, E. Royer, and C. Shields, "Aran: A secure routing protocol for ad hoc networks," University of Massachusetts, Amherst, Tech. Rep. UMass Tech Report 02-32, 2002.
19. Y.-C. Hu, A. Perrig, and D. B. Johnson, "Ariadne:: a secure on-demand routing protocol for ad hoc networks," in *The 8th annual international conference on Mobile computing and networking MobiCom '02*. ACM Press, 2002, pp. 12–23.