A Novel Approach for Selfish Nodes Detection in MANETs: Proposal and Petri Nets Based Modeling

Djamel Djenouri [*], Nadjib Badache [§]
[*] CERIST, Basic Software Laboratory, Algiers, Algeria
[§]USTHB, Computer Science Department, Algiers, Algeria
ddjenouri@mail.cerist.dz, badache@wissal.dz

***Abstract*— The resource limitation of nodes used in *self-organized ad hoc networks*, along with the multi-hop nature of these networks, may cause a new behavioral phenomena which does not exist in traditional infrastructured environments. To save its energy, a node may behave *selfishly*, then it would not forward packets for other nodes while using their resources to forward its own packets. This deviation from the normal behavior is a potential threat against the service *availability*, one of the most important security requirements. Resolving this problem is challenging, due to the self-organization and the infrastructureless features of these networks.**
Recently, some solutions have been proposed, but almost all these solutions rely on the watchdog technique [1], which suffers from many problems. Especially when using the power control technique, employed by some new power-aware routing protocols following the watchdog's proposal. To overcome this problem, we propose in this paper a new approach we call *two hops ACK* (acknowledgment). Using petri nets we model and analyze our solution based on this novel approach.

I. Introduction

A mobile ad hoc network, or a MANET, is a temporary self-organized infrastructureless network, formed by a set of mobile hosts that dynamically establish their own network *on the fly*, without relying on any central administration. Mobile hosts used in MANET have to ensure the functions ensured by the powerful fixed infrastructure in traditional networks, such as packet forwarding. When a destination node is unreachable from a source node, this latter asks other intermediaries to forward packets towards the finale recipient, this way packets follow *multi-hop* routes and travels several wireless links and mobile nodes.

In some MANETs applications, like networks of battlefields or rescue operations, all nodes have a common goal and their applications belong to a single authority. For this reason, nodes are *cooperative by nature*. However, in many civilian applications, such as networks of cars and provision of communication facilities in remote areas, nodes typically do not belong to a single authority and do not pursue a common goal. In such networks, forwarding packets for other nodes is not in the direct interest of any node, thus there is no good reason to trust nodes and assume that they always cooperate. Indeed, nodes try to save resources, particulary their batteries which are precious resources. Recent studies show that most of nodes' energy in MANETs is likely to be devoted to forward packets in behalf of other nodes. For instance, Buttyan and Hubaux simulation study [2] shows that when the average number of hops from a source to a destination is around 5, then almost 80% of transmission energy will be devoted to packet forwarding.

Therefore, the nodes may misbehave and tend to be *selfish*, aiming at saving their energy. A selfish node regarding the packet forwarding process is a node which takes advantage of the forwarding service, and asks others to forward its own packets, while not cooperating to forward packets originated from other nodes. Some solutions have been recently proposed, but almost all these solutions, however, rely on the watchdog technique [1] which suffers from many problems, especially when employing the power control technique. To basically mitigate this problem, we propose a new technique which we call two hops ACK (acknowledgment). Basing on this technique, we define a protocol that we will model and verify its correctness using a petri net.

The remainder of this paper is organized as follows: In the next section we present the related work, and we briefly present the watchdog technique in section 3. Section 4 is devoted to the presentation of our solution, followed by the modeling and the correctness proof overview in section 5. Finally, section 6 concludes the paper and summarizes the future work.

II. Related work

The emergent problem of selfishness on packet forwarding in MANET has recently received attention among researchers, and some solutions have been proposed. In [3], we have surveyed these solution and classified theme into two main categories: reactive solutions that aim at detecting the misbehavior when it appears in the network, and preventive solutions which try to inhibit the misbehavior, either by motivating nodes to cooperate or by taking measures to prevent packets from being dropped. To the best of our knowledge, Marti et al. are the firsts who dealt with this emergent problem. In [1], they define two techniques which they call *watchdog* and *pathrather*, the former is to identify misbehaving nodes, whereas the latter helps the routing protocol to avoid these nodes. These techniques are used along with DSR [4] to built a misbehavior mitigating routing protocol. The watchdog has been used by almost all the subsequent proposed reactive solutions. Nevertheless, this technique has many drawbacks, as it will be shown in the next section.

In [5], Yang et al. describe a unified network layer solution, to protect both routing and data forwarding in the context of AODV [6]. Michiardi and Molva [7] suggest a generic reputation-based mechanism, namely CORE (COllaborative REputation Mechanism to enforce node cooperation in MANETs), it is supposed to be easily integrated with any network function. Another interesting reputation-based solution has been proposed by Buchegger and Le Boudec [8]; the protocol called CONFIDANT (Cooperation Of Nodes Fairness in Dynamic Ad hoc Networks). This protocol relies on the DSR [4] routing protocol, which is used as benchmark in their GloMosim-based simulation study, performed to evaluate the new DSR fortified by CONFIDANT.
All these solutions, however, rely on the watchdog technique, and inherit all its problems.

Buttyan and Hubaux [9] propose a preventive economic-based approach which stimulates the nodes to cooperate, this solution is modelized and analyzed in [2]. They introduce what they call *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. Another preventive mechanism is the game theory approach. In [10], Srinivasan et al. propose a solution based on this approach. All these preventive solutions just motivate nodes to cooperate, but do not aim at detecting the misbehaving nodes.

In [11], Papadimitratos and Haas present the SMTP protocol. A hybrid solution that prevents the selfishness effects (packets lost) by dispersing packets, and detects it by employing the end-to-end feedbacks. This kind of feedbacks allows detection of routes containing selfish nodes, but fails to detect these nodes.

III. Overview of the watchdog

The watchdog method is a basic technique on which many further solutions rely. It aims to detect misbehaving nodes that do not forward packets, by monitoring neighbors in the promiscuous mode.

When a node A transmits a packet to B to forward to C, A monitors B's forwarding by promiscuously overhearing the packets sent in its neighborhood. The watchdog is implemented at A by maintaining a buffer of recently sent packets, and comparing each overheard packet with the packets in the buffer to check if there is a match. If so, the packet in the buffer is removed and forgotten by the watchdog, since it has been forwarded. If a packet has remained in the buffer for longer than a certain timeout, the watchdog increments a *failure tally* of the node responsible for forwarding the packet. If the tally exceeds a certain threshold, the monitor (node A) considers B as misbehaving, and sends a message to the source notifying it of the misbehaving node. The watchdog technique supposes that each transmission can be overheard by all neighbors, if no collision takes place. This solution requires no overhead, and it is a relevant solution for detecting selfish nodes when the previous assumption is held.

Subsequent works in the field of the power consumption optimization, such as [12], [13], [14], have proposed to use the power control technique [12] when routing packets. That is the transmitter does not use a fixed full-power when transmitting data packets to a given receiver, but an adaptable power according to the distance separating the two nodes. Using the watchdog with this power-efficient technique could cause false detections, as illustrated in the following example:
Assume that B uses controlled transmission powers, and the required power from B to C is less than the one needed to reach A from B, thereby, the packets sent from B to C will not be received at A. The node A may accuse wrongly B as misbehaving even though this latter forwards packets to C. Hence, the watchdog fails when the power control technique is employed. The main purpose of our proposal is to overcome this problem.

Moreover, the watchdog cannot detect the misbehavior in many cases, such as:

1. Partial dropping: node B can circumvent the watchdog by dropping packets at a lower rate than the watchdog's configured minimum misbehavior threshold
2. Collisions: after a collision at node C, B could skip retransmitting the packet without being detected by A
3. False misbehavior: A node may falsely report other innocent nodes in its neighborhood as misbehaving, to avoid getting packets to forward to them
4. Insufficient transmission power: B can control its transmission power to circumvent the watchdog. If B is closer to A than C, then B could attempt to save its energy by adjusting its transmission power, and making it strong enough to be overheard by the previous node (A), but too weak to be received by the true recipient (C). Note that performing this way is power efficient for B.
5. Cooperated misbehavior: B and C could collude to cause mischief. In this case, B forwards a packet to C but does not report to A when C drops the packet. C does the same thing when it is B's predecessor.

IV. New approach

A. Solution overview

In this paper, we define a new approach to mitigate some watchdog's problems, especially that related to the transmission power control technique usage. In our approach, like in the watchdog, each node monitors the forwarding of each packet it sends, and a source routing protocol is assumed to be used. As in the previous section, we suppose without loss of generality that A sends packets to B and monitors the forwarding to C.

We define a new kind of feedback that we call *two hops ACK*, it is an ACK (acknowledgment) that travels two hops (two wireless links) and passes through an

intermediary. Node C acknowledges packets sent from A, and sends this latter,via B, a two hops ACK. The node B could, however, escape from forwarding the packet without being detected, this by sending A a *falsified* two hops 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 sending a data packet. To avoid 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 as well as A's address. When C receives the packet, it gets the number back, decrypts it using its secret key (SK), encrypts it using A's PK, puts it in a two hops ACK, and sends this packet back to A via B. A decrypts the random number, and checks if the number within the packet matches with the one it has generated, to validate the B's forwarding regarding the appropriate packet. However, if B does not forward the packet, A will not receive the two hops ACK, then it will be able to detect this misbehavior after a time out. To ensure that nodes share their PK with each other, a security association between each per of nodes is needed. This requires a key distribution mechanism, which is out of the scope of our purpose. Anyway, a mechanism like [15] or [16] can be used.

Another problem whould take place when node C misbehave. If C does neither forward the packet nor send the two hops ACK back to A, B could be supposed by A to not forward the packet even it actually does. To overcome this problem, we suggest that the sending of the two hops ACKs would be provided implicitly upon the reception of the packet at the *MAC layer*. The lower layers (the MAC and physical layers) are supposed to be robust and tamper resistant. Therefore, node C could not escape from sending the two hops ACK back to A upon the reception of the packet. However, the upper layers including the network layer may be tampered by a selfish or a malicious, and *falsified* packets could be sent.
Our solution is composed of two parts, the first one is located at the network layer, and can be viewed as a sub layer at the bottom of it, whereas the second one is located in the MAC layer, and is a sub layer at the top of this latter. Figure 1 illustrates this framework.

Getting rid of the promiscuous mode makes our solution independent of transmission powers, and resolves the watchdog false detection problem related to the employment of the power-control technique. Moreover, our solution resolves the problem 2 of the watchdog (section 2). 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 watchdog where the validation is only related to B's first transmission. Our solution also solves the problem 4 of the watchdog. Remember that when A is closer to B than C, then B could save its energy and makes the transmission power strong enough to be overheard by A

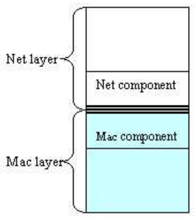

Fig. 1. Solution's framework

but less than the required one to reach C. This problem is eliminated in our solution, since B's forwarding validation at A is not just related to B's transmission, but to C's reception. Furthermore, the two-hop ACK we use allows to detect the *appropriate* selfish node, unlike the end-to-end ACKs [11].

B. The protocol

Each node, except the destination, is monitored by its predecessor. To monitor its successor, each node i adds to each packet it receives from the routing protocol a random number it generates, encrypted with the PK of the successor's successor, along with i's address. The monitor (node i) maintains the generated random number as well as the monitored node (i's successor) address in an entry within the Wait2HopsACK buffer. When a packet is received from another node X, i's MAC component automatically generates and sends X back a two hops ACK, after encrypting and decrypting again the random number as described previously. The Network layer component removes the appropriate entry upon the reception of the two hops ACK, and as a timeout is associated to each entry, the lack of the two hops ACK after this timeout results in the increasing of the rating regarding the appropriate forwarder node. A node is considered as selfish if its rating exceeds a given threshold. Like the watchdog, we use this rating and we do not accuse directly the forwarder, since lost of packets may be caused by channel conditions or nodes mobility, and is not inevitably due to an intentional misbehavior.
Algorithms 1 and 2 describe respectively the network component and the MAC component.

V. Modeling and correctness proof

In this section, we first model our previous protocol using petri nets. After that, we apply some linear algebra results on our model, to proof the protocol's correctness. Due to space limitation the proof will be summarized, more details can be found in [17]. We also omit all the basic definitions related to petri nets. Note that the reader of this section is supposed to have basic knowledge about this mathematical modeling method.

A. Notations and definitions

In the rest of the paper we will use the following notations:

Algorithm 1 Network module of solution 1

```
When receive a packet D from the routing protocol to send 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
    R = a generated random number
    Y = X's successor in the source route
    append $(R_{P_Y}, i)$ to D's header
    add(R,X) to the buffer Wait2HopsACK
  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 hops 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
  end if
When a timeout out of a Wait2HopsACK entry (R,X) is exceeded
  increment the rating regarding node X
  if (X's rating > threshold) then
    consider X as a misbehavior
  end if
```

Algorithm 2 MAC layer located component of solution 1

```
when receive a packet D sent by X from the MAC protocol
  if $(X \neq D'ssource)$ then
    Y = X's predecessor in the source route
    Get the random number R generated by y
    $R' = R^{S_I}$
    $R'' = R'_{P_Y}$
    construct a two hops ACK packet TwoHopsACK
    TwoHopsACK.Rand= $R''$
    TwoHopsACK.sender=I
    TwoHopsACK.dest=Y
    send two hops ACK to X
  end if
  pass the packet up to the network component
when receive a two hops ACK packet TwoHopsACK
  if (TwoHopsACK.dest $\neq$ i) then
    TwoHopsACK.sender=i
    forward TwoHopsACK to TwoHopsACK.dest
  else
    pass the packet up to the network layer component
  end if
```

- $< P, T, I, O, M_0 >$: 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
- $|x|_{sup}$: the upper integer part of x defined by: $\forall x \in \mathbb{R}, [x]_{sup} = m \in \mathbb{N} \diagup m \geq x$ and $m - 1 < x$.
- $||S||$: the cardinal 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 $t \in T^*$
- $R(< Net, M >)$: the set of marking reachable from the marking M in the petri net $Net < P, T, I, O >$, formally speaking:

  $R(< Net, M >) = \{M_R \in \mathbb{N}^{||P||} \diagup \exists (t_1, \cdots, t_n) \in T^n, M(t_1 \cdots t_n > M_R\}$

In the following, we define some concepts we will use later

Definition 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 $M(p) < I(p, T_i)$ [18].

Definition 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 $\forall M \in R(< Net, M_0 >)$, $M_h \in R(< Net, M >)$ [18].

Definition 3: A norm for a marking M_a is an application v from the markings set to $\mathbb{N}$, such that $\forall M \in R(< Net, M_0 >)$, v fulfils the following conditions:
i) $v(M) = 0 \Leftrightarrow M = M_a$
ii) $v(M) \neq 0 \Rightarrow \exists M' \in R(< Net, M >)$ such that $v(M') < v(M)$ [18].

B. The model

As we have seen previously, in our 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 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, if B drops m out of these n packets then A validates **exactly** n-m forwarding, thereby, it detects the m packets dropping. In this proof, we assume that channels are reliable, that is all packets sent will be correctly received at the recipient, and the sole cause of packets dropping is nodes misbehavior. We model our protocol by the following petri net: $Net_0 =< P, T, I, O, M_0 >$
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 hops ACK 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 hops ACK packets sent by C, not

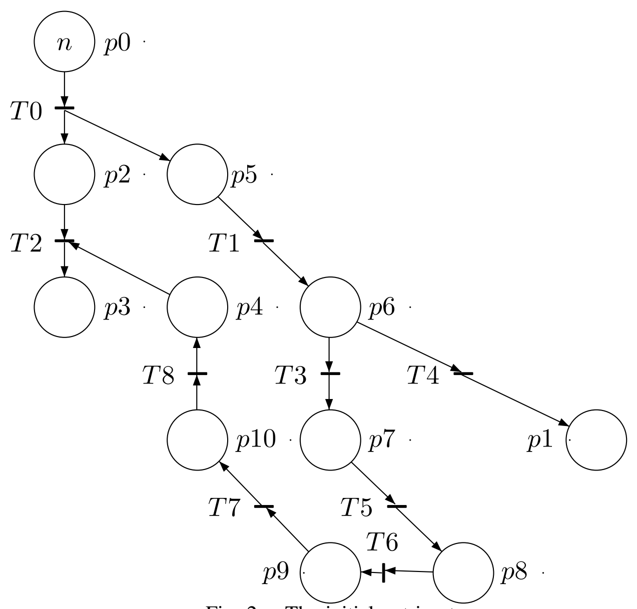

Fig. 2. The initial petri net

already received by B
P_{10}: buffer of two hops ACK 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 remove 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 hops ACK packet
T_7: B receives a two hops ACK packet
T_8: B forwards a two hops ACK packet

M_0 is represented by the following vector: $\begin{bmatrix} n & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^t$.

Figure 2 illustrates this petri net. I and O could be easily deduced from this graph [17].

C. Proof overview

Theorem 1: Let n be the number of packets sent by A, and m the number of packets dropped by B, then 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 hops ACK 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. Hence, validating exactly n-m packets is equivalent to detect m B's dropping. Thereby, this theorem show the protocol correctness. We bring the problem to our petri net model, and we propose the following lemma:

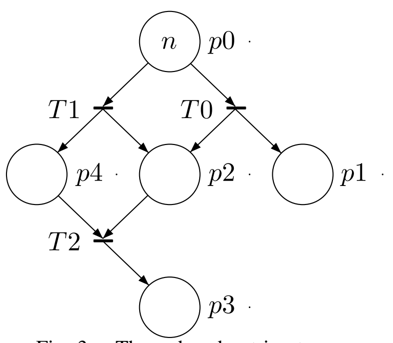

Fig. 3. The reduced petri net

Lemma 1: $\forall n \in \mathbb{N}$, $M_f = \begin{bmatrix} 0 & m & m & n-m & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^t, m \leq n$, 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 (p_1's tokens) is m, the number of packets whose forwarding is validated by A (p_3's tokens) 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.

1) Net reduction: We reduce the initial petri net using the *place substitution* [18]. Places $p_5 ... p_{10}$ are *substitutable*, the result of their substitutions is the net $Net_r < P_r, T_r, I_r, O_r, M_{r_0} >$ illustrated on figure 3. A detailed description of these substitutions is available in [17].

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 [18], the lemma 1 is reducible to the following

Lemma 2: $\forall n \in \mathbb{N}$, $M_f = \begin{bmatrix} 0 & m & m & n-m & 0 \end{bmatrix}^t, m \leq n$, 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 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 [18].

2) The norm: We propose the following application that we will use in the next lemma:

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

To prove lemma 2, we suggest the following one:

Lemma 3: V is a norm for M_f in Net_r

To prove this, we have to prove that V fulfills the two conditions of definition 3. Because of space limitation, we are unable to illustrate this verification here, the reader can refer to [17] for more details.

From this lemma we realize the correctness of lemma 2, lemma 1, and theorem 1.

VI. Conclusion and future work

In this paper, we have dealt with packets forwarding monitoring, which is at the core of any misbehavior detection system. As we have seen, the watch dog technique used by almost all the solutions currently proposed to detect selfish nodes on packets forwarding in MANETs, fails and may provide wrong detections when the power control technique is employed. To overcome this problem, we have proposed a new approach that gets rid of the passive monitoring in the promiscuous mode. Unlike the end to end ACKs, our approach allows to detect the misbehaving node. Moreover, it resolves the problems related to cases 2 and 4 of the watchdog (section3).

We have also modeled our solution by a petri net. First, the large net modeling our protocol has been reduced using substitutions, resulting in a reduced petri net. Then using this latter, we have proved that our solution allows the detection of exactly all the packets dropped by the forwarders. Nevertheless, in practice, packets dropping may be unintentional, i.e a packet may be dropped at a well behaving node because of channel conditions or nodes mobility. For this reason, we do not accuse a node as a selfish upon the detection of a packet dropping, but we indeed use a threshold as in the watchdog. As a perspective, we plane to complete the proposal by: giving a rigorous definition to this threshold, defining actions that have to be taken when a node is accused as a selfish, and defining a robust and a secure mechanism allowing nodes to exchange their knowledge regarding nodes that behave selfishly. We also plan to assess the performance of our protocol by simulation.

References

[1] 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*, 2000, pp. 255–265.

[2] L. Buttyan and J.-P. Hubaux, "Stimulating cooperation in self-organizing mobile ad hoc networks," *ACM/Kluwer Mobile Networks and Applications, Vol 8, N 5*, October 2003.

[3] D. Djenouri and N. Badache, "Selfishness an emergent threat on packet forwarding in mobile ad hoc networks," *LSI Technical report, LSI-TR0604, University of Scinece and technology houari boumediene, Algiers, Algeria*, April 2004.

[4] B. David and A. David, "Dynamic source routing in ad hoc wireless networks," *Mobile Computing, Chapter 5*, pp. 153–181, 1996.

[5] X. M. H. Yang and S. Lu, "Self-organized network layer security in mobile ad hoc networks," in *ACM MOBICOM Wireless Security Workshop (WiSe'02), Georgia, Atlanta, USA*, September 2002.

[6] C. Perkins and E. Royer, "Ad hoc on demand distance vector (AODV) algorithm," in *the 2nd IEEE Workshop on Mobile Computing Systems and Applications (WMCSA'99)*, 1999, pp. 90–100.

[7] P. Michiardi and R. Molva, "Core: A collaborative reputation mechanism to enforce node cooperation in mobile ad hoc networks," in *Communication and Multimedia Security 2002 Conference, Portoroz, Slovenia*, September 26-27 2002.

[8] S. Buchegger and J.-Y. L. Boudec, "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.

[9] L. Buttyan and J.-P. Hubaux, "Nuglets: a virtual currency to stimulate cooperation in self-organized mobile ad hoc networks," *Technical report No. DSC/2001/001, Swiss Federal Institution of Technology, Lausanne, Switzerland*, January 2001.

[10] V. Srinivasan, P. Nuggehalli, C. F.Chiasserini, and R. R.Rao, "Cooperation in wireless ad hoc networks," in *IEEE INFOCOM'03, San Francisco, California, USA*, April 2003.

[11] P. Papadimitratos and Z. J. Haas, "Secure data transmission in mobile ad hoc networks," in *ACM MOBICOM Wireless Security Workshop (WiSe'03), San Diego, California, USA*, September 2003.

[12] S. Doshi and T. Brown, "Minimum energy routing schemes for a wireless ad hoc network," in *IEEE INFOCOM 2002*, 2002.

[13] D. Djenouri and N. Badache, "Simulation performance evaluation of an energy efficient routing protocol for mobile ad hoc networks," in *IEEE/ACS International Conference on Pervasive Services (ICPS)*, Beirut, Lebanon, 19-23 July 2004.

[14] X. Du, "A simulation study of an energy efficient routing protocol for mobile ad hoc networks," in *37th IEEE Annual Simulation Symposium*, 18-22 April 2004.

[15] S. Yi and R. Kravetso, "Moca : Mobile certificate authority for wireless ad hoc networks," in *The second anunual PKI research workshop (PKI 03), Gaithersburg*, 2003.

[16] S. Capkun, L. Buttyan, and J.-P. Hubaux, "Self-organized public-key management for mobile ad hoc networks," *IEEE Transactions on Mobile Computing, Vol.2, No.1*, pp. 52–64, January 2003.

[17] D. Djenouri and N. Badach, "A petri net based corectness proof of a selfish nodes detection protocol for mobile ad hoc networks," Computer Science department, University of Scinece and technology houari boumediene, Algiers, Algeria, Tech. Rep. LSI-TRO804, June 2004.

[18] G.W.BRAMS, *Rseau de petri: Thorie et pratique*. Edition masson, 1983.