Proceedings of the Sixth IASTED International Conference
Communications, Internet, and Information Technology
July 2–4, 2007, Banff, Alberta, Canada
ISBN Hardcopy: 978-0-88986-673-7 / CD: 978-0-88986-674-4

USING A TWO-TIMER SCHEME TO DETECT SELFISH NODES IN MOBILE AD-HOC NETWORKS

Hongxun Liu, José G. Delgado-Frias, and Sirisha Medidi
School of Electrical Engineering and Computer Science
Washington State University
Pullman, WA 99164-2752, USA
{hliu2, jdelgado, smedidi}@eecs.wsu.edu

ABSTRACT

The cooperation of wireless nodes in ad hoc networks is crucial to ensure the proper working of the whole network due to the absence of infrastructure. Misbehaving nodes can dramatically decrease the performance of ad hoc network. Detecting and isolating the misbehaving nodes can not only make sure the participating nodes forward and route packets correctly, but also discourage the individual nodes from gaining advantages by not providing services to other nodes. This paper proposes a novel hardware assisted scheme which can detect one kind of misbehavior: selfish nodes. The proposed scheme consists of two timers that help to detect 80% or more selfish nodes with a probability of nearly 90%. The detection results could be used by other nodes to disengage the selfish nodes from the network. Furthermore, the virtue of hardware based detection makes this scheme less prone to virus influence as in some software based schemes.

KEY WORDS

Network Management, Wireless networks, Mobile Ad Hoc Networks, Hardware Detection

1. Introduction

Mobile ad hoc networks (MANETs) are getting a great deal of attention due to their peer-to-peer nature. Due to the features of self-configuration and self-maintenance, MANETs could be used where there are short of communication infrastructures, for example, battlefields and disaster relief [1].

The most commonly use routing protocols for MANETs such as DSDV [2], DSR [3], and AODV [4] do not take into consideration security, assuming all the mobile nodes in the MANET will coordinate with each other. Such routing protocols along with the absence of infrastructure make MANETs prone to Byzantine faults such as packet dropping, packet misrouting, etc. The existence of misbehaving nodes is a significant problem for MANETs which wastes the scarce wireless resource and degrades network performance. Having 10% to 40% misbehaving nodes degrades the average throughput by 16% to 32%, while a specific node could experience much worse performance degradation than the average [5]. Specifically, packet dropping can cause huge degradation to TCP based traffic due to the slow start nature of TCP [6].

Encryption based techniques try to prevent misbehaving nodes from launching any attacks. However, the security history shows that getting a completely intrusion-free system is infeasible, if not impossible, no matter how carefully the prevention systems are built [7]. In this paper, a novel hardware assisted detection scheme is proposed and evaluated; this scheme can detect the misbehaving nodes and report such information to the upper layer and other nodes. The other nodes can isolate the misbehaving nodes, resulting in a MANET composed of only good-behaving nodes.

The rest of this paper is organized as follows. Related work is reviewed in the next section. Then, Section 3 describes the hardware assisted detection scheme. Next, Section 4 and Section 5 describe the simulation and performance evaluation, respectively. Finally, conclusion and further research are discussed in Section 6.

2. Related Work

Some techniques have been proposed to deal with malicious nodes in MANETs. The "watchdog" technique requires that each wireless node snoop other nodes on retransmission of packets [5]. Watchdog requires omni-directional antennas and cannot tell the difference between transmission error and malicious handling. So, watchdog could mistakenly indict some innocent nodes as misbehaving nodes. Furthermore, if a misbehaving neighbor node drops all the incoming packets, the watchdog has no means to tell if there is a misbehaving neighbor node. In [5], there is no data available for the detection accuracy purpose. Buchegger and LeBoudec [15] propose a similar technique which requires encryption based associations among wireless nodes.

Papadimitrados and Haas [8] propose the Secure Routing Protocol (SRP), which extends DSR by adding SRP header to the basic routing header. The DSR packets are protected through a hash chain. SRP assumes there is already a security association between source node and destination node, which is not easy to establish in MANET. Paul and Westhoff [9] claim their hash-chain based framework can doubtlessly accuse a misbehaving node. But the voting scheme can only guarantee the accusation by some probability. Also, if the misbehaving node chooses to drop the packets silently, there is no way

the neighbors can tell it is a misbehaving node or it doesn't even exist.

Stejano and Anderson [10] propose the "Resurrecting Duckling" model which builds security associations between master devices and slave devices. This technique simulates the phenomenon imprinting, where the device will recognize as its owner of the first device sending it a secret key [10]. Resurrecting Duckling focuses on the authentication among wireless nodes and can't detect the misbehaving nodes if the software of such nodes is compromised.

The "Unobtrusive Monitoring" technique to detect packet dropping is proposed in [11]. In this scheme, all wireless nodes collect the TCP timeout events. If there is no corresponding DSR route error message or ICMP destination unreachable message to some TCP timeout event, such TCP timeout is taken as being caused by some misbehaving node. The problem with that scheme is that it cannot pinpoint the specific node that is misbehaving. Another issue is the detection is conducted offline.

3. Hardware Assisted Detection

3.1 Overview

The hardware assisted detection scheme can be used to detect routing attacks and packet forwarding attacks [7]. In this scheme, the hardware monitors the upper and software layers of its own node. The hardware consists of two logical components. One component contains tamper-resistance mechanisms, protecting the hardware from hardware attacks and logical attacks [12]. With the help of tamper-resistance component, each wireless node could be easily identified. The other component is responsible for detecting misbehavior of the upper layer. The hardware detection unit is the foundation of defending MANET. When the software of the node is compromised or is mounting attacks, the hardware can detect the misbehavior of the software layer and report it to the network. A mobile node with the hardware detection mechanism is depicted in Figure 1.

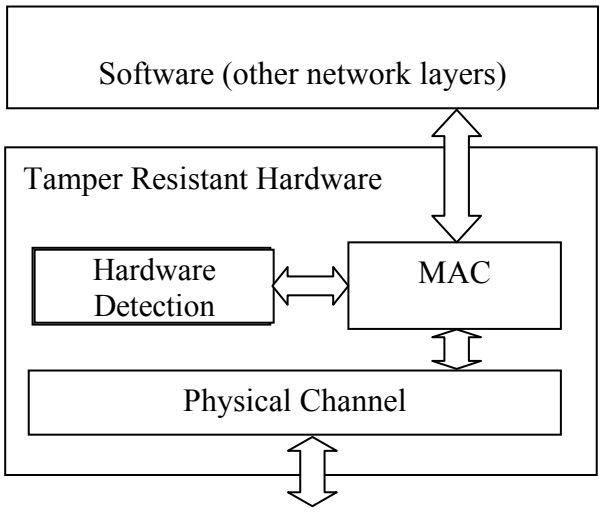

Figure 1. A mobile node with the hardware detection mechanism

This paper focuses on detecting selfish node which drops packets such that other nodes can never use it as an intermediate node. The reason for selfish node to drop packets is that the selfish node can save battery power, CPU cycles or wireless bandwidth. In this scheme, the hardware can detect the selfish nodes and report the results to other nodes or its own upper layer.

The proposed hardware detection scheme is based on the principle of packet flow conservation, i.e., the number of incoming packets, excluding the ones destined to the current node, should be the same as the number of outgoing packets, excluding the ones generated by itself [13]. Since both the incoming packets and outgoing packets need to go through the hardware, it is easy for the hardware to detect packet mishandling.

3.2 Two-timer Scheme

In the two-timer scheme, packets are classified into data packets and control packets, the latter part focusing on the route request packets. A data packet is sent through unicast, while a route request packet is sent out through broadcast. The route reply packets are not treated separately because it is forwarded through unicast. To the hardware detection, the route reply is treated the same as normal data packet.

In order to track the incoming packets and outgoing packets, two timers and a Drop Counter are used in each node. The Drop Counter is updated at two places: when a packet enters the node and when a packet leaves the node. For each incoming non-route-request packet, Drop Counter is increased by 1; for each outgoing non-route-request packet, Drop Counter is decreased by 1. If the value of Drop Counter exceeds a threshold value, the current wireless node is marked as misbehaving node and a warning message is sent to neighboring nodes

There are two timers in this scheme: Detection Timer (DeT) and Reward Timer (RwT). The Det and RwT are only used for route request packets. The DeT is used to detect if the wireless node forwards a received route request packet during the detection period. A received route request packet will first arrive at the hardware layer which triggers the DeT. Then the packet will be passed to the software layer of the same node. The software layer of a good-behaving node will process the route request packet. If the route request is original, it will forward the route request packet. If the route request is a duplicated request, the software will drop the route request. If the node never forwards the route request packet and the DeT expires, the value of Drop Counter is increased by 1. If the node's DeT keeps incrementing the counter, the Drop Counter of a misbehaving node will reach a predetermined threshold, thus triggering the warning message to be sent.

The other timer, Reward Timer (RwT), is used to reward the good-behaving node during the route discovery process of DSR. During the process of route

discovery, the route request packet is usually sent through broadcast [3], which will cause the same route request packet to be received several times by a single node. As shown in Figure 2, when node A receives a route request packet and broadcasts that packet, its neighbor B will receive and broadcast the packet. Due to the nature of broadcast, Node A will receive the same route request packet again from node B. If node A has a few neighbors within its transmission range, it is likely that A will receive a few duplicate route requests. To compensate such an irregularity, after a good-behaving node forwards a route request, the node will be rewarded a grace period by means of Reward Timer (RwT), during which the node could "drop" the duplicate route request without being penalized. As shown in Figure 3, a wireless node receives a route request packet A (RRP_A) which starts the DeT. During the period of DeT, the RRP_A is forwarded, which starts the RwT. Then the three duplicate RRP_A received during RwT will not be counted as new packets and dropping the three duplicate RRP_A will not increase the drop counter.

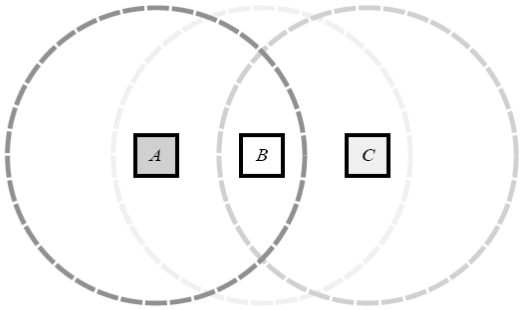

Figure 2. A simple wireless Ad Hoc Network

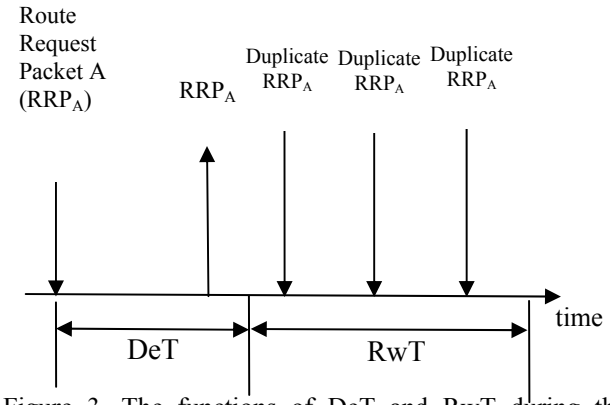

Figure 3. The functions of DeT and RwT during the processing of route request packet

RwT is only initiated when a valid route request packet is forwarded during the period of DeT. The detailed description about the processing of incoming packets and outgoing packets is showed in Algorithm 1 and 2.

Algorithm 1: incoming packet processing

```
if (incoming packet neither from nor to
        this node itself)
then
  if (packet is route request)
  then
    switch (detection status )
      case 0: {both timers are idle}
        start detection timer;
        detect_timer_.sched(detect_timer_len_ );
        drop counter increase by 1;
        detection status change to 1;
        break;
      case 1: {detect timer is pending}
        break;
      case 2: {both timers pending }
        break;
      default:
        break;
    end switch
  else  { not a route request }
    drop counter increase by 1;
  end if
end if
if ( drop counter larger than threshold )
then
  mark this node as misbehaving;
end if
```

Algorithm 2: outgoing packet processing

```
if (outgoing packet neither from nor to
        this node itself)
then
  if (outgoing packet is route request)
  then
    switch (current detection status)
      case 0: {both timers are idle}
        if (drop counter larger than 0 )
        then
          drop counter decrease by 1;
        endif
        break;
      case 1: {detect pending and reward idle}
        if (drop counter larger than 0 )
        then
          drop counter decrease by 1;
        endif
        detection status change to 2;
        break;
      case 2: {detect or reward timer pending}
        if (drop counter larger than 0 )
        then
          drop counter decrease by 1;
        endif
        break;
      default:
        break;
    end switch
  else { not a route request }
    if (drop counter larger than 0 )
    then
      drop counter decrease by 1;
    endif
  end if
end if
```

---

4. Simulations

This section describes the design and running of simulations for the two-timer scheme. The simulation is implemented on ns2, which is one of the most commonly

used network simulator. As mentioned above, the simulation is based on DSR, which could be easily extended to simulate other wireless routing protocols, such as DSDV, AODV, etc.

4.1 Simulation Scenario

The simulations use Random Waypoint Model [14], in which a node moves from one location to another location by randomly selecting some direction and speed. According to [14], Random Waypoint Model creates realistic mobility pattern for the way people might move in and is used in many prominent simulation studies on ad hoc network protocols.

4.2 Misbehaving Nodes

Ns2 simulator has been modified to enable some nodes to be configured as misbehaving. The misbehavior here is defined as dropping all the incoming packets except the packets for itself. The misbehaving node is selected randomly for each simulation. The percentage of misbehaving nodes varies from 10% to 40% of the total nodes.

4.3 Detection Effectiveness and False Positive

Detection Effectiveness measures how well the two-timer scheme performs in identifying the misbehaving nodes. The detection effectiveness is computed as follows.

$$DetectionEffectiveness = \frac{Detected_nodes}{Total_misbehaving_nodes} \times 100$$

In the formula above, detected_nodes are the misbehaving nodes that have been identified. For example, if the scheme detects all of the misbehaving nodes in the network, the detection effectiveness is 100%. On the other hand, if it cannot detect any misbehaving node, the detection effectiveness is 0%.

Some good-behaving nodes may be misidentified as misbehaving nodes; this is called *False Positive*. It is measured as the number of good but detected misbehaving nodes divided by the total number of detected misbehaving nodes.

$$FalsePositive = \frac{missclassified_missbehaving_nodes}{Detected_nodes} \times 100$$

The detection object would be to have 0% false positive and 100% detection effectiveness. But this in turn is extremely difficult to obtain. Our goal in this study is to keep false positive at 0% and try to get the detection effectiveness as high as possible. Having a number of good nodes being misclassified has a negative effect on the overall performance of the network. On the other hand, some nodes that have the potential of misbehaving may not be involved in any communication path.

4.4 Two-Timer Selection

The selection of the two timers is crucial to the success of the detection. The purpose of the Detection Timer (DeT) is to check if the node would forward a packet within that time period. If the value of DeT is too small, even the good-behaving node could not forward the received packet in a timely manner. On the other hand, if DeT is too large, the efficiency of the detection process will be decreased. For Reward Timer (RwT), its purpose is to give the good-behaving node more time to deal with the duplicate packets. If it is too small, the good-behaving node could be deemed as dropping some valid packets while it is really dropping the duplicate route requests. On the contrary, if RwT is too large, the granularity of the detection will be affected adversely.

4.5 Simulations

Two sets of simulations have been conducted to determine the length of the timers –simulation set 1 (SS1) and to evaluate the performance – simulation set 2 (SS2). SS1 is to find good fittings for DeT and RwT where we experiment with a range of values for these timers. After finding the good fittings, SS2 is to verify the effectiveness of the two timers selected. To eliminate the effect of fluctuation, each set of simulation is run a number of times to get the average result. Within the set of simulation, the selection of the misbehaving nodes is done by means of a random process, i.e. every simulation has a different set of misbehaving nodes. In SS1, each single simulation is run 10 times; while in SS2, each single simulation is run 100 times to verify the effectiveness of the detection scheme. For both simulations, the percentage of misbehaving nodes ranges from 10% to 40%, with an increment of 10%.

5. Performance Evaluation

In this section we evaluate the performance of the proposed two timer scheme. We, first, discuss the selection of the two timer time ranges. Once the timers are selected, we present an extensive study of the performance of the scheme under different scenarios.

5.1 Selection of Two Timers

The purpose of these simulations (SS1) is to find the best lengths for the two timers –DeT and RwT. The simulation is conducted for a large range of timers. We have restricted the selection of the timers' length to the time-span of the timers that results in 0% false positive detection. As for detection effectiveness, it is good to have high detection as long as keeping the false positive as 0%. DeT ranges from 10ms to 15ms with increments of 1ms. RwT ranges from 60ms to 75ms with increments of 5ms. DeT and RwT ranges were selected after a large number of simulations; these ranges yielded the best results. Figure 4 shows the results of simulations that have 0% false positive and at least 80% detection effectiveness. Figure 5 shows the results with 0% false positive and at least 90% detection effectiveness. For each timer pair (DeT and RwT), four kinds of simulation are

conducted for different percentages of misbehaving nodes, 10%, 20%, 30% and 40%. Our goal is to find the DeT-RwT pair which could work well for all those four cases.

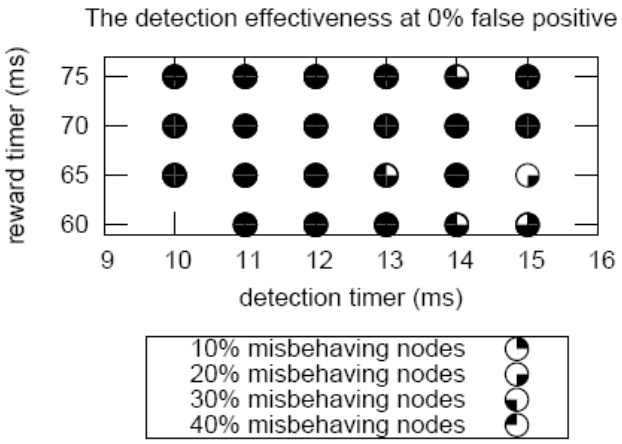

Figure 4. Simulation results with 0% false positive and at least 80% detection effectiveness

Figure 4 shows a good number of DeT-RwT pairs that work well for 0% false positive and at least 80% detection effectiveness. These results provide a good number of DeT-RwT pairs to choose from. On the other hand if detection effectiveness is set to a higher level (90+%), from Figure 5 we observed that only two pairs, (12, 70) and (13, 75), work well for all percentages of misbehaving nodes. The next simulation will confirm the effectiveness of the timers selected from SS1.

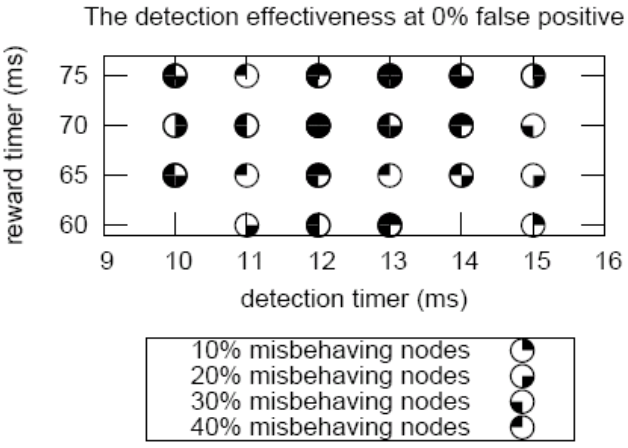

Figure 5. Simulation results with 0% false positive and at least 90% detection effectiveness

5.2 Performance

The timer pair used in this simulation is from the simulation results above. In this simulation, 100 simulations are conducted using the timer pair (13ms, 75ms). For SS2, we have kept the false positive as 0%. The results of the simulation are shown in Table 1 and in a graphical from in Figure 6. It should be pointed out that all these simulations have 0% false positives.

In Figure 6, there are four columns. Each column corresponds to a percentage of misbehaving nodes in the simulation. As mentioned earlier, for each percentage of misbehaving nodes 100 simulations are run. As shown in Figure 6, the scheme can detect 80% or above misbehaving nodes in all the percentages at a probability of almost 90%. For all the percentages, they have perfect detections at a probability of at least 55%. An interesting phenomenon is that the probability of perfect detection decreases as the percentage of misbehaving nodes increases. One reason is that more misbehaving nodes lead to more separated networks. In such cases, there is no enough traffic to pass the misbehaving nodes while the two-timer scheme needs a relatively connected network. Another reason is that as the percentage of misbehaving nodes increases the number of packets resent increases, leading to longer response time for the good-behaving nodes.

Table 1. Simulation Results

Detected Misbehaving Nodes (%)	Percentage of Misbehaving			
	10%	20%	30%	40%
0-60	8	6	4	6
60-70	5	0	5	3
70-80	0	2	5	3
80-90	12	7	14	12
90-100	0	15	16	21
100	75	70	56	55

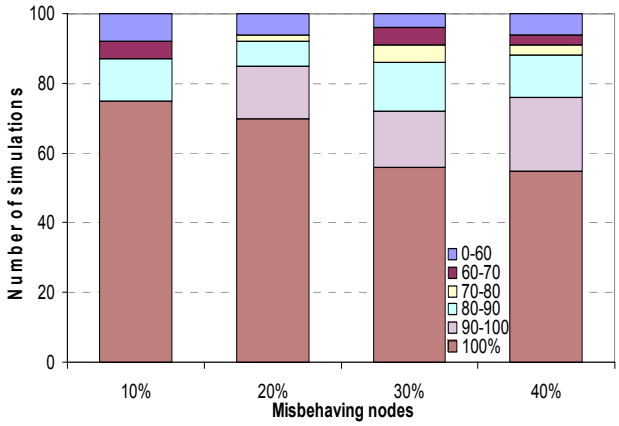

Figure 6. Simulation results with 0% false positive

The "Unobtrusive Monitoring" technique uses the same evaluation metrics of detection effectiveness and false positive [11]. Unobtrusive Monitoring tries to detect misbehaviors in MANET through TCP timeout event and DSR route error message. If there is no corresponding DSR route error message to a TCP timeout event, the TCP timeout event is caused by some misbehaving node along that TCP connection. In comparison to the two-

timer scheme proposed in this paper, the unobtrusive monitoring only has detection effectiveness of 60%-80% with the false positive at around 40% [11]. On the other hand, the unobtrusive monitoring can only detect the misbehaving routes in stead of detecting the misbehaving nodes. Thus the unobtrusive monitoring cannot isolate the misbehaving nodes. Furthermore, unobtrusive monitoring can only conduct the detection work offline. The misbehaving nodes could cause the damage to the network without being identified during the running of the network. On the contrary, the two-timer technique proposed in this paper can detect the misbehaving nodes in a real-time manner with high detection effectiveness and zero false positive.

6. Conclusion

Mobile ad hoc networks are more vulnerable to misbehaving activities than the wired networks, which makes securing the mobile ad hoc networks very promising and enormously important. This paper presents a hardware based two-timer scheme to detect the misbehaving nodes. The features of the proposed scheme are:

- *High detection of misbehaving nodes.* The proposed scheme can detect 80% or above misbehaving nodes with a probability of almost 90%.
- *Low false positive.* The two timers in the simulations are set to achieve a 0% misclassification of good-behaving nodes as misbehaving nodes
- *Minor changes to software layer.* The proposed scheme requires very little change to the present software layer and can be easily implemented at the hardware layer due to the simple nature of the scheme.
- *Simple to implement in hardware.* There are only two timers and a counter needed. Thus this scheme can be implemented at very low cost.

The simulation results show that the two-timer scheme can detect and misbehaving nodes accurately in terms of detection effectiveness and false positive.

References

[1] R. Rao and G. Kesidis, "Detecting malicious packet dropping using statistically regular traffic patterns in multihop wireless networks that are not bandwidth limited," *Proceedings of IEEE GLOBECOM*, pp 2957–2961, 2003.

[2] C. Perkins and P. Bhagwat, "Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers," *ACM SIGCOMM Computer Communication Review,* pp 234 – 244, October 1994.

[3] D. B. Johnson and D. A. Maltz, "Dynamic source routing in ad hoc wireless networks," *Mobile Computing*, vol. 353. Kluwer Academic Publishers, 1996.

[4] C. Perkins and E Royer, "Ad Hoc On-Demand Distance Vector Routing," *Second IEEE Workshop on Mobile Computer Systems and Applications,* pp. 90-100, February 1999.

[5] S. Marti, T. J. Guili, K. Lai, and M. Baker, "Mitigating routing misbehavior in mobile ad hoc networks," *Proceedings of ACM SIGCOMM*, pp 255–265, 2001.

[6] C. Barakat, E. Altman, and W. Dabbous, "TCP Performance in a heterogeneous Network: A survey," *IEEE Communication*, vol. 38, pp 40–46, January 2000.

[7] H. Yang, H. Luo, F. Ye, S. Lu, and L. Zhang, "Security In Mobile Ad Hoc Networks: Challenges And Solutions," *IEEE Wireless Communications*, pp 38-47, February 2004.

[8] P. Papadimitrados and Z. Haas, "Secure Routing for Mobile Ad hoc Networks," *SCS Communication Networks and Distributed Systems Modeling and Simulation Conference (CNDS 2002),* January 2002.

[9] K. Paul, D. Westhoff, "Context Aware Detection of Selfish Node in DSR based Ad-hoc Network," *IEEE GLOBECOM 2002,* November 2002.

[10] F. Stejano and R. Anderson, "The resurrecting duckling: Security Issues for Ad-hoc Wireless Networks," *Proceedings of the International Workshop on Security Protocols*, pp 172–194, April 1999.

[11] S. Medidi, M. Medidi, and S. Gavini, "Detecting Packet-dropping faults in Mobile ad-hoc networks," *Thirty-Seventh Asilomar Conference on Signals, Systems and Computers,* pp 1708 – 1712, November 2003.

[12] S. Ravi, Anand Raghunathan and Srimat Chakradhar, "Tamper Resistance Mechanisms for Secure Embedded Systems," *Proceedings of the 17th International Conference on VLSI Design*, 2004.

[13] X. Zhang, S.Wu, Z. Fu, and T. L.Wu, "Malicious Packet Dropping: How It might Impact the TCP Performance and How We can Detect It," Proceedings International Conference on Network Protocols, pp 263–272, 2000.

[14] T. Camp, J. Boleng, and V. Davies, "A survey of mobility models for ah hoc network research," *Wireless Communication and Mobile Computing*, pp 483 – 502, 2002.

[15] S. Buchegger and J. Y. Le Boudec, "Nodes bearing grudges: Towards routing security, fairness, and robustness in mobile ad hoc networks," *Proceedings of the Parallel, Distributed and Network-Based Processing*, pp 403–410, January 2002.