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 CACHE 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

This paper presents a hardware based cache scheme to detect selfish nodes in mobile ad hoc network. In this scheme, the hardware monitors the activities of the upper-layer software and reports the misbehavior about the software to other mobile nodes in the network. The hardware cache stores the identity information of recently received packets. The detection mechanism uses the cache to tell the difference between original packet and duplicate packet. The simulation results show that the cache scheme can detect nearly 100% selfish nodes with nearly 0% false positive in the simple dropping scenario. In the selective dropping scenario, the detection has nearly 0% false positive and reasonable detection rate. The detection result could be used by other nodes to protect the network.

KEY WORDS

Network Management, Wireless networks, hardware detection, Mobile ad hoc networks, and packet dropping

1. Introduction

Mobile ad hoc network (MANET) is an ongoing research topic. Unlike traditional network, MANET does not need infrastructure to work properly. 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]. As the cost of the wireless devices drops, more and more peer-to-peer usage of MANETs are appearing; these networks are replacing fixed connections for some casual usage.

Some popular used routing protocols for MANETs such as DSDV [2], DSR [3], and AODV [4] do not take into consideration security issues, 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 active attacks and passive attacks. For example, the selfish nodes may drop packets silently to save energy. Another reason for dropping packets may be that the mobile node's software has been compromised by some virus. 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 worse performance degradation than the average [5]. Specifically, packet dropping can cause huge degradation to TCP traffic due to the slow start nature of TCP [6].

Some techniques were proposed to protect MANET. For example, proactive techniques usually use encryption to protect MANET by preventing attacks from happening. 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]. Another issue about encryption based techniques is that it requires high computation ability on the mobile node while usually the mobile node has only limited ability. Some reactive techniques detect the misbehaving nodes and isolate the misbehaving nodes from the network [5].

In this paper a novel hardware assisted detection scheme is proposed and evaluated. In this scheme, the hardware can detect the misbehavior conducted by the selfish nodes. Selfish nodes are mobile nodes optimizing their own gain and neglecting for the welfare of other nodes. In this paper, the selfish node either drops all the packets not related to it or drops the data packets only. Upon detecting the misbehavior, the hardware will report the misbehaving node (itself) to other nodes. The other nodes will use the information received to protect the network. For example, other nodes could isolate the misbehaving nodes from the network. In this scheme, there is a separation between software and hardware inside a single mobile node. The software could be misbehaving, but the hardware is tamper resistant [8] and is the cornerstone of building trust relationship among mobile nodes.

The rest of the paper is organized as follows. Related work is reviewed in section 2. Then, section 3 describes the hardware assisted cache scheme. Next, section 4 and section 5 describe the simulation and performance evaluation, respectively. Finally, conclusion is reached in Section 6.

2. Related Work

Some techniques have been proposed to deal with misbehaving nodes in MANETs. The "watchdog" technique requires that each wireless node snoop other nodes on retransmission of packets [5]. The watchdog

raises a warning message if it finds out the other node didn't retransmit the packet correctly. Watchdog requires omni-directional antennas and cannot tell the difference between transmission error and misbehavior. So, watchdog could mistakenly indict some innocent nodes as misbehaving nodes. In [5], there is no data available for the detection accuracy purpose. Buchegger and LeBoudec [9] propose a similar technique which requires encryption based associations among wireless nodes. The encryption work requires the wireless nodes having good calculation ability which is not conveniently available to wireless nodes due to battery usage. This technique can not prevent mobile node making false indictments. Papadimitrados and Haas [10] 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 difficult to establish in MANET. Paul and Westhoff [11] 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 [12] 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 [12]. 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 [13]. 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. One issue with that scheme is that it can only tell that some wireless node along the TCP connection is behaving. It cannot pinpoint the specific node that is misbehaving. Another problem is the detection is conducted offline.

3. Hardware Assisted Detection

3.1 Overview

Nodes in MANET are independent. It can not be guaranteed that every node in MANET will be good-behaving. It is very difficult to build a reliable trust relationship among mobile nodes without predefined security association. In this case, tamper-resistance hardware can act as the foundation building reliable trust relationship among mobile nodes in MANET. Tamper resistant hardware is very popular in the smart card field. Some quite successful techniques are applied to build tamper resistant hardware [8]. When the software of a mobile node is tampered with or without the knowledge of user, the hardware will detect the misbehavior and send warning message to other nodes in MANET. In this paper, misbehaving and selfish are used interchangeably.

3.2 Packet Dropping

This paper focuses on the detection of misbehaving node dropping packet. There are two kinds of packet dropping conducted by the misbehaving nodes, simple dropping and selective dropping. In simple dropping, the misbehaving nodes will drop all the packets not to or from them; while in selective dropping, the misbehaving nodes will only drop data packets not to or from them while forwarding the control packets, such as route request, route reply, etc. Because in selective dropping the misbehaving node forwards some of the packets, it is more difficult to detect selective dropping than detect simple dropping. By dropping the data packets only, the misbehaving node can ask other nodes to forward packets for it without forwarding data packets for other nodes. In selective dropping, the misbehaving node could only drop control packet and forward data packet. But such case could not happen. Because the misbehaving node never forwards control packets including route reply packet, it will not appear as a valid node in the paths collected by other nodes. Thus, it will not receive request to forward data packet from other nodes. Thus, selectively dropping control packet only is equal to simple dropping.

3.3 Basic Cache Scheme

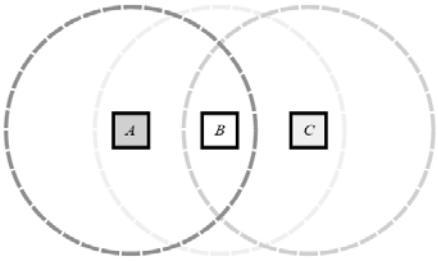

Figure 1. A simple wireless Ad Hoc Network

In hardware assisted detection scheme, the hardware is responsible to detect the misbehavior of the software and report such misbehavior to other nodes. In the cache based detection scheme, there is a cache unit as well as a few counters. The cache stores the identity information of the recently received packets and is used to differentiate original packets from duplicate packets received by wireless node. A mobile node could receive the same route request multiple times due to the broadcast effect during the route discovery process. As shown in Figure 1, when node A receives a route request packet and broadcasts that packet, its neighbor B will receive and

broadcast the route request 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. The cache can help the detection hardware recognize the original route request from the duplicate route requests.

There are four counters used in the cache based detection scheme: TC (Total Counter), DC (Drop Counter), TDC (Total Data Counter) and DDC (Data Drop Counter). The first two counters are used to detect simple dropping while TDC and DDC are used to detect selective dropping. TC is used to record the total number of unique packets received, while DC is used to record how many unique packets are dropped by this node. TDC is used to record how many data packets are received by the node while DDC records the number of data packets dropped.

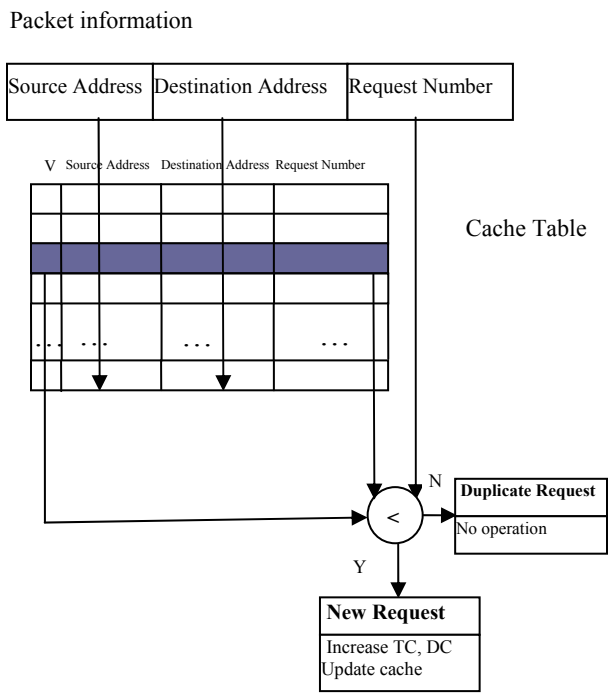

Figure 2. Processing of incoming Route Request Packet

The processing of incoming route request packet is depicted in Figure 2. As shown in Figure 2, each cache item has four fields: valid flag, source address, destination address and request number. When the detection hardware receives a route request, it will query the cache table if the same route request has been received before by using the source address and destination address as index. If there is a match in the cache table and the valid flag is set, the request number of the incoming packet is compared with the request number stored in the cache table. If the request number of the incoming route request is larger than the one in the cache table, the route request is original. Other wise, the incoming route request is a duplicate request. If the incoming route request is original, the cache table is updated with the new route request and the counters TC and DC are increased by 1. When the cache table is full, there needs a replacement strategy to find a slot for the new item in the cache table. The replacement strategy applied is first-in-first-out, a simple and effective algorithm.

If the incoming packet is a data packet, there is no need to query the cache table. In this case, the counters TDC and DDC are simply increased by 1.

For each outgoing packet, the counters are processed according to the type of packet received. For a route request packet, DC is decreased by 1. For a data packet, DDC is decreased by 1.

At some point, if either the ratio of DC to TC, or the ratio of DDC to TDC exceeds some predetermined thresholds, the detection hardware will send out a warning message to other nodes or the upper layer of the node. To decrease the false positive, DDC and DC need to be larger than some base value before the node could be indicted as misbehaving. The other nodes receiving the warning message could lower the rating of the node indicted or even isolate the node.

3.4 Modified Cache Scheme

The basic cache scheme described in last section needs another timer to improve the detection performance. The timer is used to give additional penalty if a node doesn't forward a route request. The penalty timer (PeT) is started when an original route request is received. If the node doesn't forward the route request during the period of PeT, an extra penalty is added to DC. PeT is only started when an original route request is received. A duplicate route request will not initiate PeT. The cache unit inside the detection hardware can tell if a received route request is original or duplicate.

4. Simulations

In this section, we will describe the design and running of simulations for the cache scheme. The simulation is implemented using NS2, a widely used network simulator. The simulation uses DSR as routing protocol.

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. This mobility model is memory-less, since it has no knowledge of its past locations and speeds. And the new direction and speed are independent of the previous direction and speed.

4.2 Misbehaving Nodes

Ns2 simulator has been modified to enable some nodes to be configured as misbehaving. The misbehavior here is defined as either simple dropping or selective dropping.

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 cache scheme performs in identifying the misbehaving nodes. The detection effectiveness is computed as follows.

$$DetectionEffectiveness = \frac{Detected\_nodes}{Total\_misbehaving\_nodes} \times 100$$

Detected_nodes are the misbehaving nodes that have been indicted. 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, which leads to the metric of *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 perfect detection would 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 as low as possible and try to get the detection effectiveness as high as possible.

4.4 Cache

The cache unit inside the detection hardware can differentiate the original packets from the duplicate packets. It stores the identity information of the recently received packets. Because it needs not to store the whole packet but the identity information, the size of each cache item is small. Furthermore, the duplicate packet could only be route request packets due to the broadcast nature. The fields of cache item are shown in Figure 2.

The number of cache items is expected to affect the detection performance in terms of detection effectiveness and false positive. On one hand, if the number of cache items is too small, the cache will misidentify some duplicate packets as original packets due to the quick replacement of the old items. On the other hand, too many cache items waste resource without gaining significant performance improvements. The simulations will find the optimal number of cache items.

4.5 Simulations

Two sets of simulations are conducted to evaluate the detection performance. Simulation set 1 (SS1) is used to detect simple dropping. Simulation set 2 (SS2) is used to detect selective dropping. For both simulations, the percentage of misbehaving nodes ranges from 10% to 40%, with an increment of 10%. To eliminate the effect of fluctuation, each simulation is run 20 times with different sets of misbehaving nodes to get the average result. Within the set of simulation, the selection of the misbehaving nodes is done by the means of a random process.

5. Performance Evaluation

5.1 Simple Dropping Scenario

The set of simulations conducted in this section is to show the detection performance of cache scheme in the simple dropping scenario.

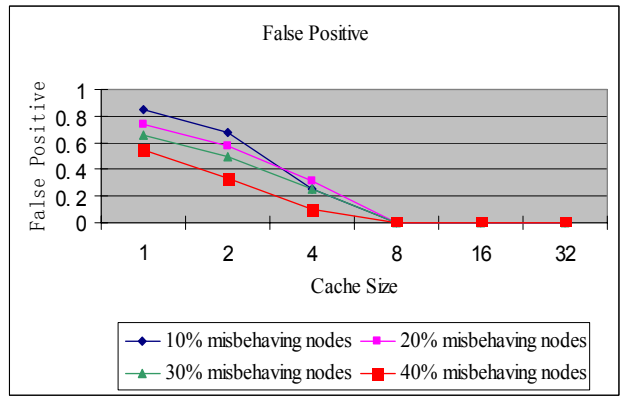

Figure 3. False positive in simple dropping scenario

Figure 3 shows the false positive using different cache sizes. The false positive decreases as the size of cache increases. This trend holds for all the four percentages of misbehaving nodes. When the size of cache increases to 8, the false positive is 0%. After this point, the increment of cache size will not gain further significant benefit. The false positive stays steadily at 0% when the cache size is at least 8.

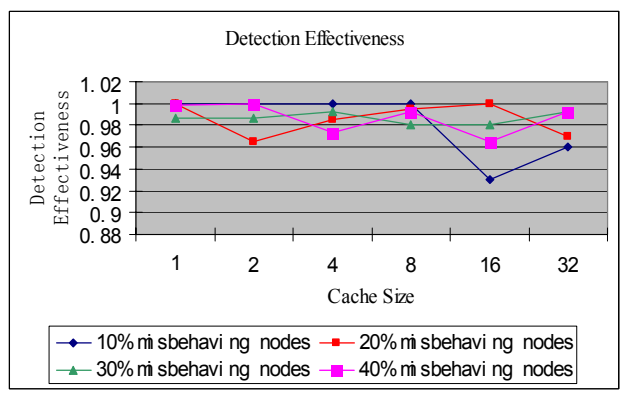

Figure 4. Detection effectiveness in simple dropping scenario

In Figure 4, the detection effectiveness does not change a lot when the cache size increases from 1 to 32. When the cache size is 1, the detection effectiveness is at least 98%, of course with a lot of good-having nodes misidentified as misbehaving. As the cache size increases, the detection effectiveness decreases just a little bit. When the cache size reaches 32, the detection effectiveness is

still at least 96%.

The goal is to keep the false positive as low as possible while keeping the detection effectiveness relatively high. According to the discussion above, as the size of cache increases there is no significant change in the detection effectiveness. So we can mainly focus on false positive. The false positive reduces to almost 0% when the cache size is equal to or larger than 8. Since there is no significant gain when the cache size is larger than 8, we could decide that 8 cache items are enough in the cache scheme proposed.

5.2 Selective Dropping Scenario

In selective dropping scenario, the misbehaving node will drop every incoming data packet unless that packet is from itself or to itself. At the same time, the node will forward control packets, such as route request, route reply, etc. The cache size is set as 8 which is confirmed in the simple dropping scenario above.

Table 1. False Positive and Detection Effectiveness for Different Percentages of Misbehaving Nodes

	10% misbehaving nodes	20% misbehaving nodes	30% misbehaving nodes	40% misbehaving nodes
False Positive	0%	0%	0%	0%
Detection Effectiveness	39%	45%	43%	41%

Table 1 shows the false positive and the detection effectiveness for different percentages of misbehaving nodes.

According to the results shown in Table 1, the cache scheme has almost 0% false positive. But the detection effectiveness is only around 40%. In comparison with the simple dropping scenario, the detection effectiveness in the selective dropping scenario seems not promising.

Actually, the low detection effectiveness is caused by the method used to calculate the detection effectiveness. In the formula, the numerator is the number of misbehaving nodes detected while the denominator is the total number of misbehaving nodes configured. The low detection effectiveness is caused by using the nodes configured as misbehaving as denominator. But a node configured as misbehaving might not have the chance to misbehave. If a node was configured as misbehaving but did nothing wrong, it should not be counted as misbehaving.

The essential reason that a node configured as misbehaving does not misbehave is that the node does not have the opportunity to drop packets. First, the misbehaving node might not on any data path. Thus, the misbehaving node has no chance to drop data packets. In another case, the misbehaving node is on some data path. But there are more than one misbehaving nodes on the same data path. The first misbehaving node on the data path will drop all the data packets, leaving the misbehaving nodes behind nothing to drop.

Another set of simulations are done to manifest the comments above. Since the low detection effectiveness is due to the phenomenon that the misbehaving node has no chance to drop data packets, this set of simulations use larger number of connections. Larger number of connections gives the misbehaving node more opportunity to be the first bad node on some data path, thus more opportunity to drop data packets.

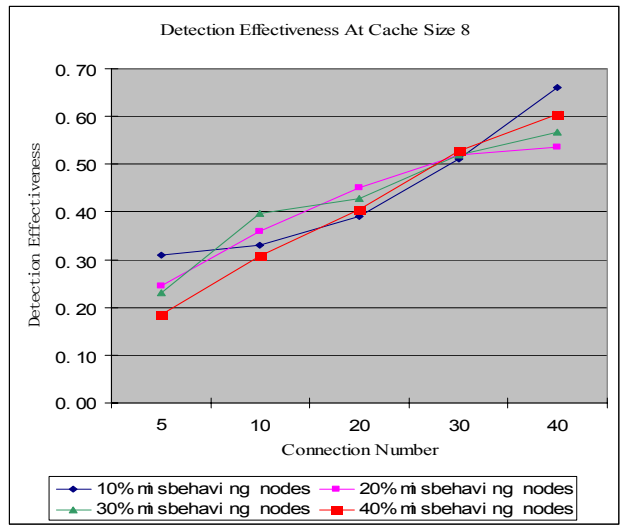

Figure 5. Detection effectiveness for different numbers of connections

Figure 5 shows the detection effectiveness for different numbers of connections. As the number of connections increase from 5 to 40, the detection effectiveness also increases. The results in Figure 5 confirm our explanation of the low detection effectiveness in Table 1.

In reality, having low detection effectiveness is not a big issue in the cache scheme. In a real protection system, there is a reaction system besides the detection system. Such hidden misbehaving nodes will be detected when the network reacts to the detection result, for example isolating the misbehaving nodes detected. At the beginning, some misbehaving nodes have no chance to misbehave. But when the detection process proceeds, the misbehaving nodes will be identified and isolated from the network. Then the hidden misbehaving nodes will have chance to drop data packets, leading them to be detected.

In the discussion above, the misbehaving node not being detected does not do anything wrong. So we should not count such "misbehaving" nodes as misbehaving when calculating the detection effectiveness. Table 2 shows the false positive and the detection effectiveness when only the nodes committing data packets dropping are counted as misbehaving.

Table 2. False Positive and Detection Effectiveness of only Counting Nodes Committing data Packets Dropping

	10% misbehaving nodes	20% misbehaving nodes	30% misbehaving nodes	40% misbehaving nodes
False Positive	0%	0%	0%	0%
Detection Effectiveness	100%	100%	99.5%	100%

As shown in Table 2, when we only count the nodes committing packet dropping as misbehaving, the detection effectiveness is almost 100% while the false positive still stays at 0%.

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 cache scheme to detect the misbehaving nodes. The features of the proposed scheme are:

- *High detection of misbehaving nodes.* The proposed scheme can detect nearly 100% misbehaving nodes in the simple dropping scenario. When only counting the nodes committing data packet dropping misbehavingly as misbehaving, the detection effectiveness in the selective dropping scenario is also nearly 100%.
- *Zero false positive.* The cache scheme can achieve almost 0% misclassification of good-behaving nodes as misbehaving nodes in both simple dropping and selective dropping scenarios.
- *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.

The simulation results also show that the cache scheme seems to have low detection effectiveness in the case of selective dropping scenario. But we should not count the "misbehaving" nodes which never commits data packet dropping as misbehaving. Thus, the right calculation of detection effectiveness should only count the nodes which commits data packet dropping as misbehaving. Now, the detection effectiveness will be nearly 100%.

The cache scheme can detect the misbehaving nodes accurately in terms of detection effectiveness and false positive in both the simple dropping and the selective dropping scenarios.

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

[9] 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.

[10] 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.

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

[12] 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.

[13] 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

[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.