Discovery of Direction-Finding Misconduct in MANETS Using 2ACK Scheme

[1]S. Aarthi, [2]D. Madhu Babu
[1]Dept. of CSE, G. Pulla Reddy Engineering College(Automonous), G.Pulla Reddy Nagar, Nandyal Road, Kurnool, AP, India
[2]Dept. of CSE, Ravindra College of Engineering for Women, Nandikotkur Road,Pasupula Village Venkayapalli, Kurnool, AP, India

Abstract
Mobile Ad hoc Networks (MANETs) operate on the basic underlying assumption that all participating nodes fully collaborate in self-organizing functions. However, performing network functions consumes energy and other resources. Therefore, some network nodes may decide against cooperating with others. Providing these selfish nodes, also termed misbehaving nodes, with an incentive to cooperate has been an active research area recently. In this paper, we propose two network-layer acknowledgment-based schemes, termed the TWOACK and the S-TWOACK schemes, which can be simply added-on to any source Steering protocol. The TWOACK scheme detects such misbehaving nodes, and then seeks to alleviate the problem by notifying the Steering protocol to avoid them in future routes. Details of the two schemes and our evaluation results based on simulations are presented in this paper. We have found that, in a network where up to 40% of the nodes may be misbehaving, the TWOACK scheme results in 20% improvement in packet delivery ratio, with a reasonable additional Steering overhead.

Keywords
Mobile Ad-Hoc Networks (MANETs), Steering misconduct , node misconduct , network security, Dynamic Source Steering (DSR)

I. Introduction

A. Mobile Ad-Hoc Network

Mobile Ad Hoc Network (MANET) can be described as an autonomous collection of mobile nodes (users) that communicate over relatively low capacity wireless links, without a centralized infrastructure. In these networks, nodal mobility and the wireless communication links may lead to dynamically changing and highly unpredictable topologies. All network functions such as Steering, multi-hop packet delivery and mobility management have to be performed by the member nodes themselves, either individually or collectively. So, network performance becomes highly dependent on collaboration of all member nodes. MANETs find applications in diverse fields ranging from low-power military wireless sensor networks to large-scale civilian applications, and emergency search/rescue operations.

An Example is shown in fig. 1. Node A can communicate directly (single hop) [4] with node C, node D and node B. If A wants to communicate with node E, node C must work as an intermediate node for communication between them. That's why the communication between nodes A and E is multi-hop. The operation of MANETs does not depend on preexisting infrastructure or base stations. Network nodes in MANETs can move freely and randomly.

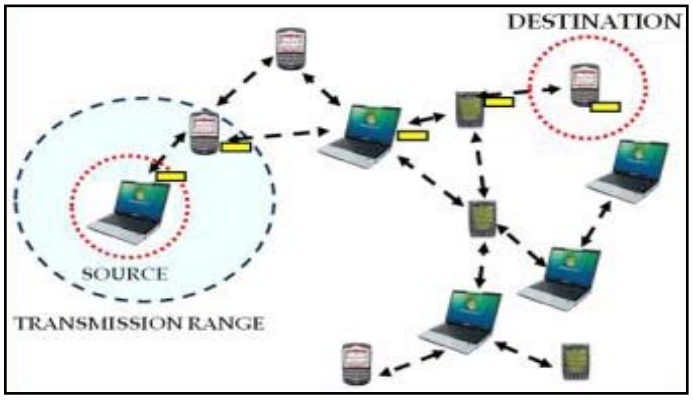

Fig. 1: A Mobile Ad-Hoc Network

There are two types of MANETs: open and closed An open MANET comprises of different users, having different goals, sharing their resources to achieve global connectivity, as in civilian applications. This is different from closed MANETs where the nodes are all controlled by a common authority, have the same goals, and work toward the benefit of the group as a whole. Open environment of a MANET may lead to misbehaving nodes. Misbehaving nodes come into existence in a network due to several reasons:

- Mobile hosts lack adequate physical protection (due to the open communication medium), making them prone to be captured and compromised;
- Usually mobile hosts are resource constrained computing devices. Performing network functions consumes significant energy of participating nodes, as communication is relatively costly.

Selfish nodes are unwilling to spend their precious resources for operations that do not directly benefit them. MANETs lack a centralized monitoring and management point, making it a challenging task to detect such misbehaving nodes effectively.

B. Characteristics of Manets

1. It having the dynamic topology, which links formed and broken with mobility.
2. Possibly uni-directional links [4].
3. Constrained resources like battery power and wireless transmitter range.
4. Network partitions.

C. Manet Steering

To find and maintain routes between dynamic topology with possibly uni-directional links, using minimum resources. The use of conventional Steering protocols in a dynamic network is not possible because they place a heavy burden on mobile computers and they present convergence characteristics that do not suit well enough the needs of dynamic networks [5]. For Example, any Steering scheme in a dynamic environment for

instance ad hoc networks must consider that the topology of the network can change while the packet is being routed and that the quality of wireless links is highly variable. The network structure is mostly static in wired networks that are why link failure is not frequent. Therefore, routes in MANET must be calculated much more frequently in order to have the same response level of wired networks. Steering schemes in MANET are classified in four major groups, namely, proactive Steering, flooding, reactive Steering, and hybrid Steering

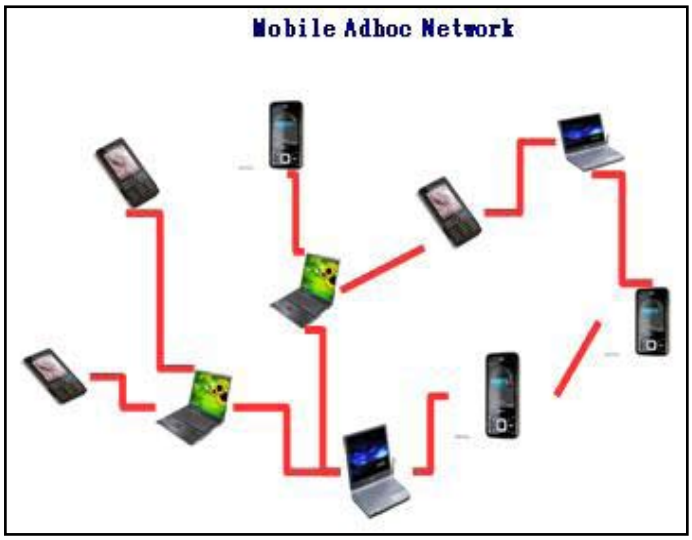

Fig. 2:

D. Misconduct of Nodes in Manet

Ad-Hoc networks increase total network throughput by using all available nodes for forwarding and Steering. Therefore, the more nodes that take part in packet Steering, the greater is the overall bandwidth, the shorter is the Steering paths, and the smaller the possibility of a network partition. But, a node may misbehave by agreeing to forward packets and then failing to do so, because it is selfish, overloaded, broken, or malicious. An overloaded node lacks the buffer space, CPU cycles or available network bandwidth to forward packets.

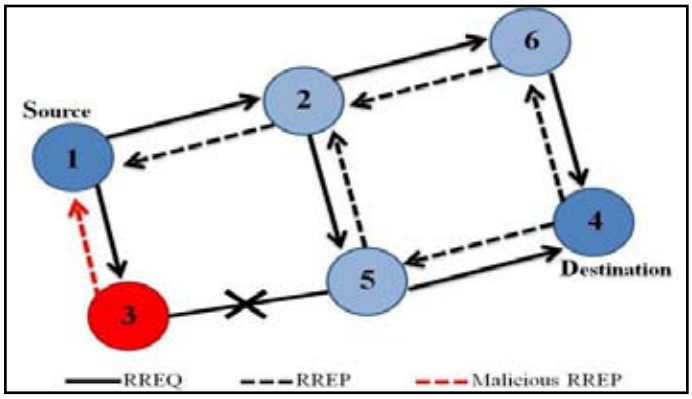

Fig. 3:

A selfish node is unwilling to spend CPU cycles, battery life or available network bandwidth to forward packets not of direct interest to it, even though it expects others to forward packets on its behalf. A malicious node creates a denial of service (DOS) attack by dropping packets. A broken node might have a software problem which prevents it from forwarding packets.

III. Related Work

These schemes can be broadly classified into two categories: Recognition-based schemes and standing-based schemes.

A. Recognition-Based Schemes

The basic idea of Recognition-based schemes is to provide incentives for nodes to faithfully perform networking functions. In order to achieve this goal, virtual (electronic) currency or similar payment system may be set up. Nodes get paid for providing services to other nodes.

Various techniques have been proposed to prevent selfishness in MANETs. As described in [15], these schemes can be broadly classified into standing-based schemes [2, 9] and Recognition-based schemes [3, 5-6], the basic idea being to provide incentives to nodes to faithfully perform networking functions. In a standing-based approach, nodes (either individually or collectively) detect, and then declare another node to be misbehaving. This declaration is then propagated throughout the network, leading to the misbehaving node being avoided in all future routes. A Recognition-based approach, on the other hand, uses the concept of virtual currency. Nodes pay virtual money for services (networking resources) that they get from other nodes, and similarly, get paid for providing services to other nodes. Since our TWOACK scheme is standing based, we only describe previous work of that category in this section. In [9], Marti et al. proposed a standing-based scheme. Two modules called watchdog and path rater are implemented at each node, to detect and mitigate, respectively, Steering misconduct s in MANETs. Nodes operate in a promiscuous mode wherein, the watchdog module overhears the medium to check whether the next-hop node faithfully forwards the packet or not. At the same time, it maintains a buffer of recently sent packets. A data packet is cleared from the buffer when the watchdog overhears the same packet being forwarded by the next hop node over the medium. If a data packet remains in the buffer too long, the watchdog module accuses the next hop neighbor to be misbehaving. Thus, the watchdog enables misconduct Discovery at the forwarding level as well as the link level. Based on watchdog's accusations, the path rater rates every path in its cache and subsequently chooses the path that best avoids misbehaving nodes. However, the watchdog technique may fail to detect misconduct in the presence of ambiguous collisions, receiver collisions, limited transmission power, false misconduct and partial dropping [9]. The CONFIDANT protocol proposed by Buchegger et al. in [2] is another example of a standing-based scheme. The protocol is based on selective altruism and utilitarianism, thus making misconduct unattractive. CONFIDANT consists of four important components - the Monitor, the Standing System, the Path Manager, and the Trust Manager. They perform the vital functions of neighborhood watching, node rating, path rating, and sending and receiving alarm messages, respectively. Each node continuously monitors the behavior of its first-hop neighbors. If a suspicious event is detected, details of the event are passed to the Standing System. Depending on how significant and how frequent the event is, the Standing System modifies the rating of the suspected node. Once the rating of a node becomes intolerable, control is passed to the Path Manager, which accordingly controls the route cache. Warning messages are propagated to other nodes in the form of an Alarm message sent out by the Trust Manager. Of course, trust relationships and Steering decisions depend on experienced, observed or reported behavior of other nodes, i.e. a node would obviously trust a firsthand experience of misconduct much more than if the misconduct were reported by a third party.

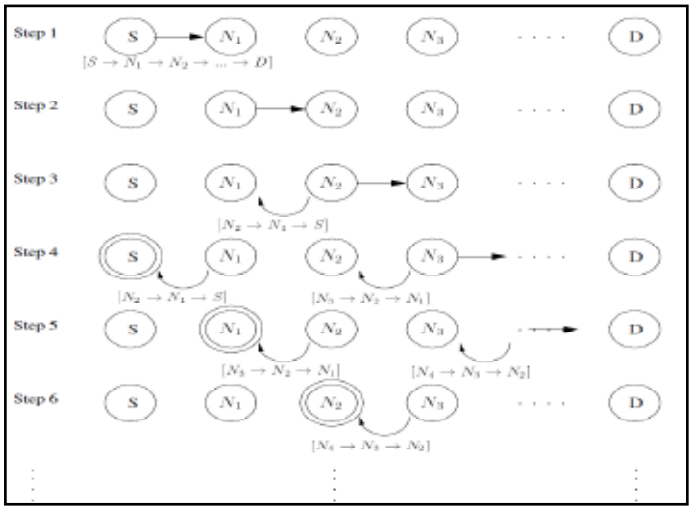

Fig. 4:

In the Packet Purse Model, nuggets are loaded into the packet before it is sent. The sender puts a certain number of nuggets on the data packet to be sent. Each intermediate node earns nuggets in return for forwarding the packet. If the packet exhausts its nuggets before reaching its destination, then it is dropped. In the Packet Trade Model, each intermediate node earns some nuggets for providing the forwarding service and the overall cost of sending the packet is borne by the destination.

The counter is decreased when the node sends packets of its own, but increased when it forwards packets for the other nodes. The counter should be positive before a node is allowed to send its packet. Therefore, the nodes are encouraged to continue to help other nodes. Tamper resistant hardware modules are used to keep nodes from increasing the nugget counter illegally. Another Recognition-based scheme, termed Sprite, was proposed by Zhong et al. [8]. In Sprite, nodes keep receipts of the received/forwarded messages. When they have a fast connection to a Recognition Clearance Service (CCS), they report all of these receipts.

Fig. 5:

The CCS then decides the charge and Recognition for the reporting nodes. In the network architecture of Sprite, the CCS is assumed to be reachable through the use of the Internet, limiting the utility of Sprite. The main problem with Recognition-based schemes is that they usually require some kind of tamper-resistant hardware and/or extra protection for the virtual currency or the payment system. We focus on standing-based techniques in this paper instead.

B. Standing-Based Schemes

Cooperation based on standing--based trust schemes in mobile ad hoc networks has been proposed with the intention of securing networks against possible selfish behaviour. Selfishness is defined as refusal of (non--malicious) nodes to participate in network activities such as packet forwarding. We analyse the effectiveness of one such cooperation enforcement mechanism, namely standing--based cooperation, using a simulated ad hoc network environment. The energy consumption of selfish nodes is also analysed, to identify whether selfish behaviour is actually beneficial to a node, and thus, whether selfish behaviour is something that might actually be observed in a real ad hoc network. We find that standing--based cooperation enforcement is only effective in non--mobile ad hoc network environment; however, we also find evidence that selfish behaviour may not have benefits when the cost of energy consumption of a node is considered. Based on the watchdog's accusations, the path rater module rates every path in its cache and sub sequently chooses the path that best avoids misbehaving nodes. Due to its reliance on overhearing, however, the watchdog technique may fail to detect misconduct or raise false alarms in the presence of ambiguous collisions, receiver collisions, and limited transmission power, as explained in [4].

The Monitor component in the CONFIDANT scheme observes the next hop neighbor's behavior using the overhearing technique. This causes the scheme to suffer from the same problems as the watchdog scheme. In [1], Miranda and Rodriguez adopted a similar approach. Each node i maintain a data structure about every other node j as an indication of what impression node i has about node j. Along with a Recognition counter, node I also maintains lists of nodes to which node j will and will not provide service. Every node periodically broadcasts relevant information in the form of a self-state message. Other nodes update their own lists based on the information contained in these self-state messages.

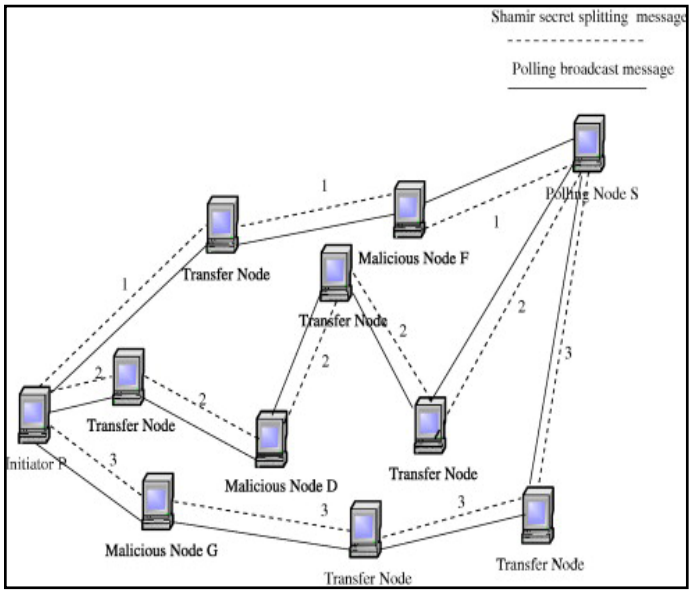

Fig. 6:

C. Back-To-Back Acknowledgment Schemes

There are several schemes that use back-to-back acknowledgments (ACKs) to detect Steering misconduct or malicious nodes in wireless networks. In the TCP protocol, back-to-back acknowledgment is employed. Such acknowledgments are sent by the end receiver to notify the sender about the reception of data packets up to some locations of the continuous data stream.

The Selective Acknowledgment (SACK) technique is used to acknowledge out-of-order data blocks. The 2ACK technique differs from the ACK and the SACK schemes in the TCP protocol in the following manner: The 2ACK scheme tries to detect those misbehaving nodes which have agreed to forward data packets for the source node but refuse to do so when data packets arrive. TCP, on the other hand, uses ACK and SACK to measure the usefulness of the current route and to take appropriate action. For example, congestion control is based on the reception of the ACK and the SACK packets. In order to identify malicious routers that draw traffic toward them but fail to correctly forward the traffic, Padmanabhan and Simon proposed the secure trace route protocol [16]. The normal trace route protocol allows the sender to simply send packets with increasing Time-To- Live (TTL) values and wait for a warning message from the router at which time the packet's TTL value expires. The secure trace route protocol authenticates the trace route packets and disguises them as regular data packets. In [17], a werbuch et al. proposed an On-Demand Secure Steering Protocol to adaptively probe faulty links on the route being used. Similarly to the secure trace route scheme, binary search is initiated on faulty routes. Asymptotically, logon probes are needed to identify a faulty link on a faulty n-hop route. This technique only works with static misconduct s and needs to disguise the probing messages as regular Steering control packets. Once a link is identified as faulty, the link weight is increased so that future link selections will avoid this link. he Best-effort Fault-Tolerant Steering (BFTR) scheme

This is compared with the predefined expected behavior of good routes. If the behavior of the route in use deviates from the behavior of good routes, it is marked as "infeasible" and a new route is used. Since BFTR throws out the entire route before detecting the misbehaving nodes, the newly chosen route may still include the same misbehaving nodes. Even though the new route will be detected as infeasible by the source after a period of observation time, data packet loss will occur in traffic flows when using protocols such as UDP. Such a repeated Discovery process is inefficient. In contrast with BFTR, we try to identify such misbehaving links in this work. In such a combined scheme, the 2ACK transmission and the monitoring processes are turned on only when Steering performance degrades. It will further reduce the Steering overhead of the 2ACK scheme. In [19], Conti et al. proposed a scheme to choose routes based on the reliability index of each outgoing neighbor. Each node maintains a table of reliability indices of its neighbors. Such a reliability index reflects the past success/failure experience of packet transmissions through this neighbor. For example, a successful back-to-back transmission will result in an increase of the reliability index of the neighbor associated with the route. When choosing routes for data transmissions, nodes prefer those rooted at the neighbors with higher reliability indices. Since a source node judges all potential routes through its immediate neighbors, the overall reliability of the chosen route depends on how the neighbors choose the rest of the route. Here, we propose a scheme to detect misbehaving links and to avoid them as much as possible.

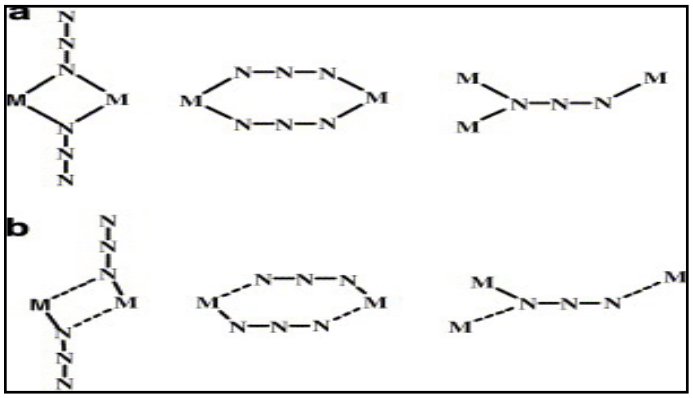

Fig. 7:

D. State-of-the-Art Schemes

The misconduct problem that we focus on in this work was referred to as the Black Hole attack in, [14, 20]. In Aad et al. investigated the Jellyfish attack for closed-loop flows such as TCP. It was shown that a Jellyfish attacker may stealthily rearrange, delay, or periodically drop packets while still remaining protocol-compliant. Such attacks may cause back-to-back throughput of closed-loop flows to drop. Similarly, the Black Hole attack was also shown to have adverse effect on open-loop flows such as UDP. Unlike [14], we propose a 2ACK technique to detect such misconduct s. Several other interesting techniques have been proposed to address the issue of potential node misconduct in MANETs. For example, Srinivasan et al. addressed the issue 490 IEEE TRANSACTIONS ON MOBILE COMPUTING, VOL. 6, NO. 5, MAY 2007 of user cooperation in MANETs [21]. The behavior of nodes was assumed to be rational, i.e., their actions were strictly determined by self-interest. A Generous TIT-FOR-TAT (GTFT) scheme was used to make sure that Nash equilibrium would be achieved. Such equilibrium will lead to optimized throughput performance for all nodes in the network. The problem of a few misbehaving nodes cannot be solved by this approach. Mahajan et al. proposed a CATCH scheme to allow cooperative nodes to detect free-riders in the neighborhood. The CATCH scheme also allows the cooperative neighbors of a free-rider to isolate it from the rest of the network.

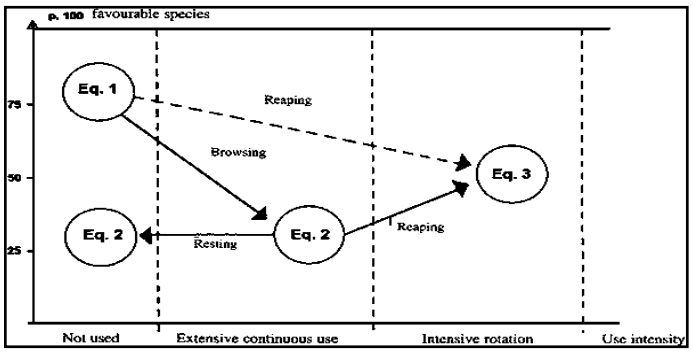

Fig. 8:

E. 2ACK and S-TWOACK Schemes

We proposed an early version of the 2ACK scheme, termed TWOACK. The 2ACK and the TWOACK schemes have the following major differences:

- The receiving node in the 2ACK scheme only sends 2ACK packets for a fraction of received data packets, while, in the TWOACK scheme, TWOACK packets are sent for every data packet received. Acknowledging a fraction of received data packets gives the 2ACK scheme better performance with

respect to Steering overhead.

- The 2ACK scheme has an authentication mechanism to make sure that the 2ACK packets are genuine. The Selective TWOACK (S-TWOACK) scheme proposed in [23] is different from 2ACK as well. Mainly, each TWOACK packet in the S-TWOACK scheme acknowledges the receipt of a number of data packets, but a 2ACK packet in the 2ACK scheme only acknowledges one data packet. With such a subtle change, the 2ACK scheme has easier control over the trade-off between the performance of the network and the cost as compared to the S-TWOACK scheme.

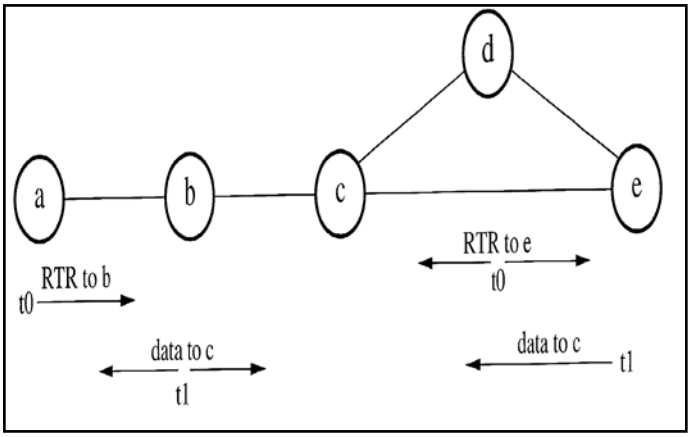

Fig. 9:

III. Problem of Steering Misconduct

In this section, we describe the problems caused by Steering misconduct. But first, we summarize the notations and assumptions used throughout this paper.

A. Notations and Assumptions

This section outlines our assumptions regarding the properties of the physical and network layers. Throughout this paper, we assume bidirectional communication. Such symmetry of links is needed for the transmission of the designed 2ACK packets. Our scheme works with source steering, such as DSR [10]. We further assume that there is no collusion among misbehaving nodes. We argue that misconduct caused by selfishness is usually limited to individual nodes in MANETs.

We use the following notations throughout the paper:

- X -Y: the size of network area.
- N: the total number of nodes in the network.
- R: the transmission range of each node. We assume that the transmission of all nodes is Omni-directional and the transmission range is homogeneous. We assume R ¼ 250 m in our simulations.
- Vm: the maximum speed of a mobile node.
- h: the average number of hops from the source node to the destination node. the expected progress of one-hop transmission.
- D: the expected distance between the source node and the destination node.
- Pm: the fraction of nodes that are misbehaving. This is also the probability of a node being a misbehaving node. The misbehaving nodes are selected among all network nodes randomly. In our simulations, pm ranges from 0 to 0.4.
- pr: the probability of a misbehaving route, i.e., the
- Probability of a route with at least one misbehaving router.
- Rmis: the threshold to determine the allowable ratio of the total number of 2ACK packets missed to the total number of data packets sent.
- Rack: the acknowledgment ratio, the fraction of data packets that are acknowledged with 2ACK packets (maintained at the 2ACK sender). : the value of timeout, beyond which time a data packet will be considered to be unacknowledged.
- Tobs: the observation period prior to declaring node misconduct.
- Cmis: the counter of missing 2ACK packets (maintained at the observing node).
- Cpkts: the counter of forwarded data packets (maintained at the observing node).

B. Steering Misconduct Model

We present the Steering misconduct model considered in this paper in the context of the DSR protocol [10]. Due to DSR's popularity, we use it as the basic Steering protocol to illustrate our proposed add-on scheme. The details of DSR can be found in [10]. The implementation of our scheme as an add-on to other Steering schemes will be discussed in Section 6. We focus on the following Steering misconduct: A selfish node does not perform the packet forwarding function for data packets unrelated to itself.2 However; it operates normally in the Route Discovery and the Route Maintenance phases of the DSR protocol. Since such misbehaving nodes participate in the Route Discovery phase, they may be included in the routes chosen to forward the data packets from the source. The misbehaving nodes, however, refuse to forward the data packets from the source. This leads to the source being confused. In guaranteed services such as TCP, the source node may either choose an alternate route from its route cache or initiate a new Route Discovery process. The alternate route packets. As a result, the network fails to provide reliable Communication for the source node even though such routes are available. In best-effort services such as UDP, the source simply sends out data packets to the next-hop node, which forwards them on. The existence of a misbehaving node on the route will cut off the data traffic flow. The source has no knowledge of this at all. In this paper, we propose the 2ACK technique to detect such misbehaving nodes. Routes containing such nodes will be eliminated from consideration. The source node will be able to choose an appropriate route to send its data. In this work, we use both UDP and TCP to demonstrate the adverse effect of Steering misconduct and the performance of our proposed scheme. The attackers (misbehaving nodes) are assumed to be capable of performing the following tasks:

- dropping any data packet,
- masquerading as the node that is the receiver of its next-hop link,
- sending out fabricated 2ACK packets,
- sending out fabricated hn, the key generated by the 2ACK packet senders, and
- claiming falsely

IV. The 2ACK Scheme

The main idea of the 2ACK scheme is to send two-hop acknowledgment packets in the opposite direction of the Steering path. In order to reduce additional Steering overhead, only a fraction of the received data packets are acknowledged in the 2ACK scheme. Thus it detects the misbehaving nodes, eliminate them and choose the other path for transmitting the data. The watchdog Discovery mechanism has a very low overhead. Unfortunately, the watchdog technique suffers from several problems such as ambiguous collisions, receiver collisions, and limited transmission

power [8]. The main issue is that the event of successful packet reception can only be accurately determined at the receiver of the next-hop link, but the watchdog technique only monitors the transmission from the sender of the next-hop link. In the next-hop link, a misbehaving sender or a misbehaving receiver has a similar adverse effect on the data packet. It will not be forwarded further. The result is that this link will be tagged. 2ACK scheme significantly simplifies the Discovery mechanism.

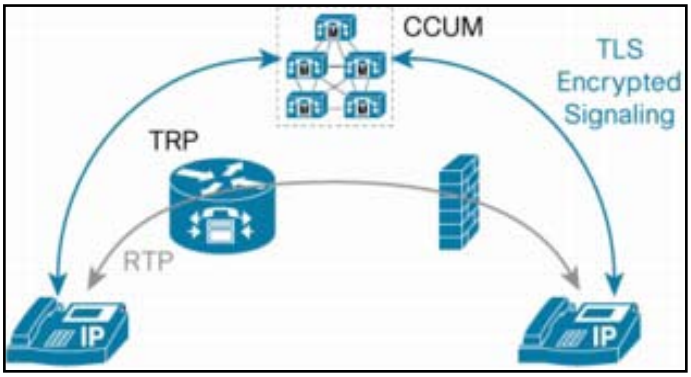

Fig. 10: Scenario for Packet Dropping and MisSteering

Noting that a misbehaving node can either be the sender or the receiver of the next-hop link, we focus on the problem of detecting misbehaving links instead of misbehaving nodes.

A. Details of the 2ack Scheme

The 2ACK scheme is a network-layer technique to detect misbehaving links and to mitigate their effects. It can be implemented as an add-on to existing Steering protocols for MANETs, such as DSR. The 2ACK scheme detects misconduct through the use of a new type of acknowledgment packet, termed 2ACK. A 2ACK packet is assigned a fixed route of two hops (three nodes) in the opposite direction of the data traffic route.

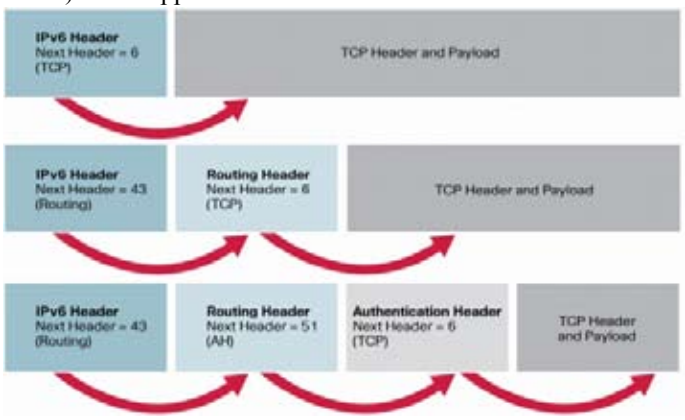

Fig. 11: Illustrates the Operation of the 2ACK Scheme

Suppose that N1, N2, N3 and N4 are three consecutive nodes (Tetra) along a route [9]. The route from a source node, S, to a destination node, D, is generated in the Route Discovery phase of the DSR protocol. When N1 sends a data packet to N2 and N2 forwards it to N3 and so on, it is unclear to N1 whether N3 or N4 receives the data packet successfully or not. Such an ambiguity exists even when there are no misbehaving nodes. The problem becomes much more severe in open MANETs with potential misbehaving nodes. The 2ACK scheme requires an explicit acknowledgment to be sent by N3 and N4 to notify N1 of its successful reception of a data packet: When node N3 receives the data packet successfully, it sends out a 2ACK packet over two hops to N1 (i.e., the opposite direction of the Steering path as shown), with the ID of the corresponding data packet. The triplet is derived from the route of the original data traffic. Such a tetra is used by N1 to monitor the link N2 >N3>N4. For convenience of presentation, we term N1 in the tetra N1 > N2> N3> N4 the 2ACK packet receiver or the observing node and N4 the 2ACK packet sender. Such a 2ACK transmission takes place for every set of tetra along the route. Therefore, only the first router from the source will not serve as a 2ACK packet sender. The last router just before the destination and the destination will not serve as 2ACK receivers.

V. Pseudocode of the 2ACK Scheme

We use the triplet in Fig. 1 as an example to illustrate 2ACK's pseudo code. Note that such codes are run on each of the sender/ receiver of the 2ACK packets.

2ACK Packet Sender Side (Node N_3)

```
1:  publish h_n  // Send authenticated element to node N_1
2:  C_pkts ← 0, C_ack ← 0, i ← n  // Initialization at node N_3
3:  while true do
4:      if (data packet received) then
5:          C_pkts ++  // Increase the counter of received packets
6:          if (C_ack/C_pkts < R_ack) then  // The data packet needs
            to be acknowledged
7:              prepare MAC with h_{i-1}
8:              prepare 2ACK with ID, MAC, h_i  // Add
                authentication to 2ACK packet
9:              send 2ACK
10:             C_ack ++, i --  // Increase the counter of
                acknowledged packets
11:         end
12:     end
13: end
```

Receiver (Observer) Side (Node N_1)

Parallel process 1 (receiving h_n)

```
1:  while true do
2:      if receive h_n from the 2ACK packet sender then
3:          record h_n, i ← n
4:      end
5:  end
```

Parallel process 2 (receiving 2ACK packets)

```
6:  while true do
7:      randomly select T_start > current time  // Start the
        observation
8:      while current time < T_start do
9:          // null
10:     end
11:     LIST ← φ, C_pkts ← 0, C_mis ← 0  // Initialization at
        node N_1
12:     while current time < T_start + T_obs do  // Observation
        period is not expired
```

```
13:     if (data packet forwarded) then
14:       LIST ← LIST ∪ data ID  // Add a data ID to LIST
15:       C_pkts ++  // Increase the counter of forwarded
          packets
16:       setup timer (τ) for data ID  // Record the time
17:     end
18:     if (2ACK packet received) then
19:       search data ID carried by 2ACK from LIST
20:       if (found) then  // A 2ACK packet for a data ID
          received
21:         check validity of h_i
22:         LIST ← LIST - data ID  // Remove data ID from
            LIST
23:         clear timer for ID
24:       end
25:     end
26:     if (timeout event happens) then  // 2ACK packet for a
        data ID is not received
27:       LIST ← LIST - data ID  // Remove data ID from
          LIST
28:       C_mis ++  // Increase misbehavior counter
29:     end
30:   end
31:   if (C_mis/C_pkts > R_mis) then  // The observation period
      expires
32:     send link misbehavior report
33:   end
34: end
```

A. Authenticating the 2ACK Packets

We look into the problem of 2ACK packet fabrication in this subsection. Since the 2ACK packets are forwarded by an intermediate node (e.g., node N2 in fig. 1), without proper protection, a misbehaving node N2 can simply fabricate 2ACK packets and claim that they were sent by node N3. Therefore, an authentication technique is needed in order to protect 2ACK packets from being forged. A straightforward way to stop N2 from forging the 2ACK packets is to use the digital signature algorithm. A digital signature is a small number of extra bits of information attached by node N3. The signature is unique and usually computationally impossible to forge unless the security key of node N3 is disclosed. Furthermore, the signature may be used to assure the integrity of the transmitted data, i.e., any changes on the signed information will be detected.

- The input can be of any length.
- The output has a fixed length.
- H(x) is relatively easy to compute for any given input x.
- It is computationally infeasible to calculate x from H(x).
- H(x) is collision-free.

An alternative technique to delivering the hn element is the "multipath transmission" mechanism. In this method, N3 sends its hn through a number of different paths. For instance, a packet carrying the hn element may be flooded to the local neighborhood. The packet has a Time-To-Live (TTL) value of two or three hops. This is similar to the broadcast of the RREQ packets in DSR. N1 employs a majority vote technique to obtain hn after it receives several copies of HN. Note that only the misbehaving N2 is interested in forging a new hn. Since a majority of the nodes are well-behaved, the true value of HN can be obtained. Once the hn element is distributed from N3 to N1, N3 can use hi (0 - i < n) sequentially to sign the 2ACK packets to be sent to N1. The hi elements will be disclosed by N3 one at a time. Assume that hiþ1 has been disclosed (initially, i ¼ n _ 1). When node N3 needs to send a 2ACK packet, it calculates a Message Authentication Code (MAC) based on hi_1, ½N2; N1; ID-hi-1, and attaches the MAC and the hi value to the 2ACK packet. Fig. 4 illustrates the packet format of a 2ACK packet. The fields in Fig. 4 are explained below:

N2: the receiver of the next hop, in the opposite direction of the route.
N1: the destination of the 2ACK packet, the observing node, that is two-hop away from the 2ACK packet sender.
ID: the sequence number of the corresponding data packet.
½N2; N1; ID-hi-1: Message Authentication Code (MAC), signed with hi_1.
Hi: the newly disclosed element in the one-way hash chain, 0 < i < n.
In this work, we do not study the overhead caused by the authentication of the 2ACK packets. Compared to traditional security measures, the computation cost of the one-way hash function is relatively low [26]. The communication overhead depends on the length of each element and the value of n, i.e., the size of the one-way hash chain. When n and the size of each element are chosen reasonably, We expect low overhead due to the transmission of HN.

B. Timeout for 2ACK Reception

The parameter timeout will be used to set up a timer for 2ACK reception. If the timer expires before the expected 2ACK packet is received, the missing 2ACK packet counter, Cmis, will be incremented. Thus, an appropriate value of important for the successful operation of the 2ACK scheme. It is clear that false alarms may be triggered if too small. On the other hand, if too large, the observing node will have to maintain a longer list, requiring a large memory size. Therefore, should be set at a value that is large enough to allow the occurrence of temporary link failures (for example, the unsuccessful transmission due to node mobility or local traffic congestion). It is essential that should satisfy
Where a single-hop transmission delay includes packet transmission delay, random back-off delay at the Medium Access Control (MAC) layer, data processing delay, and potential retransmission delay.

C. Acknowledgment Ratio, Rack

The additional Steering overhead caused by the transmission of the 2ACK packets can be controlled by the parameter acknowledgment ratio, Rack, at the 2ACK packet sender. With the use of the parameter Rack in the 2ACK scheme, only a fraction of the received data packets will be acknowledged. Therefore, the parameter Rack provides a mechanism to tune the overhead.5 The reduction of overhead comes with a cost: the shrinking of the range over which Rmis can take values.

D. Partial Data Forwarding

A misbehaving node may forward data packets partially by forwarding a fraction of the packets and try to cheat the monitoring system. Such a behavior will be detected by the 2ACK scheme. We use the triplet N1 ! N2! N3 in Fig. 1 as an example for explanation. Assume a misbehaving node N2 receives ND data packets from N1 successfully and only forwards a fraction of the data packets, say, Rpart, of ND toward N3. We further assume that all data packets forwarded by N2 are successfully received by N3. Thus, N3 receives Rpart _ ND data packets and only Rack Rpart _ ND of them will be acknowledged by 2ACK packets sent from N3.

$1 - R_{ack} \cdot R_{part} < R_{mis}$.

As the gap between 1 _ Rack and Rmis shrinks, the feasible value of Rpart approaches 1. Therefore, the 2ACK scheme effectively guards against partial forwarding.

$$R_{part} > \frac{1 - R_{mis}}{R_{ack}}$$

Thus, by increasing, we force N2 to forward more data packets. The disadvantage of such an approach is the loss of protection from false alarms

VI. Advantages

As compared to the watchdog, the 2ACK scheme has the Following advantages:

A. Flexibility [9]

One advantage of the 2ACK scheme is its flexibility to Control overhead with the use of the Rack parameter.

B. Consistent Data Transmission

It deals with the reliable transfer of file from source to destination. The file needs to be stored at source for certain amount of time even if it has been transmitted. This will help to resend the file if it gets lost during transmission from source to destination.

C. Reliable Route Discovery [10]

Reliable Route Discovery deals with discovering multi-hop route for wireless transmission. Steering in a wireless ad-hoc network is complex. This depends on many factors including finding the Steering path, selection of routers, topology, protocol etc.

D. Narrow Overhear Range [10]

A well-behaved N3 may use low transmission power to send data toward N4. Due to N1's limited overhearing range, it will not overhear the transmission successfully and will thus infer that N2 is misbehaving, causing a false alarm. Both this problem occurs due to the potential asymmetry between the communication links. The 2ACK scheme is not affected by limited overhearing range problem.

E. Partial Transmission Power

A misbehaving N2 may maneuver its transmission power such that N1 can overhear its transmission but N4 cannot. This problem matches with the Receiver Collisions problem. It becomes a threat only when the distance between N1 and N2 is less than that between N2 and N3 and so on. The 2ACK scheme does not suffer from limited transmission power problem.

VII. Conclusion

The proposed system is a simulation of the algorithm that detects misbehaving links in Mobile Ad Hoc Networks. The 2ACK scheme identifies misconduct in Steering by using a new acknowledgment packet, called 2ACK packet. A 2ACK packet is assigned a fixed route of two hops (four nodes N1, N2, N3, N4), in the opposite direction of the data traffic route. The system implements the 2ACK scheme which helps detect misconduct by a 3 hop acknowledgement. The 2ACK scheme for detecting Steering misconduct is considered to be network layer technique for mitigating the Steering effects. Mobile Ad Hoc Networks (MANETs) have been an area for active research over the past few years due to their potentially widespread application in military and civilian communications. Such a network is highly dependent on the cooperation of all of its members to perform networking functions. This makes it highly vulnerable to selfish nodes. One such misconduct is related to Steering. When such misbehaving nodes participate in the Route Discovery phase but refuse to forward the data packets, Steering performance may be degraded severely. In this paper, we have investigated the performance degradation caused by such selfish (misbehaving) nodes in MANETs. We have proposed and evaluated a technique, termed 2ACK, to detect and mitigate the effect of such Steering misconduct .

The 2ACK technique is based on a simple 2-hop acknowledgment packet that is sent back by the receiver of the next-hop link. Compared with other approaches to combat the problem, such as the overhearing technique, the 2ACK scheme overcomes several problems including ambiguous collisions, receiver collisions, and limited transmission powers. The 2ACK scheme can be used as an add-on technique to Steering protocols such as DSR in MANETs. We have presented the 2ACK scheme in detail and discussed different aspects of the 2ACK scheme. Extensive simulations of the 2ACK scheme have been performed to evaluate its performance. Our simulation results show that the 2ACK scheme maintains up to 91 percent packet delivery ratio even when there are 40 percent misbehaving nodes in the MANETs that we have studied. The regular DSR scheme can only offer a packet delivery ratio of 40 percent. The false alarm rate and Steering overhead of the 2ACK scheme are investigated as well. One advantage of the 2ACK scheme is its flexibility to control overhead with the use of the Rack parameter. In this work, we have focused only on link misconduct . It is more difficult to decide the behavior of a single node. This is mainly due to the fact that communication takes place between two nodes and is not the sole effort of a single node. Therefore, care must be taken before punishing any node associated with the misbehaving links. When a link misbehaves, either of the two nodes associated with the link may be misbehaving. In order to decide the behavior of a node and punish it, we may need to check the behavior of links around that node. This is a potential direction for our future work.

References

[1] J. J. Garcia-Luna-Aceves et al.,"Source Tree Adaptive Steering (STAR) protocol", draft-ietf-manet-star-00.txt, 1998, IETF Internet Draft.

[2] D. Johnson, D. Maltz, Y.C. Hu, J. Jetcheva,"The Dynamic Source Steering Protocol for Mobile Ad-Hoc Networks (DSR) in 10th IEEE International Conference", 27-30 Aug 2002,Year of Publication :2002, ICON 2002

[3] Kong, P. Zerfos, H. Luo, S. Lu, L. Zhang,"Providing Robust and Ubiquitous Security Support for Mobile Ad-Hoc Networks", Proc. IEEE Int"l Conf. Network Protocols (ICNP "01), 2001.

[4] L.M. Feeney, M. Nilsson,"Investigating the Energy Consumption of a Wireless Network Interface in an Ad-Hoc Networking Environment", Proc. IEEE INFOCOM, 2001: Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies, Vol. 3, pp.1548-1557, 2001.

[5] Elizabeth Royer, C-K Toh,"A Review of Current Steering Protocols for Ad-Hoc Mobile Wireless Networks", IEEE Personal Communications Magazine, pp. 46-55, April 1999.

[6] Josh Broch, David A. Maltz, David B. Johnson, Yih-Chun Hu, Jorjeta Jetcheva,"A performance comparison of multi-hop wireless Ad-Hoc network Steering protocols",

Proceedings of the 4th annual ACM/IEEE international conference on Mobile computing and networking, pp.85-97, October 25-30, 1998, Dallas, Texas, United States [doi>10.1145/288235.288256].

[7] K. Balakrishnan, J. Deng, P.K. Varshney,"TWOACK: Preventing Selfishness in Mobile Ad-Hoc Networks", Proc. IEEE Wireless Comm.and Networking Conf. (WCNC ‟05), Mar. 2005, Vol. 4, pp. 2137-2142, IEEE Press 2005,Year of Publication:2005

[8] Charles E. Perkins, Pravin Bhagwat,"Highly dynamic Destination Sequenced Distance-Vector Steering (DSDV) for mobile computers", ACM SIGCOMM Computer Communication Review, Vol. 24, no. 4, pp. 234-244, Oct. 1994.

[9] Charles E. Perkins, Elizabeth M. Royer,"Ad-Hoc On-Demand Distance Vector Steering", Proceedings of the Second IEEE Workshop on Mobile Computer Systems and Applications, pp. 90, February 25-26, 1999.

[10] David B. Johnson, David A. Maltz,"Dynamic Source Steering (DSR) in Ad-Hoc Wireless Networks". In Mobile Computing, edited by Tomasz Imielinski and Hank Korth, chapter 5, pp. 153-181. Kluwer Academic Publishers, 1996.

[11] L. Zhou, Z.J. Haas,"Securing Ad Hoc Networks", IEEE Network Magazine, Vol. 13, No. 6, Nov./Dec. 1999.

[12] F. Stajano, R. Anderson,"The Resurrecting Duckling: Security Issues in Ad-Hoc Wireless Networks", Proc. Seventh Int'l Workshop Security Protocols, 1999.

[13] J. Kong, P. Zerfos, H. Luo, S. Lu, L. Zhang,"Providing Robust and Ubiquitous Security Support for Mobile Ad-Hoc Networks", Proc. IEEE Int'l Conf. Network Protocols (ICNP '01), 2001.

[14] I. Aad, J.-P. Hubaux, E-W. Knightly,"Denial of Service Resilience in Ad-Hoc Networks", Proc. MobiCom, 2004.

[15] L. Buttyan, J.-P. Hubaux,"Stimulating Cooperation in Self-Organizing Mobile Ad Hoc Networks", ACM/Kluwer Mobile Networks and Applications, Vol. 8, No. 5, 2003.

[16] V.-N. Padmanabhan, D.-R. Simon,"Secure Traceroute to Detect Faulty or Malicious Steering", SIGCOMM Computer Comm. Rev., Vol. 33, No. 1, Jan. 2003.

[17] B. Awerbuch, D. Holmer, C.-N. Rotaru, H. Rubens,"An On-Demand Secure Steering Protocol Resilient to Byzantine Failures", Proc. ACM Workshop Wireless Security (WiSe), Sept. 2002.

[18] Y. Xue, K. Nahrstedt,"Providing Fault-Tolerant Ad-Hoc Steering Service in Adversarial Environments", Wireless Personal Comm., Vol. 29, No. 3-4, pp. 367-388, 2004.

[19] M. Conti, E. Gregori, G. Maselli,"Towards Reliable Forwarding for Ad-Hoc Networks", Proc. Personal Wireless Comm.(PWC '03), Sept. 2003.

[20] Y. Hu, A. Perrig, D.B. Johnson,"Ariadne: A Secure On-Demand Steering Protocol for Ad Hoc Networks", Proc. MobiCom, Sept. 2002.

[21] V. Srinivasan, P. Nuggehalli, C.F. Chiasserini, R.R. Rao,"Cooperation in Wireless Ad Hoc Networks", Proc. INFOCOM,Mar.-Apr. 2003.

[22] R. Mahajan, M. Rodrig, D. Wetherall, J. Zahorjan, "Sustaining Cooperation in Multi-Hop Wireless Networks", Proc. Second Symp.Networked Systems Design and Implementation, Apr. 2005.

[23] K. Balakrishnan, J. Deng, P.K. Varshney,"TWOACK: Preventing Selfishness in Mobile Ad-Hoc Networks", Proc. IEEE Wireless Comm. and Networking Conf. (WCNC '05), Mar. 2005.

[24] M. Mathis, J. Mahdavi, S. Floyd, A. Romanow,"RFC 2018—TCP Selective Acknowledgement Options", technical report, PSC, LBNL, Sun Microsystems, Oct. 1996.

[25] D.B. Johnson,"ECC, Future Resiliency and High SecuritySystems", white paper, Certicom, www.certicom.com, Mar. 1999.

[26] Y. Hu, D.B. Johnson, A. Perrig,"SEAD: Secure Efficient Distance Vector Steering for Mobile Wireless Ad-Hoc Networks", Ad Hoc Networks, Vol. 1, No. 1, pp. 175-192, 2003.

[27] L. Lamport,"Password Authentication with Insecure Communication", Comm. ACM, Vol. 24, No. 11, pp. 770-772, Nov. 1981.

[28] C.E. Perkins, P. Bhagwat,"Highly Dynamic Destination-Sequenced Distance-Vector Steering (DSDV) for Mobile Computers", Proc. ACM Special Interest Group on Data Comm. (SIGCOMM

[29] R.L. Rivest,"RFC 1321—The MD5 Message-Digest Algorithm", technical report, MIT Laboratory for Computer Science and RSa Data Security, Inc., Apr. 1992.

AARTHI.S has completed the course of B.Tech in CSE and M.Tech in Computer Science (CS) from Rajeev Gandhi Memorial College of Engineering & Technology, Nandyal. Currently she is working as Assistant Professor in G.Pulla Reddy Engineering College, KURNOOL, Andhra Pradesh, and started research work in the area of " Discovery Of Direction-finding Misconduct In MANETS Using2ACK Scheme" in Ad hoc Networks. Her research interests include COMPUTER NETWORKS, Object Oriented Programming, Cloud Computing.

MADHU BABU.D has completed the course of B.Tech in CSE and M.Tech in Computer Science (CS) from RAJEEV GANDHI MEMORIALCOLLEGEOFENGINEERING&TECHNOLOGY,Nandyal. Currently He is working as Assistant Professor in Ravindra College of Engineering for Women, Kurnool,Andhra Pradesh, and started research work in the area of " Discovery Of Direction-finding Misconduct In MANETS Using2ACK Scheme" in Ad hoc Networks. His research interests include COMPUTER NETWORKS.