# DETECTING AND MITIGATING DOS ATTACKS IN WIRELESS NETWORKS WITHOUT AFFECTING THE NORMAL BEHAVING NODES

Yi Xu and Wenye Wang
Department of Electrical and Computer Engineering
North Carolina State University
Raleigh, NC 27606
Email: {yxu2, wwang}@ncsu.edu

*Abstract*— **In this paper we investigate the DoS attack detection and mitigation problem in wireless networks. The DoS attacks are difficult to mitigate because the legitimate nodes can also generate large amount of packets in a short time. The difficulty in differentiating between the malicious nodes and the legitimate nodes always prevents the DoS detection and mitigation schemes from achieving satisfactory performance. We propose a new scheme for DoS mitigation, which requires a node to undertake packet forwarding responsibility if it sends large amount of packets through other nodes. The responsibility is proportionate to the amount of packets the network delivers for the node. By placing this requirement, we are able to differentiate the normal nodes from the malicious nodes, since a normal node is willing to undertake its responsibility while a malicious node would not. However, if a malicious node drops the packets that are supposed to be forwarded, its neighbors are able to detect it and then isolate the malicious node. As the result, a malicious node will have to either pay for its attack by helping forward other nodes' packets or drop the packets and then be isolated.**

## I. INTRODUCTION

The Denial-of-Service (DoS) attacks continue to pose a grave risk to the network users even though this kind of attack is known for several years. In fact, 39 percent of the surveyed security professionals reported experiencing DoS attacks in 2004 [1]. The DoS attacks target the service availability to deny the authorized users from accessing the services. They come in a variety of forms and have a variety of objectives. CERT/CC described three basic types of DoS attacks [2]: (1) consumption of scarce, limited, or non-renewable resources, (2) destruction or alteration of configuration information, (3) physical destruction or alteration of network resources.

The DoS attacks take place in various network layers: from the physical layer all the way up to the application layer. The attackers choose different targets in different layers. The DoS attacks can also be launched in many network topologies, including both wired and wireless. In the multihop ad hoc wireless networks, nodes communicate without any fixed infrastructure such as access points or base stations. The self-organization structure exposes the nodes to higher risk of potential attacks than the infrastructure-based networks. The DoS attacks on ad hoc wireless networks are especially problematic since the nodes in such topology have to readily accept routing information and data forwarding requests from each other in order to ensure the correct functioning of the network. Resistence to DoS attacks is therefore important in order to guarantee the ad hoc network performance.

There are two types of network layer DoS attacks in the ad hoc networks, the routing attacks and the traffic attacks. Though their ultimate goals are both to disrupt the correct delivery of packets, they use different approaches. The routing attacks advertise false routing messages to mislead the legitimate nodes into wrong routing decisions. The attackers may cause blackhole [3], congestion, or path loop. The attacks on data traffic disrupt the legitimate traffic transmission by injecting large amount of junk traffic that takes away the usable bandwidth or by silently dropping the legitimate packets when they travel through the attackers.

We study the traffic attack in this paper. By launching a flooding attack, a malicious node can severely burden the network, thereby effectively degrading the overall network performance. An ideal countermeasure to the traffic attack would be shutdown of the malicious traffic and avoidance of forwarding legitimate traffic through the malicious nodes. However, this is always difficult to achieve. The malicious traffic is difficult to be distinguished from the normal traffic, because the normal behaving nodes might also have large amount of legitimate traffic to send and the wily attackers might flood the network with controlled and less detectable traffic rates. Adopting a rigid mechanism to rate-limit the suspected nodes may cause loss of meaningful

1-4244-1513-06/07/$25.00©2007 IEEE.

data and adopting a slack approach may allow the malicious nodes to successfully continue with their attacks. As such, differentiation of the malicious nodes from the normal nodes is the key to successfully mitigating the attacks.

In this paper we present the Reciprocal Routing Protocol (RRP), a new measure to detect and mitigate IP layer DoS attacks in wireless networks. We differentiate between the malicious nodes and the normal nodes in a new perspective: instead of assuming the nodes with high traffic rates to be malicious, we take a fairness approach. In our scheme, if a node has high traffic rates, its neighbors will try to use that node more often to forward their traffic, since this will be fair to all the neighbor nodes. This fairness based routing scheme does not judge if the suspected node is malicious or not, so it avoids the false positive problem seen in the usual rate-limit methods that often penalize the normal nodes as a side effect of blocking the malicious nodes. The underlying principle is that a normal node will be happy to serve its neighbors since its traffic is being forwarded with the help from its neighbors but a malicious node will probably not. This fairness routing actually incurs extra workload on the malicious nodes. However, we are still cautious in preventing the malicious nodes from dropping the legitimate traffic going through them. We use a node testing technique to monitor if the legitimate traffic is dropped by a suspected node or not. If packet dropping is detected, the suspected node is then confirmed to be malicious and follow-up measures are taken to isolate it. Otherwise, the node is proven to be normal behaving. In summary, the RRP and node misbehavior detection protocols mitigate DoS attacks without negatively impacting the normal nodes.

The rest of this paper is organized as follows. Section II describes and compares the related DoS attack detection and mitigation schemes. Section III presents our new scheme to counter DoS attackes. The validity and performance of this new scheme are evaluated in Section IV. Finally, we conclude this paper in Section V.

## II. Related Work

The Aggregate-based Congestion Control (ACC) [4] was proposed by Mahajan *et al*. An aggregate is defined as a collection of packets that share some property. This technique provides some mechanisms for detecting and controlling aggregates at a router using an attack signature and applying a pushback mechanism. It depends on the mechanism by which attacks are carried out and therefore may at times not be able to apply meaningful restriction on the attackers, especially when they vary their traffic characteristics over time. The ACC scheme also pushes back all the traffic irrespective of the fact if it is genuine or junk. So the legitimate traffic is also likely to suffer.

The Congestion Puzzles (CP) [5] proposed by Wang *et al* is a new countermeasure to bandwidth-exhaustion attacks. It attempts to force attackers to invest vast resources in order to effectively perform DoS service attacks. The calculation that a node has to do in order to transmit some data is directly proportional to the amount of data it needs to send, such that it becomes resource expensive for an attacker to transmit large amount of junk packets. But puzzle solving also slows down the transmission of the legitimate packets. In comparison, our scheme does not slow down the transmission of legitimate packets.

Filtering techniques are used in several methods, such as SAVE [6] and Hop-by-hop Authentication [7]. These approaches are of less utility against non-spoofed traffic. In addition, these schemes rely upon some way of distinguishing attack packets from legitimate ones.

SAVE [6] proposed by Li *et al* forces all IP packets to carry correct source addresses and requires each router along the way to build an incoming table that associates each incoming interface of the router with a set of valid source address blocks. However this scheme permits a malicious node to carry out the attack by using a subnet-wide randomized IP address in its packets thereby not getting detected. A malicious node could also use its neighbors' IP addresses in random order to pretend that it is forwarding their packets, thereby injecting useless data into the network. However our scheme does not depend upon the verity of a data packet but on the verity of the transmitting node. Once that is established, the node is allowed to send as many data packets as it requires.

Hop-by-hop Authentication [7] proposed by Zhu *et al* presents an interleaved hop-by-hop authentication scheme that guarantees that the base station will detect any injected false data packets when no more than a threshold number of nodes are compromised. This scheme assumes a static security setup which is only available in sensor networks, so it may not work when the path of data transmission changes over time. Also like several others, this scheme judges the packet legitimacy whereas our scheme judges the node legitimacy to establish communications.

In [8], Marti *et al* proposed a way of detecting malicious nodes through overhearing the next node's transmissions. They introduced the concept of watchdog and pathrater. The watchdog identifies misbehaving nodes, while the pathrater avoids routing packets through these nodes. The limitation of this method is that it cannot detect the malicious nodes who flood the network without dropping others' packets.

The reputation-based schemes such as CONFIDANT [9], SORI [10], and Catch [11] discourage node selfishness. They encourage packet forwarding and discipline selfish nodes by quantifying the reliability of a node through objective measures. Similar to [8], these schemes detect the selfish nodes but do not address the problem of mitigating the flooding attacks.

M. Just *et al* [12] present a proactive distributed probing protocol to detect and mitigate the malicious packet dropping attacks. In their approach, every node proactively monitors the packet forwarding behavior of the others by mixing probing messages into the usual traffic. The probing messages look indistinguishable from normal packets, and they may be piggybacked on regular packets. A node infers the legitimacy of its neighbor from the received acknowledgments in response to the probes. In their protocol, the probing packet anonymity is achieved through packet encryption, while our approach utilizes the randomized node addresses and port numbers to hide the probing packets. In networks where packet encryption is not widely used, packet address and port randomization achieves better communication anonymity than encryption.

## III. Reciprocal Routing and Misbehavior Detection for Attack Mitigation

In this section we present our new scheme for DoS detection and mitigation. It consists of two steps. In the first step, we achieve the fairness of routing cost among the nodes, which is done without determining the node legitimacy. Every node is assumed to be normal behaving, though they differ in the traffic transportation demand. The fairness is defined as commensurate routing workload with respect to each node's traffic request. In short, if a node requests its neighbors to forward some packets, it should also accommodate its neighbors' packet forwarding requests in proportion. In the second step, we detect malicious packet dropping and isolate the misbehaving nodes. The normal behaving nodes are not penalized in this scheme, but the malicious nodes will be detected and isolated or if they choose not to drop others' packets then they will have to compensate for their flooding attacks by forwarding the legitimate packets.

### A. Reciprocal Routing Protocol

The geographical routing algorithm [13] is used as the foundation of our Reciprocal Routing Protocol (RRP). In geographical routing, the knowledge of node locations is assumed to be known. Figure 1 depicts an example of route selection in geographical routing. When node A has a packet to forward to D, it finds out one of its neighbors that is the closest to the destination D, for example B in this diagram, and then uses this node as the next hop. Node B, when having received the packet, locates its next hop in the similar way.

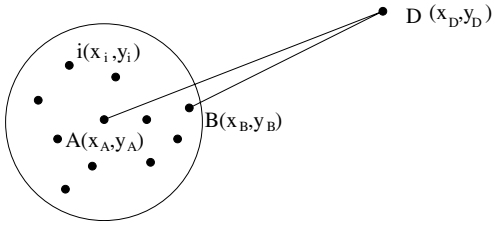


Fig. 1. The Geographical Routing Protocol.

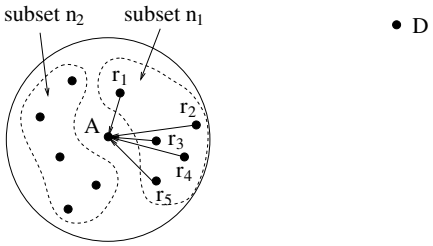


Fig. 2. The Reciprocal Routing Protocol.

We build our routing protocol upon the geographical routing algorithm for its flexibility in choosing the next hop node. In the reciprocal routing, a node distributes its outgoing traffic to its neighbors in proportion to the incoming traffic from the neighbors. This is achieved by maintaining a traffic rate list for the neighbors. For example, in Figure 2, suppose node A has $n$ neighbors, of which $n_1$ are closer to node D than node A and $n_2$ are farther to node D than node A. Among the $n_1$ nodes, each of them has sent some packets to node A recently. Let us denote the packet rate as $r_i$ for node i. When node A has a packet to send out to D, it chooses node i with probability $\frac{r_i}{\sum_{j=1}^{n_1} r_j}$. In this way, A's outgoing packets are distributed to the neighbors in proportion to the forwarding service A has offered to them. Besides, we see that if a malicious node i has a large $r_i$, the traffic load imposed on it by its neighbors will be accordingly high too.

### B. Misbehavior Detection

To detect if the next hop node is dropping the packets forwarded to it, node A randomly inserts testing packets into the packet stream. These testing packets will request a selected remote node to acknowledge A. In case A does not receive the correct acknowledgment, A will determine that its next hop node is dropping its packets. Although it

TABLE I
NOTATIONS

| | |
|---|---|
| $addr_v$ | IP address of node v |
| $port_v$ | Port number of node v |
| $k_{u,v}$ | Session key shared between node u and node v |
| $Sig_v$ | Signature of node v |
| $E_v[\cdot]$ | Encryption using node v's public key |
| $ticket_v$ | Ticket issued to node v |
| saddr | Source IP address |
| sport | Source port number |
| daddr | Destination IP address |
| dport | Destination port number |
| packetcount | Number of testing packets received |

is possible that other malicious nodes on the transmission path of the testing packets could also result in an incorrect acknowledgment, the other malicious nodes would be detected and isolated by their respective neighbors. Therefore A can use the acknowledgment to infer its next hop node's legitimacy with certain confidence.

However, in order to prevent the malicious nodes from knowing which packets are for testing and which are not, we must hide the testing packets in normal traffic. Otherwise, the malicious nodes would drop the normal traffic, while transmitting only the testing packets. We use anonymous communications to achieve this purpose. The requirements for the anonymous communications are: 1) the identities of the testing nodes must be anonymous to the malicious node, and 2) the port numbers used in the testing communications must also be anonymous. Knowing any of them will enable the malicious node to recognize the testing packets out of the usual traffic.

To facilitate the protocol presentation, we list all the notations in Table I. Our detection protocol works as follows. Assume each node in the network has a public key and private key pair. Suppose node A wants to test its neighbor node B. Node A first chooses a co-testing node T and prepares a ticket as $ticket_T = E_T[addr_A||port_A||addr_T||k_{A,T}||Sig_A]$. Then node A chooses a random node R and sends to R the message $E_R[addr_A||addr_T||ticket_T||Sig_A]$. When R receives this message, it decrypts the message and understands that the ticket should be forwarded to T. When T receives the ticket from R, it decrypts the ticket and understands A is requesting it to be a co-testing node. The ticket tells T that all of its communication to A should be directed to A's port $port_A$. T then prepares a ticket to A as $ticket_A = E_A[addr_T||port_T||addr_X||port_Y||Sig_T]$, and sends the message back to R as $E_R[addr_T||addr_A||ticket_A||Sig_T]$. When R receives and decrypts this message, R forwards $ticket_A$ to A. When A reads $ticket_A$ in the message from R,

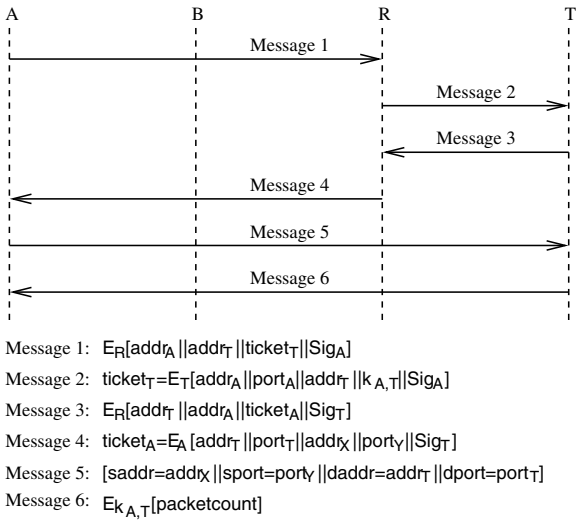


Fig. 3. The message exchanges for node misbehavior detection.

it is informed that all the testing packets should be directed to T's port $port_T$ and A should use $addr_X$ and $port_Y$ as the source address and source port in the IP packets (the reason for using this spoofed source address and source port will be explained later in the protocol validation section). Then A starts to send testing packets to T through B at random time. The testing packets bear the source address $addr_X$, source port $port_Y$, and the destination address $addr_T$, destination port $port_T$. In most of today's networks the correctness of the source address and source port in IP packets is not verified while the packets are transmitted to the destinations, so using $addr_X$ and $port_Y$ does not cause routing problems. When A has finished sending its testing packets, it requests T to acknowledge the number of packets received. T replies A with message $E_{k_{A,T}}[packetcount]$. If $packetcount < c$, where $c$ is a threshold defined by A, A determines that B is misbehaving; otherwise, A believes B is a normal node. A diagram of the message exchanges is presented in Fig. 3.

Once a node detects a misbehaving neighbor, the node blocks all the traffic from the misbehaving neighbor and does not forward legitimate traffic through the misbehaving neighbor any more. When all the neighbors of the misbehaving node have detected and isolated it, the misbehaving node cannot do any harm to the network any more.

## IV. VALIDATION AND PERFORMANCE EVALUATION OF THE PROTOCOL

### A. Anonymity of Testing Communications

The functioning of our protocol is determined by the anonymity of the testing communications. It is vital to

make this communication undetectable by the malicious node under test. Next we prove that our protocol achieves this goal.

First, we show that the identity of the co-testing node T is hidden from node B which is under test. The message sent from A to R is encrypted by A, so that B does not know the final intended destination of this message. B only knows this is a communication between A and R. Besides, B cannot modify the message since it is signed by A. Similarly, when T replies A through R, B only knows that this is another communication happening between R and A. Since $\mathsf{ticket}_\mathsf{A}$ is signed and encrypted by T, B cannot read or modify it. Also, B has no way to tell who has signed the ticket, because the ticket and T's signature are encrypted using A's public key. On the other hand, node A knows this ticket is a reply from T since A has chosen T as the co-testing node. We note that it is also unwise for node B to drop any of these two handshake messages exchanged between A and T via R. Unsuccessful handshake will lead to the conclusion that B is misbehaving.

There is a special scenario we need to discuss here though. If A unluckily chooses a malicious node as R or T, the anonymity of the testing communication might be compromised. However, since the percentage of the malicious nodes in the entire network is low in a realistic network (otherwise the network will not function anyway), the chance of choosing malicious nodes as R or T is low. In addition, because R and T do not know who A intends to test (B in this example), they have no idea who they should inform of their identities. If they simply make their identities public to every node, node A will also detect they are disclosing their identities and therefore reselect other R' or T' for the test. A possible way for malicious R or T to inform B but not being detected by A is to send separate message to every node except A and encrypt each message with the public key of the informed node. This is however too expensive to do. Therefore, the only feasible harm that R or T can do is not to cooperate when A is testing a legitimate node to mislead A into believing the legitimate neighbor is malicious. We mitigate this problem by testing a suspicious node from time to time and with different R and T selections.

Second, we show that the testing traffic does not take place on fixed ports. This prevents the malicious nodes from distinguishing the testing traffic out of the usual traffic based on the port numbers. Our protocol lets A choose $\mathsf{port}_\mathsf{A}$ and T choose $\mathsf{port}_\mathsf{T}$, which vary from node to node and from session to session, thus hiding the port numbers used in a testing session.

Third, as the testing packets are always originated from neighbors (for example A and B are neighbors), a clever malicious node might take advantage of this information such that it never drops packets originated from its neighbors but drops all the other packets. To prevent this, we randomize the source address and the source port in the testing packets sent out from A to make them look like from other sources. In order to enable T to recognize this traffic, T chooses the randomized source address $\mathsf{addr}_\mathsf{X}$ and source port number $\mathsf{port}_\mathsf{Y}$. T may make this selection based on its current ongoing communication sessions: it chooses $\mathsf{addr}_\mathsf{X}$ and $\mathsf{port}_\mathsf{Y}$ as long as T is not communicating to node X and its port Y at this time.

Till now we have shown that the testing packets are transmitted to the co-testing node undetected by the node under test. In the final step of the test, T acknowledges A the number of testing packets it has received. To avoid the expensive computations in public key encryption and private key signature, this acknowledgment is encrypted using the secret key $\mathsf{k}_{\mathsf{A,T}}$. We show that this encryption is sufficient to prevent the malicious node from modifying it. As the node under test does not know how many testing packets have been sent through it and it cannot read the packet count in the acknowledgment, randomly modifying the encrypted acknowledgment will result in a count far from the actual number. If A sees a packet count larger than the actual number of testing packets it has sent, it detects the modification and believes the node under test is malicious. A also believes the node to be malicious when the packet count is unexpectedly less than what it has sent. The chance for a malicious node to correctly increase the value of the packet count is very slim. Moreover, since the malicious node does not know the identity of T, it is very difficult for it to even find the acknowledgment packet in the packet stream going through it in the first place.

Lastly, we would like to emphasize the difference of our approach from the method used in [12], though both achieve anonymous testing communications. The method proposed in [12] encrypts all the testing communications to prevent the malicious nodes from knowing the information carried in each testing packet. This is a computationally expensive protocol if the testing takes place in a large-scale network. Furthermore, if packet encryption is not widely used in the network, the encryption itself is actually alerting the malicious nodes that the encrypted packets are likely to be part of the testing communications. Thus a clever malicious node is able to avoid being detected by not touching the encrypted packets while dropping all the unencrypted packets. In our scheme, only the handshake messages and the acknowledgment message are encrypted. Though the encrypted packets might alert the malicious nodes, the

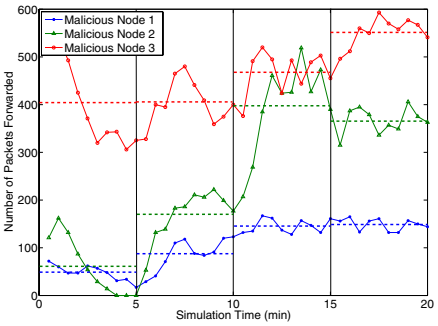


Fig. 4. The packet forwarding load on the three malicious nodes.

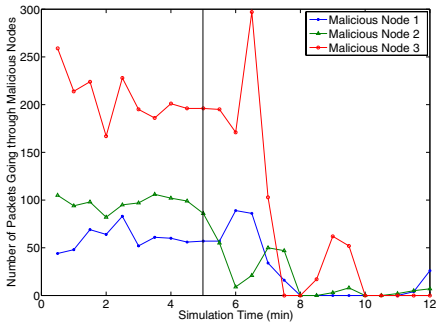


Fig. 5. The packet forwarding load on the three malicious nodes with misbehavior detection.

malicious nodes cannot acquire sufficient information to avoid being tested. Also as we have discussed, disrupting the test through dropping the encrypted messages is not wise for the attackers: that will only result in being detected by the testing nodes.

### *B. Performance Evaluation of the Protocol*

We evaluate the performance of the protocol in three metrics: 1) the cost we place on an attacker, 2) the detection effectiveness measured by the change of the traffic load forwarded to a malicious node before and after detection, 3) the percentage of the delivered packets that are malicious.

We have simulated the reciprocal routing and the misbehavior detection protocols using NS-2. When a node has a packet to forward out, it chooses the next hop node using the rate distribution probability $\frac{r_i}{\sum_{j=1}^{n_1} r_j}$. As the traffic changes dynamically, the rate distribution probabilities also change over time. In the misbehavior detection protocol, since we have proven that the testing packets are anonymous to the nodes under test, a malicious node randomly drops packets including both the usual traffic and the testing packets. The simulated network consists of 50 nodes, out of which 3 are malicious. These nodes are distributed randomly in a network area of 1000m$\times$1000m. Each node has a radio coverage of 250m.

In the first simulation, the malicious nodes send junk packets but do not drop others' packets. Each legitimate node generates new packets at a rate randomly distributed between 1 packet/second and 10 packet/second. The packets have random destinations. We set the 3 malicious nodes' packet generating rates to 0 packet/second initially, and increase their rates to 1 packet/second at the simulation time of 5 minutes, 5 packet/second at 10 minutes, 10 packet/second at 15 minutes. Simulation ends at 20 minutes. Fig. 4 shows the number of packets forwarded to the three malicious nodes in 30-second intervals. The average values in every 5-minute interval are marked in the figure. We see that the packet forwarding load imposed on the malicious nodes by their neighbors increases as the malicious nodes increase their flooding rates, which indicates the malicious nodes have to compensate in the form of forwarding the legitimate traffic. We note that the 3 malicious nodes have different load though they are generating the junk packets at the same rates. This is explained by their different geographical locations. A node close to the boundary of the network and/or having few neighbors tends to have less load than another node located in the center with many neighbors. The maximum load that can be placed on a malicious node is bound by the total legitimate traffic traversing the malicious node's neighbors.

In the second simulation, the 3 malicious nodes drop packets randomly with a 0.5 time-average dropping ratio. Each node periodically tests its neighbors when it sends traffic through these neighbors. A node tests one of its neighbors in every 15 seconds. If the testing node receives acknowledgment of no less than 70% testing packets, it believes the neighbor under test is normal; otherwise, it believes the neighbor to be malicious. In order to have a comparison with the situation without misbehavior detection, we let the nodes start testing at the simulation time of 5 minutes and their tests continue until the simulation ends at 12 minutes. Fig. 5 shows that the malicious nodes are successfully detected and their neighbors avoid forwarding traffic to them after the detection. In the first 5 minutes of the simulation, the normal nodes forward their packets to the malicious nodes as before. Then the detection is activated. As a node needs to test its neighbors one by one, there is a delay before the malicious nodes are detected by their neighbors, which is observed in the figure that the load on the malicious nodes drops to low values after the 8th minute. In order to minimize the false positive problem in which normal nodes are identified as misbehaving, our

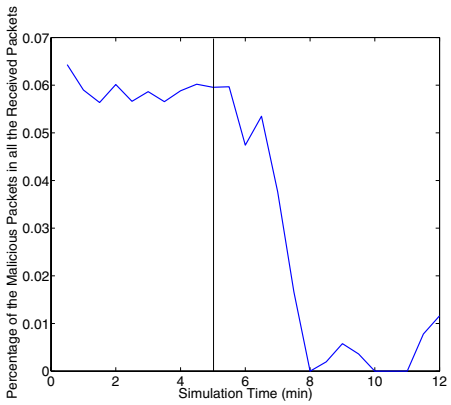


Fig. 6. The percentage of the delivered packets that are malicious.

protocol requires periodical misbehavior tests such that the misjudged nodes can be accepted back into the network. The periodical test sometimes also judges a malicious node as normal, if the malicious node happens to drop less than $30\%$ testing packets during a test. A few such cases are observed in the figure that the malicious nodes still receive a few packets from their neighbors sometimes after they are first detected. On average, however, most legitimate traffic is routed away from the malicious nodes. Fig. 6 shows that most of the attacking packets are successfully blocked after the malicious nodes are detected.

## V. Conclusions

In this paper, we have studied a new mitigation method for the DoS flooding and packet dropping attacks in wireless networks. The lack of satisfactory solutions to DoS attack prevention is partially due to the difficulty in distinguishing between the malicious and the normal traffic and nodes. In our method, we first achieve fairness of the routing cost among the nodes in a wireless network. The routing fairness objective places workload onto the attackers in proportion to their attacking strength while not penalizing the normal behaving nodes. In the second step of our protocol, we prevent the malicious nodes from dropping legitimate traffic through a misbehavior detection mechanism. The combined use of the reciprocal routing and the misbehavior detection protocols forces the DoS attackers to compensate for their attacks, or to be caught and isolated if they continue dropping other nodes' packets. We have proven the communication anonymity of our node misbehavior detection protocol, which is vital for the correct functioning of the protocol. Our simulation results have demonstrated its effectiveness in DoS attack mitigation.

## References

[1] D. Moore and C. Shannon. "Inferring Internet Denial-of-Service Activity". In *ACM Transactions on Computer Systems (TOCS)*, May 2006.

[2] CERT/CC. "Denial of Service Attacks". Available Online: http://www.cert.org/tech_tips/denial_of_service.html, 2001.

[3] Y.C. Hu, D.B. Johnson, and A. Perrig. "Secure Efficient Distance Vector Routing Protocol in Mobile wireless Ad Hoc Networks". In *Proceedings of the Fourth IEEE Workshop on Mobile Computing Systems and Applications (WMCSA 2002)*, June 2002.

[4] R. Mahajan, S. Bellovin, S. Floyd, J. Ioannidis, V. Paxson, and S. Shenker. "Controlling High Bandwidth Aggregates in the Network. ACM CCR, 32(3):62-73, 2002.

[5] X. Wang, M.K. Reiter. "Mitigating Bandwidth-Exhaustion Attacks using Congestion Puzzles. In *Proc. of ACM CCS*, pages 257–267, 2004.

[6] J. Li, J. Mirkovic, and M. Wang. "Save: Source Address Validity Enforcement Protocol". In *Proc. IEEE INFOCOM 2002*.

[7] S. Zhu, S. Setia, S. Jajodia, and P. Ning. "An Interleaved Hop-by-Hop Authentication Scheme for Filtering of Injected False Data in Sensor Networks". In *Proc. IEEE S&P 2004*.

[8] S. Marti, T.J. Giuli, K. Lai, and M. Baker. "Mitigating routing misbehavior in mobile ad hoc networks". In *Proc. 6th Annual International Conference on Mobile Computing and Networking*, pages 255–265, 2000.

[9] S. Buchegger and J.Y. Le Boudec. "Performance Analysis of the CONFIDANT Protocol (Cooperation Of Nodes - Fairness In Dynamic Ad-hoc Networks)". In *Proceedings of the Third ACM International Symposium on Mobile Ad Hoc Networking and Computing, (MobiHoc 2002)*, June 2002.

[10] Q. He, D. Wu, and P. Khosla. "SORI: a secure and objective reputation-based incentive scheme for ad-hoc networks". In *Wireless Communications and Networking Conference*, IEEE Volume 2, Page(s):825 - 830, March 2004.

[11] Mahajan, M. Rodrig, D. Wetherall, and J. Zahorjan. "Sustaining Cooperation in Multihop Wireless Networks". In *Proceedings of 2nd Symposium on Networked Systems Design and Implementation (NSDI)*, May 2005.

[12] M. Just, E. Kranakis, and T. Wan. "Resisting Malicious Packet Dropping in Wireless Ad Hoc Networks". In *Proc. of 2nd Annual Conference on Adhoc Networks and Wireless (ADHOCNOW '03)*, Springer Verlag, LNCS vol. 2856, pp. 151-163, Montreal, Canada, October 2003.

[13] B. Karp. *Geographic Routing for Wireless Networks*, Ph.D. Thesis, Harvard University, 2000.