Resisting Malicious Packet Dropping in Wireless Ad Hoc Networks

Mike Just[1] Evangelos Kranakis[2] * Tao Wan[2] **

[1] Treasury Board of Canada, Secretariat, 2745 Iris St., Ottawa, ON, K1A 0R5, Canada
[2] School of Computer Science, Carleton University, Ottawa, ON, K1S 5B6, Canada

Abstract. *Most of the routing protocols in wireless ad hoc networks, such as DSR, assume nodes are trustworthy and cooperative. This assumption renders wireless ad hoc networks vulnerable to various types of Denial of Service (DoS) attacks. We present a distributed probing technique to detect and mitigate one type of DoS attacks, namely malicious packet dropping, in wireless ad hoc networks. A malicious node can promise to forward packets but in fact fails to do so. In our distributed probing technique, every node in the network will probe the other nodes periodically to detect if any of them fail to perform the forwarding function. Subsequently, node state information can be utilized by the routing protocol to bypass those malicious nodes. Our experiments show that in a moderately changing network, the probing technique can detect most of the malicious nodes with a relatively low false positive rate. The packet delivery rate in the network can also be increased accordingly.*

Keywords: Security, Denial of Service (DoS), Wireless Ad Hoc Networks, Distributed Probing, Secure Routing Protocols

1 Introduction

A wireless or mobile ad hoc network (MANET) is formed by a group of wireless nodes which agree to forward packets for each other. One assumption made by most ad hoc routing protocols [16, 21] is that every node is trustworthy and cooperative. In other words, if a node claims it can reach another node by a certain path or distance, the claim is trusted. If a node reports a link break, the link will no longer be used. Although such an assumption can simplify the design and implementation of ad hoc routing protocols, it does make ad hoc networks vulnerable to various types of denial of service (DoS) attacks, which are discussed in detail in Section 2. One class of DoS attacks is malicious packet dropping. A malicious node can silently drop some or all of the data packets sent to it for further forwarding even when no congestion occurs.

Malicious packet dropping attack presents a new threat to wireless ad hoc networks since they lack physical protection and strong access control mechanism. An adversary

* Research supported in part by NSERC (Natural Sciences and Engineering Research Council of Canada) and MITACS (Mathematics of Information Technology and Complex Systems) grants.

** Research supported in part by OCIPEP (Office of Critical Infrastructure Protection and Emergency Preparedness) Research Fellowship.

can easily join the network or capture a mobile node and then starts to disrupt network communication by silently dropping packets. It is also a threat to the Internet since the various software vulnerabilities would allow attackers to gain remote control of routers on the Internet. If malicious packet dropping attack is used along with other attacking techniques, such as shorter distance fraud, it can create more powerful attacks (i.e., *black hole* [12]) which may completely disrupt network communication.

Current network protocols do not have the capability to detect the malicious packet dropping attack. Network congestion control mechanisms do not apply here since packets are not dropped due to congestion. Link layer acknowledgment, such as IEEE 802.11 MAC protocol [1], can detect link layer break, but cannot detect forwarding level break. Although upper layer acknowledgment, such as TCP ACK, allows for detecting end-to-end communication break, it can be inefficient and it does not indicate the node at which the communication breaks. Moreover such mechanism is not available in connectionless transport layer protocols, such as UDP. Therefore, it is important to develop mechanisms to render networks the robustness for resisting the malicious packet dropping attack.

In this paper, we present a proactive distributed *probing* technique to detect and mitigate the malicious packet dropping attack. In our approach, every node proactively monitors the forwarding behavior of other nodes. Suppose node A wants to know if node B performs its forwarding functions, it will send a probe message to a node one hop away from node B, let us say to node C. C is supposed to respond to the probe message by sending back an acknowledgment to A. If A can receive the acknowledgment within a certain time period, it acts as a confirmation that node B forwarded the probe message to C. With the assumption that a probe message is indistinguishable from a normal data packet, A knows that B will forward all the other packets.

Our experiments demonstrate that in a moderately changing network, the probing technique can detect most of the malicious nodes with a relatively low false positive rate. The packet delivery rate in the network can also be increased if the detected malicious nodes are bypassed from network communication. We argue that the probing technique is of practical significance since it can be implemented in the application layer and does not require the modification of underlying routing protocols.

The remainder of the paper is organized as follows. In Section 2, we analyze the DoS attacks against a network infrastructure and review the corresponding prevention mechanisms. In Section 3, we define frequently used notation and terminology. In Section 4, we present our solution for monitoring wireless ad hoc networks. In section 5, we describe the implementation and simulation of our solution. We conclude the paper in the last section.

2 DoS Attacks on Routing Infrastructure

Wireless ad hoc networks are vulnerable to various types of DoS attacks, such as signal injection, battery drain, among others. This paper focuses on the DoS attacks on its routing infrastructure. Based on the types of traffic transmitted in a network, we can classify these DoS attacks into two categories: *DoS attacks on routing traffic* and *DoS attacks on data traffic*. Such classification is also applicable to the Internet.

2.1 DoS Attacks on Routing Traffic

An attacker can launch DoS attacks against a network by disseminating false routing information so that established routes for data traffic transmission are undesirable or invalid. There are at least three possible consequences. *Firstly*, data traffic may be captured in a *black hole* [13] and never leave out. For example, in a distance vector routing protocol, an attacker can attract data traffic by advertising shorter distance and then drop the attracted traffic. *Secondly*, data traffic may not flow through routing paths fairly and some of them are dropped due to network congestion. For example, an attacker can avoid some traffic or redirect traffic to other nodes by advertising carefully crafted routing update messages. *Thirdly*, an attacker may disseminate arbitrary routing information to mislead other routers to create invalid paths in their routing table. As a result, data traffic flowing through those paths will eventually be dropped due to network unreachability or life time expiration (i.e., in presence of routing loops).

2.2 DoS Attacks on Data Traffic

An attacker can launch two types of DoS attacks on data traffic. *First*, it can inject a significant amount of data traffic into the network to clog the network. If there is no protection mechanism in place for provisioning data traffic, legitimate user packets will be dropped along with malicious ones as the result of congestion control. In the worst case, the network could be completely shutdown.

Second, if a malicious user manages to join a network or compromise a legitimate router, it can silently drop some or all of the data packets transmitted to it for further forwarding. We call it the *malicious packet dropping attack*. Malicious packet dropping attack is a serious threat to the routing infrastructure of both MANET and the Internet since it is easy to launch and difficult to detect. To launch the attack, an attacker needs to gain the control of at least one router in the target network. The router used to launch the attack can be a specialized router or a computer running routing software. To gain access to a specialized router, an attacker can explore the software vulnerability of a router (e.g., buffer overflow) or explore the weakness of logon authentication process (i.e., weak password). Many routers run vulnerable software and open the vulnerability to the world. For example, a survey [17] on 471 Internet routers shows that majority of them run SSH, Telnet or HTTP and 17% of them accept connections from arbitrary IP addresses. An attacker can also explore the vulnerabilities of routing protocols to join the network with his own computer or a compromised inside machine. This is possible due to the fact that most routing protocols only deploy very weak authentication mechanisms, such as plain text passwords.

2.3 Preventing DoS Attacks on Routing Traffic

Significant work has been done to secure routing protocols against DoS attacks on routing traffic. Most of them apply cryptographic techniques (asymmetric or symmetric) to authenticating routing traffic.

Asymmetric cryptographic techniques, such as public-key based digital signatures, can be used to sign routing messages [24–26] to prevent external intruders from joining the network or malicious insiders from spoofing or modifying routing messages. The disadvantages are: 1) They are quite inefficient since both the signature generation and

verification process involve the execution of computationally expensive functions. 2) They cannot prevent internal attacks.

Given the inefficiency of digital signature mechanisms, some researchers [7, 27] proposed to use symmetric cryptographic primitives (i.e., one-way hash chains, one-time signatures, authentication tree, etc.) for authenticating routing messages. Unfortunately, these approaches still do not prevent attacks from compromised internal routers. Hu, Johnson, and Perrig [13, 14] take the step further in securing distance vector routing protocols by forcing a node to increase metrics when forwarding routing update messages. Therefore, their approaches can prevent compromised nodes from claiming shorter distances. The disadvantage is that a malicious node can avoid traffic by claiming longer distances.

2.4 Preventing DoS Attacks on Data Traffic

It has been hypothesized that a network with QoS support can well resist DoS attacks since malicious packets will be dropped in the first place when facing network congestion. Other researchers proposed mechanisms [3, 6] to trace back to the origin of the malicious packets which cause the network congestion and drop them in the routers where they first enter into the victim network. *Ingress/Egress filtering* can also be helpful if IP spoofing is utilized in the attack.

Several approaches have been proposed to prevent DoS attacks on data forwarding level. Perlman [22] proposed hop-by-hop packet acknowledgment to detect packet dropping in a network. The disadvantage is that it will generate significantly high routing overhead. Cheung et al [8] proposed a probing method for defeating denial of service attacks in a fixed routing infrastructure using neighborhood probing. It requires a testing router to have a private address which allows it to generate a packet destined to itself but goes through the tested router. This requirement is not practical in MANETs. A distributed monitoring approach is proposed in [4] for detecting disruptive routers. The protocol is based on the principle that any packets sent to a router and not destined to it are supposed to leave that router. This principal is not applicable to MANET due to their changing network topology.

Marti et al [19] proposed and implemented two protocols for detecting and mitigating misbehaving nodes in wireless ad hoc networks by *overhearing* neighborhood transmissions. Their method is very effective for detecting misbehaviors in one-hop away. To monitor the behavior of nodes two or more hops away, one node has to trust and rely on the information from other nodes, which introduces the vulnerability that good nodes may be bypassed by malicious or incorrect accusation.

Buchegger and Le Boudec [5] developed the CONFIDANT protocol for encouraging node cooperation in dynamic ad-hoc networks. Each node monitors the behavior and maintains the reputation of its neighbors. The reputation information may be shared among friends. A trust management approach similar to Pretty GOOD Privacy (PGP) is used to validate received reputation information. Nodes with bad reputation may be isolated from the network. As a result, nodes are forced to be cooperative for their own interest. Our proposed probing technique can be used as one of the monitoring techniques in the CONFIDANT protocol.

Awerbuch et al [2] proposed a secure routing protocol for resisting byzantine failures in a wireless ad hoc network. The protocol requires an ultimate destination to send

an acknowledgment back to the sender for each of its successfully received packets. If the loss rate of acknowledgment packets exceeds the predefined threshold, which is set to be slightly above the normal packet loss rate, the route used for sending packets from the source to the destination is detected as faulty and a binary search probing technique is deployed to locate the faulty link. The disadvantages of this protocol are: 1) it may incur significant routing overhead; 2) a data packet with an inserted probe list can be distinguished from those without probe lists, although the probe list is onion encrypted and cannot be tampered en route. Our proposed probing technique differs in that it can be implemented above the network layer (e.g., based on UDP), and the end-to-end encryption of IP payload using pair-wise shared keys can prevent intermediate nodes from distinguishing probe messages from data packets.

Padmanabhan and Simon [20] proposed a secure traceroute to locate faulty routers in wired networks. In their approach, end hosts will monitor network performance. If an end-to-end performance degrade is detected by a host to a destination, a *complaint* bit is set in all the subsequent traffic to that destination. The complaining host itself or the router sitting closest to the complaining host may start the troubleshooting process if it observes enough complaints. It first sends a secure traceroute packet to the next hop, which can be derived from its routing table. The router receiving the secure traceroute packet is expected to send a response back which also includes a next hop address. This process repeats until a faulty router is located (no response received from it) or every router on the path to the ultimate destination proves healthy. Our approach is different from the secure traceroute in that 1) our approach is proposed for MANET using source routing protocols (e.g., DSR), the secure traceroute is mainly used in wired networks. 2) our approach does not require modification to existing routing infrastructures, the secure traceroute may need to modify IP layer in order to monitor performance problem; 3) our approach utilizes redundant routing information for diagnosis, the secure traceroute does not.

Malicious nodes silently dropping packets exhibit the same behavior as selfish nodes, which may choose to drop packets for the sake of saving its own constraint resources, such as battery or CPU cycle. Selfishness and its threat to the network performance have been well studied by Roughgarden [23]. Incentive mechanisms have been proposed to encourage selfish nodes to be cooperative and to forward packets for others. Unfortunately, incentive mechanisms don't work for malicious users since they never play by rules. Our proposed probing scheme can be used to detect and mitigate selfishness problem.

3 Definitions and Assumptions

3.1 Node States

We classify the states of a node as follows. A node is *GOOD* if it responds to probe messages for itself and forwards other probe messages along their source routes. A node is *BAD* if it responds to probe messages destined to itself but fails in forwarding probe messages for others. A benign link failure may also be detected as *BAD* behavior if it is not cleared by other mechanisms (e.g., route error in DSR). A node is considered *DOWN* if 1) it is a neighbor node to the probing node and it doesn't respond to probe messages; or 2) it is not a neighbor node and it doesn't respond to probe messages

through *all* the known paths. A node is considered at the *UNKNOWN* state if on all known paths from the probing node to the node, there exists at least one node in BAD or DOWN state.

3.2 Assumptions

Probe messages are indistinguishable from normal packets. One limitation of the probing technique is that it can be easily defeated if probe messages can be distinguished from normal data packets. For example, a malicious node may forward probe messages, but drop all the other data packets, thereby avoiding detection. This assumption can be realized using end-to-end encryption of IP payload by pair-wise shared keys. Since a malicious node can understand only the IP header, it does not have the information of upper layer protocols, such as TCP/UDP port numbers. By implementing the probing technique above the network layer (e.g., based UDP), an adversary will not be able to distinguish a probe message from a other data packet (e.g., HTTP or SMTP packet). Some other options are: 1) piggybacking a probe message on a normal data packet which requires acknowledgment, such as TCP SYN. The disadvantage is that such data packets may not be available during the time of probing. 2) assuming that an adversary cannot modify the forwarding software of the compromised router. Therefore, the adversary can only make decisions based on IP addresses, which does not allow for distinguishing a probe message from a normal data packet.

Multi-hop source routing protocols. The probing technique assumes a multi-hop source routing protocol since a probing node needs to specify the source route by which a probe message takes to get to the destination. This assumption is practical since some routing protocols, such as Dynamical Source Routing (DSR) [16], are multi-hop source routing protocols.

Bi-directional communication links. We assume that all communication links are bi-directional. This assumption is practical in some wireless networks, such as IEEE 802.11 [1], where all links have to be bi-directional for link layer acknowledgment to work.

4 The Distributed Probing Scheme

In order to monitor the behavior of mobile nodes by the probing technique, we need to decide which node should probe and how far it should probe. Given a network with n nodes, there are several interesting possibilities: 1) there is only one probing node and it probes all the other nodes; 2) there are k probing nodes ($1 < k < n$) and each probes over a distance of r ($1 < r < \infty$); 3) there are n probing nodes and each probes over a distance of infinity. The last approach is preferred in a MANET since it can detect selective packet dropping (i.e., based on IP addresses). Another advantage is that one node does not need to rely upon the information from other nodes to detect malicious ones. The disadvantage is that it may generate significant network overhead. The network overhead can be reduced if probe messages piggyback normal packets. To simplify the problem, we divided the probing technique into three algorithms: 1) the probing path selection algorithm; 2) the probing algorithm; and 3) the diagnosis algorithm. These are described below.

4.1 Probing Path Selection Algorithm

The probing paths are selected solely from the routing cache maintained by a mobile node. There are usually many redundant paths in the routing cache. Although probing over each of them may allow for validating all the known paths, it will also produce significant network overhead. The ideal strategy shall select a minimum number of paths but allows for monitoring the forwarding behavior of as many nodes in the routing cache as possible. The probing path selection algorithm returns a set of paths with the following properties.

1) For any two paths p_i and p_j, $p_i \nsubseteq p_j$. Since probing over a path can always disclose the forwarding behavior of those nodes in any of its subsets, any path which is a subset of another path will be eliminated.

2) For any two paths p_i and p_j, if the second farthest node in p_i is an intermediate node of p_j, the farthest node of p_i will be removed. For example, given two paths $p_1 = A \rightarrow B \rightarrow C \rightarrow D$ and $p_2 = A \rightarrow E \rightarrow F \rightarrow C \rightarrow G \rightarrow H$, node D will be removed from p_1. With D in p_1, A can monitor the forwarding function of C. Since such monitoring can be achieved by probing over p_2, there is no need to keep D in p_1. C will still be kept in p_1, since A needs to monitor B by sending a probe message to C.

3) The length of any path (in terms of number of hops) is greater than 1. Since we are interested in monitoring the forwarding function of mobile nodes, probing over a one hop path offers no information. A probe message is sent to a neighbor node only when a node subsequent to it doesn't respond to the probe message and the probing node needs to know if the neighbor node is BAD or has moved out of its transmission range.

4.2 The Probing Algorithm

With a set of selected probing paths, the probing algorithm will probe over each of them. Given a probing path, there are at least two ways of probing. One way is to probe from the farthest node to the nearest. The other way is to probe from the nearest node to the farthest. Each has its own advantages and disadvantages. Probing from far to near is better if the probing path is *GOOD* since it takes only one probe message and proves the goodness of all the intermediate nodes. But it may take more probe messages if a *BAD* node is located near the probing node. This method can be applied to a network where we have the confidence that the majority of the nodes in the network are *GOOD*. The advantage of probing from near to far is that it generates smaller number of probing messages to detect a *BAD* node located near the probing node. Another advantage is that we have the prior knowledge of the states of all the intermediate nodes along the path to the probed node except its immediate predecessor node. The disadvantage is the an intelligent attacker may be able to avoid detection by forwarding all packets (including probe messages destined to the downstream nodes) for a certain period of time immediately after receiving a probe message for itself. A received probe message therefore serves as a signature to an attacker that a diagnosis process is ongoing, and it would start to behave normally for a short period of time. Other search strategy (e.g., binary search) can also be deployed to reduce network overhead.

In this paper, we present the algorithm for the first method, probing from the farthest nodes to the nearest, since it is stronger than the other alternatives in detecting

malicious nodes. For a probing path, the probing node sends a probe message to the farthest node. If an acknowledgment message is received within a certain period of time, all the intermediate nodes are shown to be GOOD. Otherwise, a probe message is sent to the second farthest node. This process is repeated until one node responds to the probe message or the nearest node (a neighbor node) is probed and it is not responsive. In the latter case, we know that the neighbor node in the probed path either is *DOWN* or has moved out to another location. Since the neighbor node is not responsive, there is nothing we can do to monitor the rest nodes in the path. Therefore, probing over this path is stopped. If an intermediate node is responsive but a node subsequent to it is not, it is possible: 1) the intermediate node failed forwarding the probe message to the next node; 2) the link between the two nodes is broken by location change; 3) the unresponsive node is incapable of responding to the probe message. The diagnosis algorithm will then be called to decide which one is the case.

4.3 The Diagnosis Algorithm

After the probing node detects a node (v_i) is responsive but the subsequent node (v_{i+1}) is unresponsive, it calls the diagnosis algorithm to determine if the link $v_i \leftrightarrow v_{i+1}$ is broken at the link level or forwarding level.

The probing node first searches the routing cache for another path to v_{i+1}. If such a path exists, it will probe v_{i+1} through this path. If v_{i+1} is still unresponsive, it searches the routing cache for another path. This process repeats until 1) there is a route (p) through which node v_{i+1} is responsive, or 2) the routing cache is exhausted.

In case 1, the diagnosis algorithm appends v_i to the path p and sends a probe message to v_i over p. If an acknowledgment is received from v_i, v_i is diagnosed as *BAD* since the link $v_{i+1} \rightarrow v_i$ is good but link $v_i \rightarrow v_{i+1}$ is not. Based on the assumption that any link is bidirectional, $v_i \leftrightarrow v_{i+1}$ should be good at link level. Therefore, it is broken at forwarding level. If v_i is unresponsive over the new path, the link $v_i \leftrightarrow v_{i+1}$ is diagnosed as broken in link layer. It is also possible that both v_i and v_{i+1} are *BAD*. Since there is no sufficient information available to distinguish this situation from the link layer break, we treat this situation as link layer break. It causes false negatives.

In case 2, node v_{i+1} may have moved out from its previous location and a new path to v_{i+1} is not discovered by the probing node yet. It is also possible that node v_{i+1} has moved out from the network or is *DOWN*. Although a route discovery may be able to disclose further information, it is also very expensive. Therefore, the diagnosis algorithm simply treats node v as being *DOWN*.

When a node is detected as *BAD*, the routing cache is updated by removing all nodes subsequent to the bad node. When a link is detected as broken, the routing cache is also informed and the link is truncated from all the paths. When the routing cache adds a route to the cache, it looks up the node state table and truncates the route accordingly if there is any BAD node in the path.

5 Simulations

We study the detection rate of the probing technique and and its impact on network performance using the *NS-2* network simulator [18] with the wireless extension from

Rice University. The simulation is performed on Sun Ultra 10 workstations running Solaris 5.7.

5.1 Simulation Environment

We implemented the probing technique in *NS-2* version 2.1b9a with wireless extension. The routing protocol we use is Dynamic Source Routing (DSR) and the routing cache is path cache with a primary and a secondary FIFO cache [11]. The probing technique is implemented as a part of DSR and the probe message is a new type of DSR packet.

We simulate a network with 670m x 670m space and 50 mobile nodes. The simulation time is 100 seconds. The mobile nodes move within the network space according to the *random waypoint mobility model* [15] with a maximum speed of 20m/s. The pause time is 50 seconds, which represents a network with moderately changing topology. The communication patterns we use are 10 constant bit rate (CBR) connections with a data rate of 4 packets per second. Those simulation parameters are widely used by the community. We chose them to make our simulation results comparable with others.

We randomly choose 0, 3, 5, 8, 10, 13, and 15 BAD nodes in each of the simulation. Security researchers like to assume the worst case, but it rarely happens in real life. Since it is realistic that the majority of nodes in a network should be GOOD, we simulate at most 15 BAD nodes, which represent 30 percent of the total number of nodes.

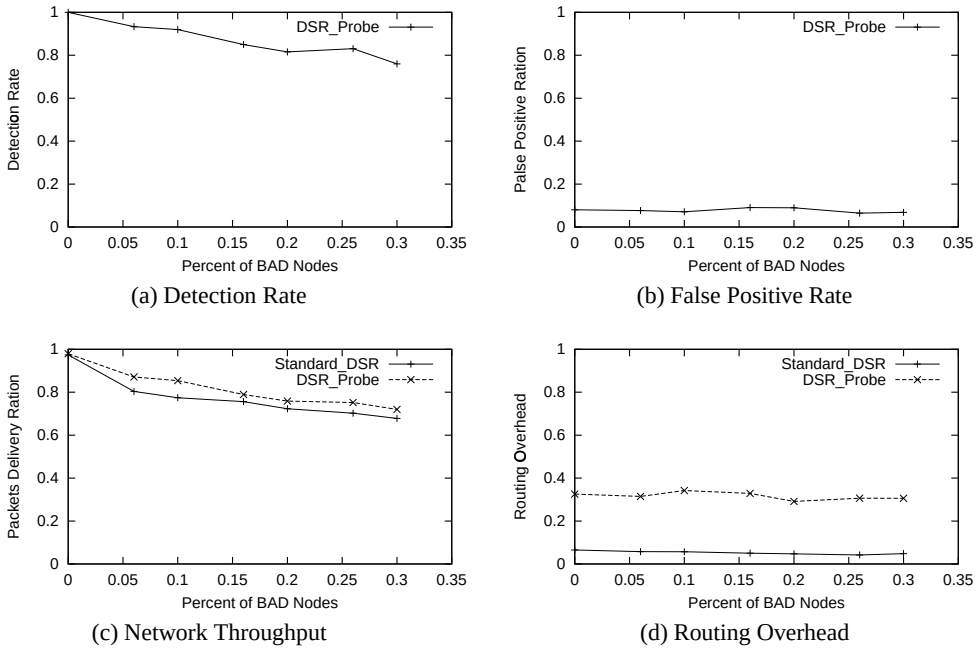
5.2 Metrics

We chose the following metrics for measuring the probing technique: 1) *Detection Rate*, the ratio of the number of detected BAD nodes and the total number of actual BAD nodes. 2) *False Positive Rate*, the ratio of number of GOOD nodes mistakenly detected as BAD and the total number of GOOD nodes. The combination of this metric and the detection rate tells us the overall performance of the probing technique. 3) *Packet Delivery Rate*, the ratio of total number of data packets received and the total number of data packets sent in application level. In our simulation, the data packets refer to the CBR traffic. 4) *Network Overhead*, the ratio of total number of routing related transmissions (including all DSR related traffic and probe messages) and the total number of packet transmissions (including both routing related transmissions and data transmissions). Each packet hop is counted as one transmission.

5.3 Simulation Results

We study the probing technique using the above defined metrics. The standard DSR (Standard_DSR) is used as a baseline to compare with the DSR with the extension of the probing technique (DSR_Probe). We run the simulation 5 times and all the graphs (Figure 1) are plotted from the data averaged from the 5 runs.

Detection Rate Figure 1.a shows the detection rate. In the best case, 94% of the bad nodes can be detected. In the worst case, the detect rate is 76%. There are several reasons why a BAD node is not detected. First, the BAD node is not in any path in the routing cache each time when the probing technique starts to probe. Since the probing paths are selected solely based on the paths maintained by the routing cache, if a node is



(a) Detection Rate (b) False Positive Rate

(c) Network Throughput (d) Routing Overhead

Fig. 1. Simulation Results

not contained in any path, its forwarding function will not be monitored. Second, there are two consecutive BAD nodes in a path, and the bad behavior of one is hidden by the other. The link between the two bad nodes is detected as link layer break, the the bad behavior is not detected. Although this affects the detection rate, it does not have impact on packet delivery rate since the link is removed from the routing cache in any way.

False Positive Rate Figure 1.b shows the false positive rate. We can see from the graph that the highest false positive rate is below 9%, which is relatively low. The false positive is caused mainly by node movement since some link layer breaks are detected as forwarding level misbehavior. Therefore, it will decrease when the node motion becomes slower.

Packet Delivery Rate The graph of packet delivery rate (Figure 1.c) has two curves and they represent the throughput of standard DSR and the DSR with the extension of the probing technique. The graph demonstrates that the DSR with the probing technique extension always performs better than the standard DSR. This is in line with our expectation since the bad nodes which failed in forwarding packets are removed from the routing cache. The result is that good paths are used for transmitting packets.

We can also see from the graphs that packet delivery rate sometimes is higher when there is a higher percentage of BAD nodes than when there is a lower percentage of

BAD nodes. This is contrary to the common expectation. As explained in [19], the randomness of NS-2 results in this effect due to the fact that route replies may arrive at nodes in different orders in different runs. Therefore, a node may choose a path with BAD nodes in one run but choose a good path in another run.

Overhead As shown in Figure 1.d, the routing overhead is increased significantly when the network topology changes faster or there is a high percentage of BAD nodes in the network. In both scenarios, a large number of probe messages have to be sent out to finalize the node states. The overhead can be reduced dramatically if probing messages piggyback normal data packets.

6 Conclusion

Wireless ad hoc networks are vulnerable to various types of DoS attacks. We presented a distributed probing technique to detect and migitate the malicious packet dropping attack in MANETs. We implemented the probing technique in NS-2 with wireless extensions. Our experiments show that in a moderately changing network, the probing technique can detect most of the malicious nodes with a relative low false positive rate. The packet delivery rate can also be increased if the node state information is shared with routing cache. We think the probing technique is of practical significance since it can be implemented independently from routing software and does not require modification to the existing infrastructure. The disadvantage of the probing technique is that it generates relatively high network routing overhead if probe messages do not piggyback data packets.

References

[1] ANSI/IEEE std 802.11. Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specification, 1999.

[2] B. Awerbuch, D. Holmer, C. Nita-Rotaru, and H. Rubens. An On-Demand Secure Routing Protocol Resilient to Byzantine Failures. In *ACM Workshop on Wireless Security* (WiSe), September 2002.

[3] S.M. Bellovin, M. Leech, and T. Taylor. ICMP Traceback Messages. Internet draft: draft-ietf-itrace-03.txt, January 2003.

[4] K.A. Bradley, S. Cheung, N. Puketza, B. Mukherjee, and R.A. Olsson. Detecting Disruptive Routers: A Distributed Network Monitoring Approach. In *Proceedings of the IEEE Symposium on Research in Security and Privacy*, pages 115-124, May 1998.

[5] 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.

[6] H. Burch and H. Cheswich. Tracing anonymous packets to their approximate source. In *Proceedings of USENIX LISA*. pages 319-327, New Orleans, LA, December 2002.

[7] S. Cheung. An Efficient Message Authentication Scheme for Link State Routing. In *Proceedings of the 13th Annual Computer Security Applications Conference*, San Diego, California, USA, December 1997.

[8] S. Cheung and K. Levitt. Protecting routing infrastructure from denial of service using cooperative intrusion detection. In *Proceedings of New Security Paradigms Workshop*, Great Langdale, Cumbria, UK, September 1997.

[9] B.P. Crow, I.K. Widjaja, G. Jeong, and P.T. Sakai. IEEE 802.11 Wireless Local Area Networks. *IEEE Communications Magazine*, vol. 35, No. 9: pages 116-126, September 1997.

[10] A. Habib, M. Hefeeda, and B. Bhargava. Detecting Service Violations and DoS Attacks. In *Proceedings of 2003 Internet Society Symposium on Network and Distributed System Security (NDSS'03)*, San Diego, California, USA. February 2003.

[11] Y.C. Hu and D.B. Johnson. Caching Strategies in On-Demand Routing Protocols for Wireless Ad Hoc Networks. In *Proceedings of the Sixth Annual IEEE/ACM International Conference on Mobile Computing and Networking (MobiCom 2000)*, pages 231-242, August 2000.

[12] Y.C. Hu, A. Perrig, and D.B. Johnson. Ariadne: A Secure On-Demand Routing Protocol for Ad Hoc Networks. In *Proceedings of the Eighth ACM International Conference on Mobile Computing and Networking (MobiCom 2002)*, September 23-28, 2002.

[13] 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.

[14] Y.C. Hu, A. Perrig, and D.B. Johnson. Efficient Security Mechanisms for Routing Protocols. In *Proceedings of 2003 Internet Society Symposium on Network and Distributed System Security (NDSS'03)*, San Diego, California, USA. February 2003.

[15] D. Johnson and D.A. Maltz. Dynamic Source Routing in Ad Hoc Wireless Networks. In *Mobile Computing*, chapter 5, pages 153-181. Kluwer Academic Publishers, 1996.

[16] D. Johnson, D.A. Maltz, Y.C. Hu, and J.G. Jetcheva. The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks (Internet-Draft). Mobile Ad-hoc Network (MANET) Working Group, IETF, February 2002.

[17] G.M. Jones. The Case for Network Infrastructure Security. ;logon: The Magazine of USENIX and SAGE. pages 25-29, Volume 27, Number 6, December 2002.

[18] K. Fall and K. Varadhan, editors. The *ns* Manual (formerly *ns* Notes and Documentation). April 14, 2002. http://www.isi.edu/nsnam/ns/doc/index.html

[19] S. Marti, T.J. Giuli, K. Lai, and M. Baker. Mitigating Routing Misbehavior in Mobile Ad Hoc Networks. In *Proceedings of the Sixth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM 2000)*, August 2000.

[20] V.N. Padmanabhan and D. R. Simon. Secure Traceroute to Detect Faulty or Malicious Routing. In *ACM SIGCOMM Workshop on Hot Topic in Networks* (HotNets-I), October 2002.

[21] C.E. Perkins, E. M. Royer, and S.R. Das. Ad Hoc On Demand Distance Vector (AODV) Routing (Internet-Draft). June 2002.

[22] R. Perlman. Network Layer Protocols with Byzantine Robustness. PhD thesis, Massachusetts Institute of Technology, August 1988.

[23] T. Roughgarden. Selfish Routing. PhD thesis, Cornell University, May 2002.

[24] B.R. Smith and J.J. Garcia-Luna-Aceves. Securing the Border Gateway Routing Protocol. In *Proceedings of Global Internet 1996.* London, UK. November 1996.

[25] B.R. Smith, S. Murthy, and J.J. Garcia-Luna-Aceves. Securing Distance-Vector Routing Protocols. In *Proceedings of 1997 Internet Society Symposium on Network and Distributed System Security (NDSS'97)*, San Diego, California, USA. February 1997.

[26] M.G. Zapata and N. Asokan. Securing Ad Hoc Routing Protocols. In *Proceedings of the ACM Workshop on Wireless Security (WiSe 2002)*, September 2002.

[27] Kan Zhang. Efficient Protocols for Signing Routing Messages. In *Proceedings of 1997 Internet Society Symposium on Network and Distributed System Security (NDSS'98)*, San Diego, California, USA, March 1998.