Evaluation of a Stastical Technique to Mitigate Malicious Control Packets in Ad Hoc Networks

Rajendra V. Boppana
Department of Computer Science
University of Texas, San Antonio, Texas
boppana@cs.utsa.edu

Saman Desilva
Department of Mathematics
St. Philips College, San Antonio, Texas
desilva.saman@gmail.com

Abstract. The on demand routing protocols for mobile ad hoc networks use network-wide broadcasts of control packets to learn routes. This feature can be exploited by malicious nodes to launch highly leveraged denial of service attacks in ad hoc networks. We use an adaptive statistical packet dropping mechanism to mitigate such attacks and their impact on throughput. The proposed mechanism does not use any additional network bandwidth. Using experiments on a wireless testbed, we evaluate the effectiveness of the statistical technique for UDP and TCP traffic.

I. Introduction

The communication protocols for mobile ad hoc networks (MANETs) are designed to work in peer-to-peer networking mode. To facilitate communication between nodes beyond each other's radio range, the other nodes in the network act as routers. Because of node mobility, network topology and hence the routes change frequently. So designing routing protocols for ad hoc networks is a challenging problem.

The security issues regarding the data are the same on both wired networks and wireless networks. Secure sockets layer and end-to-end encryption mechanisms are used to address the same. In addition, MANETs are susceptible to attacks on the routing protocol function itself since all or most nodes in the network participate in route discovery and dissemination. These attacks can be classified into (a) resource consuming and (b) route falsifying and dropping/delaying data packets. There has been a substantial amount of work done to address the latter problem [8], [12], [7], [21], [1]. Though both attacks are considered denial of service (DoS) attacks from an application or end user perspective, we use DoS to refer to the former attacks, which are investigated in this paper. Intrusion detection systems [22] address DoS as part of a variety of other security attacks, but they are rather complex programs and often depend on dissemination to identify malicious behavior. Owing to the use of flooding for route discovery by many on demand routing protocols, highly leveraged DoS attacks can be launched by malicious nodes without generating unusually high traffic. This makes the detection and mitigation of such attacks nontrivial.

We have shown in an earlier work using simulations that a malicious node flooding the MANET with control packets related to bogus route discoveries can cause a sharp drop in network throughput [6]. These malicious nodes behave like the normal nodes in all aspects except that they initiate frequent control packet floods. This type of attack is hard to detect since any normal node with frequently broken routes could legitimately initiate frequent route discoveries. Therefore, to mitigate bogus control packet floods, we have proposed a simple rate-based control packet forwarding mechanism and have shown using simulations that it works well [6].

In this paper, we demonstrate the impact of control packet flood attacks on an eight-node testbed consisting of Linksys 54G wireless routers. We reprogrammed these routers with Linux kernel and freely available AODV software. We implemented the statistical profiling and rate controlling mechanism on this testbed. Using constant bit rate (CBR) traffic over UDP and FTP traffic over TCP, we show that the rate control mechanism is very effective in detecting and mitigating the attack. An attacker is identified by other normal nodes in 30-60 seconds, and the network throughput is sustained under attack.

The rest of the paper is organized as follows. Section II describes the route discovery mechanism used in on demand routing protocols and a highly effective DoS attack on them. Section III describes a statistical rate control mechanism to mitigate the attack. Section IV describes the experimental testbed used. Section V presents experimental evaluation of the attack and the solution. Section VI concludes the paper.

II. Background

Routing Protocols for Ad Hoc Networks

MANET Routing protocols can be divided into proactive and reactive (or on demand) categories [15], [9], [2], [4], [13] were proposed for MANETs. Both proactive and reactive protocols can suffer from control packet floods caused by malicious nodes. In this paper, we investigate DoS attacks on reactive protocols using the AODV on demand routing protocol as an example.

Route discovery in on demand protocols: On demand routing protocols learn only needed routes and do not refresh them periodically. When a node attempts to send a data packet to a destination for which it does not already know the route, it uses a "route discovery" process to dynamically obtain a route. The route discovery works by flooding the network with route request (RREQ) control packets. A node, say, x, receiving

This research has been partially supported by NSF grant EIA-0117255 and AIA grant F30602-02-1-0001.

Proceedings of the 2006 International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM'06)
0-7695-2593-8/06 $20.00 © 2006 IEEE

Authorized licensed use limited to: IEEE Xplore. Downloaded on December 4, 2008 at 17:18 from IEEE Xplore. Restrictions apply.

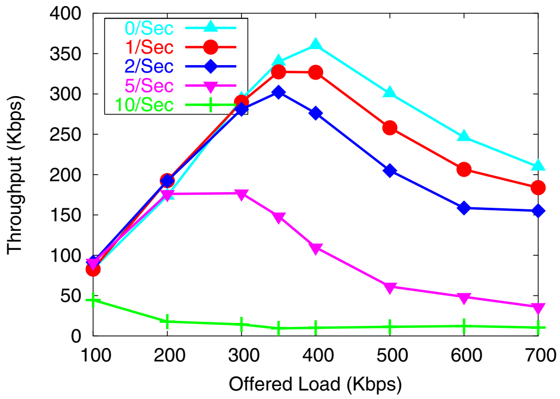

Fig. 1. Loss of throughput with bogus route discoveries by a malicious node. The route discoveries are initiated at the rate of 1,2,5 or 10/Second.

a RREQ, rebroadcasts it, unless it has already seen it from another neighbor or it has a route to the destination indicated in the RREQ. If the received RREQ is a duplicate, node x drops it. If node x has the route from another route discovery or because it is the destination, then it replies to the RREQ with a route reply (RREP) packet that is routed back to the original sender of the RREQ.

A drawback of flooding based route discovery is the high control overhead. Each RREQ initiated by a node results in up to n broadcasts in the MANET, where n is the number of nodes in the MANET. So at high loads, the wireless channel usage can be completely dominated by the control packets used for route discoveries [5]. This potential weakness of on-demand routing protocols could be exploited by malicious nodes.

DoS Attack on AODV

To evaluate the impact of bogus control packet floods by malicious nodes, we simulated a 100-node MANET with AODV routing protocol and CBR traffic over UDP [6]. One of the nodes (that is neither a sender or receiver of CBR data) was a malicious node flooding the network with bogus route discoveries at a rate of 1 to 10 RREQs/s. (Normal network performance is obtained when the specified attack rate is zero RREQs/s.) The malicious node drops any route information received in response to its route discoveries and continues to initiate route discoveries at the specified rate. This node behaves like any other node in the network in all aspects except that it sends frequently RREQ packets, which are used for route discovery. Figure 1 shows achieved throughput as a function of offered load and malicious node's route discovery rate. For traffic loads at or beyond saturation, any RREQ rate by the malicious node reduces the throughput rapidly. At 10 RREQs/second, the peak throughput is reduced by 84%.

This type of attack is hard to detect since any normal node with a broken route could legitimately initiate multiple RREQ broadcasts in a short period of time. The security enhancements such as those used for secure AODV [21] do not handle this type of attack since the malicious node is not forging any information. Broadcast management techniques [19], which minimize the number of transmissions used to achieve network-wide broadcasting, are not effective in mitigating this attack [6]. A static limit [20] on RREQs generated by a node can hurt the performance by restricting the route discovery capability of genuine nodes if the limit is too low. A high static limit is not effective.

III. Statistical Rate Control

In this section, we describe a distributed statistical profiling technique to detect misbehaving nodes and mitigate their impact on performance. We assume that all RREQs are authenticated. So every node must include its ID and authentication information, which we assume cannot be forged. So malicious nodes are at one time trusted nodes that have the appropriate authentication, but attack the network when the opportunity arises.

In our design, each node monitors the route requests it receives. Each node maintains a count of RREQs received for each RREQ sender during a preset time period (τ). At the end of the time period, the node computes the rate at which it has been receiving route requests from each sender and smoothed average, $savg$, of the same using (1) and (2). The node also computes average rate of RREQs per sender using (3) and smoothed average, $nodeavg$, of the same using (4). In addition, the node also computes the savg deviation of all RREQ sources at the end of the time period using Equation 5 repeatedly for each RREQ sender. This is denoted as $nodedev$.

$$\begin{aligned} rate_i &= RREQCount_i/\tau && (1)\\ delta_i &= rate_i - savg_i && \\ savg_i &\leftarrow savg_i + g \times delta_i && (2)\\ noderate &= \frac{TotalRREQCount/\tau}{\#ofRREQsenders} && (3)\\ delta &= noderate - nodeavg && \\ nodeavg &\leftarrow nodeavg + g \times delta && (4)\\ nodedev &\leftarrow nodedev + h(|delta_i| - nodedev) && (5) \end{aligned}$$

The $nodeavg$ and $nodedev$ calculations are based on the TCP retransmission timeout (RTO) calculations [18]. As a starting point, the value of g is set as $\frac{1}{8}$ and h is chosen to be $\frac{1}{4}$. We have experimented g values of $\frac{1}{4}$, $\frac{1}{2}$ and $\frac{1}{8}$ and found $\frac{1}{8}$ the best value for our network conditions.

To distinguish between malicious RREQ floods and those by normal nodes, we calculate a cut-off rate (denoted, *CutOffRate*) as given in (6). The RREQs from a sender whose smoothed average rate is above the *CutOffRate* will be dropped without forwarding. Dropped RREQs are counted in computing individual nodes' smoothed averages, however.

$$CutOffRate = nodeavg + 2 \times nodedev \qquad (6)$$

This technique is shown to work well for simulated networks [6]. However, simulations do not provide realistic models of noise, which can impact the network behavior in an unpredictable manner and render the solution technique ineffective.

Authorized licensed use limited to: IEEE Xplore. Downloaded on December 4, 2008 at 17:18 from IEEE Xplore. Restrictions apply.

IV. Experimental Environment

In this section, we describe the hardware, software, and operational issues regarding the testbed we set up and used for experimental evaluation of RREQ flood attacks.

Hardware

We considered two hardware platforms. One is a small form factor desktop with an 802.11g wireless network interface. The other is an off-the-shelf wireless access point with an Ethernet switch. (We did not consider laptops and PDAs since they cannot be left unsupervised in the laboratories.) We chose Linksys wrt54g routers [3] for the following reasons. The wrt54g routers are self-contained Linux-friendly light-weight boxes with small form factor. They can be easily be mounted on a wall or placed on top a desktop monitor. The wrt54g routers have a built-in 4-port 100 Mbps Ethernet switch, an 802.11g access point, 12 MB of usable RAM and 4 MB of flash memory, which serves as the disk memory [17]. The processor is a 200 MHz MIPS-compatible processor powerful enough to run various light-weight programs simultaneously. To improve signal reception, one can use extended antennas, which provide a gain of +7 dBi, from Linksys.

Software

We used the openwrt distribution of Linux kernel 2.4.29 [14]. The openwrt software comes with a packaging tool to install additional tools as needed and a crosscompiler to compile our own C programs and kernel modifications. Furthermore, an AODV ad hoc routing software package [10], which facilitates setting up an ad hoc network, is available for the openwrt software. Each of the boxes is reprogrammed with Linux OS, and AODV is added as a loadable module.

We modified the AODV module to capture additional statistics and write them to the kernel log. We developed a simple program to read AODV messages from the kernel log using the *dmesg* command and send them as UDP packets via the Ethernet to a specified desktop machine for analysis. We also developed a UDP load generator that can send packets from one node to another at a specified rate. The receiver side automatically sends throughput statistics via the Ethernet to a specified desktop machine.

Setup

We set up an 8-node ad hoc network spanning various rooms and laboratories in the department; see Figure 2. With default transmission power settings and multiple transmission rates (lower transmission rates have longer radio range), it is a nontrivial exercise to set up the network for multiple hops within in the space available. Instead, we placed the routers at reasonable distances and convenient places so that they are accessible for manual power cycle. To create multiple hops, we reduced the transmission power levels using the *wl* program that comes with the router.

To examine and gather data without disturbing the wireless network, we set up Ethernet connections to each of the routers. The Ethernet connections are used to start and stop experiments, examine router status, and gather data on-the-fly from a desktop machine. This setup facilitated a rapid develop-test-debug cycle when we were experimenting with various software options and AODV modifications.

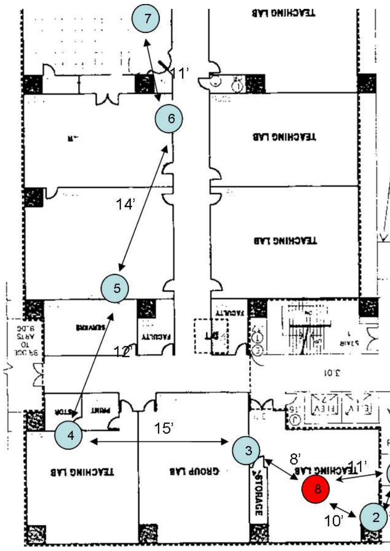

Fig. 2. Network testbed indicated by circles for wireless nodes over a background of room layout. Node 8 in the lower right corner is the attacker node.

Network Operation

During the day, the noise level changed unpredictably due to external sources of noise. This frequently changed the radio range of nodes significantly and lead to route flapping: a 3-hop route often becomes a 2-hop route and vice versa. The network performance suffers when the routes flap. This unpredictable behavior is what makes the testbed different from a simulated network or a testbed created in a single room using customized antennas with RF multiplexing and shielding [16].

Mobility

The impact of node mobility is to change the neighbors of a node and cause route breaks, which increases the control overhead. A common approach to incorporate mobility in ad hoc network testbeds is to have a several people or robots move the wireless devices in specific patterns [11]. This approach leads to realistic, but expensive and not readily repeatable, experiments. This also increases the develop-test-revise cycle time. Owing to lack of man power to physically move wireless devices and the need for quickly repeatable experiments, we opted for emulating node mobility by software means.

We added a timer function to AODV code to remove routes at random intervals bounded by a specified time, t_{max}. This does not change the neighbors of a node or any other aspect of the routing protocol, except that it causes routes to be reacquired. We experimented with various values of values of t_{max}. Smaller values, which corresponds to high node mobility, lead to significant control overhead and lower

Proceedings of the 2006 International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM'06)
0-7695-2593-8/06 $20.00 © 2006 IEEE

Authorized licensed use limited to: IEEE Xplore. Downloaded on December 4, 2008 at 17:18 from IEEE Xplore. Restrictions apply.

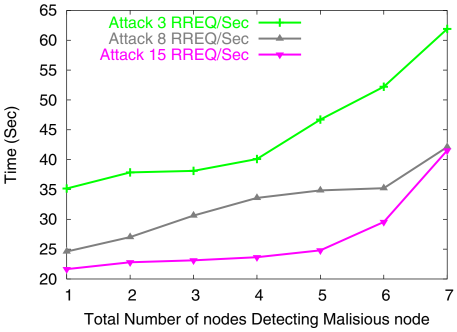

Fig. 3. Time taken to detect DoS attack for various attack rates.

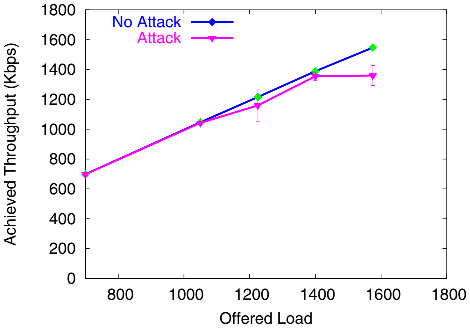

Fig. 4. Network throughput achieved under DoS attack without rate control.

peak throughputs while larger values lead to lower control overhead and better network performance. This behavior is similar to that described in many experiments on testbeds and simulations.

V. Experimental Evaluation

We conducted several experiments, each of 8-15 minutes duration, of which the first 3 minutes were used as warmup and no statistics were collected. Each configuration was run 10-20 times and the results were averaged. The 95% confidence intervals are indicated along with the averages. We used both CBR and FTP traffic over UDP and TCP transport protocols, respectively. We used 7 CBR connections and varied traffic load by varying packet injection time; for TCP traffic 7, 10 or 14 connections were used. Node 8 is the attacker node in all our experiments.

Time taken to detect an attack: Figure 3 indicates the time taken to detect an attack is measured for various attack rates with CBR traffic load at 1200 Kbps. Even without the attack, a normal node averages about 2.5 RREQs/second. So 3 RREQs/s attack by the malicious node is very close to a normal node's rate. But because of statistical averaging, it is still detected, though slowly. With higher attack rates, all nodes recognize the attack within 40 seconds. The rate control mechanism has little impact (less than 3%) on throughput in a normal network.

CBR Traffic: Figure 4 indicates the network throughput under RREQ flood attack without statistical rate control mechanism. When the rate control mechanism is invoked, the throughput is sustained under attack as shown in Figure 5. An attack rate of 15 RREQs/s is used. At a network load of 1400 Kbps, each connection source is sending out 50 500-byte packets per second. So an attack rate of 15 packets/s by the malicious node is not an excessive demand on the network BW. Since there are only seven other nodes, the leverage is not very high as in the networks used in simulations. Still, at moderate loads of 1000 Kbps, the impact of the attack can be seen. About 20% of the throughput is lost at higher loads.

It is interesting to compare the overall RREQs seen by nodes with and without rate control. Figures 6 and 7 indicate that the rate control mechanism very effective in suppressing the excess RREQs by the malicious node.

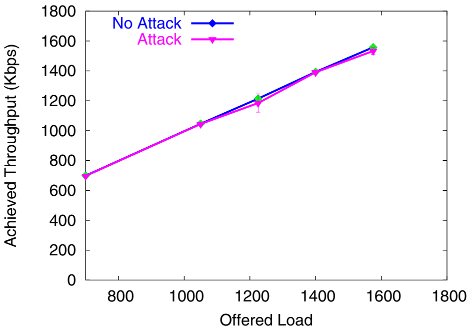

Fig. 5. Network throughput achieved under DoS attack with rate control.

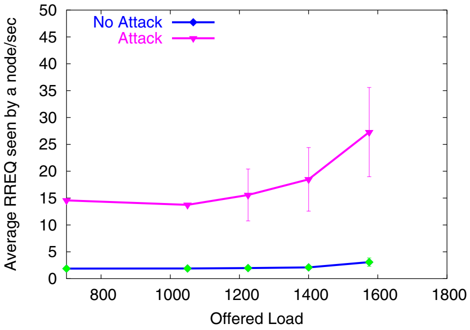

Fig. 6. RREQs seen by a node when rate control is not used.

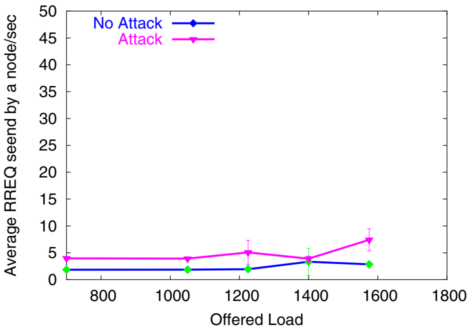

Fig. 7. RREQs seen by a node when rate control is used.

Authorized licensed use limited to: IEEE Xplore. Downloaded on December 4, 2008 at 17:18 from IEEE Xplore. Restrictions apply.

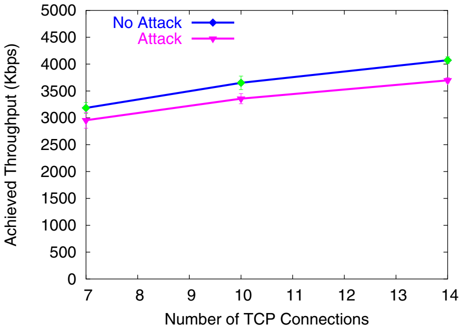

Fig. 8. TCP throughput without rate control.

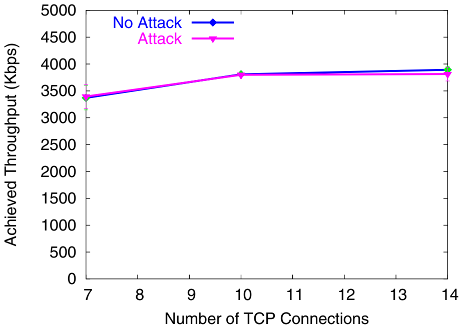

Fig. 9. TCP throughput with rate control.

TCP Throughput: Figure 8 indicates the network throughput under DoS attack without the statistical rate control mechanism. With rate control, the throughput is sustained even under attack as shown in Figure 9.

VI. Conclusions

DoS attacks that exploit flooding of control packets cause severe performance degradation. It makes malicious nodes appear as normal nodes with frequent route discoveries. Over a short period of time, route requests from normal and malicious nodes are not easy to distinguish. Our results show that even for a small network, the attack causes statistically significant impact on performance. We implemented and evaluated on a small wireless testbed, a simple statistical packet dropping mechanism that curbs attacks from malicious nodes effectively without hurting normal nodes. Furthermore, the rate control mechanism has little impact on the performance of a normal network. Another salient feature of the solution is that it causes no additional control overhead.

There has been an extensive amount of simulation study of security attacks on mobile ad hoc networks. Simulation studies do not incorporate realistic models of noise, which often causes unexpected complications. Therefore, experimental evaluation of performance and security mechanisms is necessary to validate their effectiveness. On the other hand, full-fledged experimental evaluation requires robots and vehicles to create node mobility; this makes field experiments very expensive. We believe, our testbed approach provides an intermediate step between simulation study and field-testing: actual code is developed and tested inexpensively.

References

[1] B. Awerbuch, D. Holmer, C. Nita-Rotaru, and H. Rubens, “An on-demand secure routing protocol resilient to byzantine failures,” in *Proceedings of ACM Workshop on Wireless Security (WiSe)*, 2002, pp. 21–30.

[2] R. V. Boppana and S. P. Konduru, “An adaptive distance vector routing algorithm for mobile, ad hoc networks,” in *Proceedings of IEEE INFOCOM*, 2001, pp. 1753–1762.

[3] Cisco Systems, Inc., “Linksys WRT54G wirelss-g broadand router,” 2004. [Online]. Available: http://www.linksys.com

[4] T. Clausen and P. Jacquet, *Optimized Link State Routing Protocol (OLSR)*, October 2003, IETF RFC 3626. [Online]. Available: http://www.ietf.org/rfc/rfc3626.txt

[5] S. Desilva and R. V. Boppana, “Sustaining performance under traffic overload,” in *Proceedings of Int'l Conference on Wireless Networks*, vol. 1, 2004, pp. 3–8.

[6] S. Desilva and R. V. Boppana, “Mitigating malicious control packet floods in ad hoc networks,” in *Proceedings of IEEE Wireless Communications and Networking Conference (WCNC)*, 2005, pp. 2112–2117.

[7] Y.-C. Hu, A. Perrig, and D. B. Johnson., “Ariadne: A secure on-demand routing protocol for ad hoc networks,” in *Proceedings of ACM Int'l Conference on Mobile Computing and Networking (MobiCom)*, 2002, pp. 12–23.

[8] Y.-C. Hu, A. Perrig, and D. Johnson, “Rushing attacks and defense in wireless ad hoc network routing protocols,” in *Proceedings of ACM Workshop on Wireless Security (WiSe)*, 2003, pp. 30–40.

[9] D. B. Johnson, D. A. Maltz, and Y.-C. Hu, *The Dynamic Source Routing Protocol for Mobile Ad Hoc Netowrks (DSR)*, 2004, IETF Internet Draft. [Online]. Available: http://www.ietf.org/internet-drafts/draft-ietf-manet-dsr-10.txt

[10] L. Klein-Berndt, “Kernel aodv v2.2.2,” Apr. 2004. [Online]. Available: http://w3.antd.nist.gov/wctg/aodv_kernel

[11] H. Lundgren et al., “A large-scale testbed for reproducible ad hoc protocol evaluations,” in *Proceedings of IEEE Wireless Communications and Networking Conference (WCNC)*, 2002, pp. 412–418.

[12] S. Marti, T. J. Giuli, K. Lai, and M. Baker, “Mitigating routing misbehavior in mobile ad hoc networks,” in *Proceedings of ACM Int'l Conference on Mobile Computing and Networking (MobiCom)*, 2000, pp. 255–265.

[13] R. Ogier, F. Templin, and M. Lewis, *Topology Dissemination Based on Reverse-Path Forwarding (TBRPF)*, 2004, IETF RFC 3684. [Online]. Available: http://www.ietf.org/rfc/rfc3684.txt

[14] OpenWRT Team, “OpenWRT: Experimental release,” Sept. 2004. [Online]. Available: http://downloads.openwrt.org

[15] C. E. Perkins, E. M. Belding-Royer, and S. R. Das, *Ad Hoc On Demand Distance Vector (AODV) Routing*, July 2003, IETF RFC 3561. [Online]. Available: http://www.ietf.org/rfc/rfc3561.txt?number=3561

[16] S. Sanghani et al., “EWANT: The emulated wireless ad hoc network testbed,” in *Proceedings of IEEE Wireless Communications and Networking Conference (WCNC)*, 2003, pp. 1844–1849.

[17] Seattle Wireless Group, “Seattle Wireless Project.” [Online]. Available: http://www.seattlewireless.net

[18] W. R. Stevens, *TCP/IP Illustrated, Volume 1: The Protocols*. Addison-Wesley, 1994.

[19] B. Williams and T. Camp, “Comparison of broadcasting techniques for mobile ad hoc networks,” in *Proceedings of ACM Int'l Symp. on Mobile Ad Hoc Networking and Computing (MobiHoc)*, 2002, pp. 194–205.

[20] P. Yi, Z. Dai, Y. Zhong, and S. Zhang, “Resisting flooding attacks in ad hoc networks,” in *Information Technology: Coding and Computing (ITCC)*, vol. 2, Apr 2005.

[21] M. Zapata and N. Asokan, “Securing ad-hoc routing protocols,” in *Proceedings of ACM Workshop on Wireless Security (WiSe)*, 2002, pp. 1–10.

[22] Y. Zhang and W. Lee, “Intrusion detection in wireless ad-hoc networks,” in *Proceedings of ACM Int'l Conference on Mobile Computing and Networking (MobiCom)*, 2000, pp. 275–283.

IEEE COMPUTER SOCIETY

Authorized licensed use limited to: IEEE Xplore. Downloaded on December 4, 2008 at 17:18 from IEEE Xplore. Restrictions apply.