On the Effectiveness of Monitoring for Intrusion Detection in Mobile Ad Hoc Networks

Rajendra V. Boppana, *Senior Member*, *IEEE*, and Xu Su, *Member*, *IEEE*

Abstract—Several intrusion detection techniques (IDTs) proposed for mobile ad hoc networks rely on each node passively monitoring the data forwarding by its next hop. This paper presents quantitative evaluations of false positives and their impact on monitoring-based intrusion detection for ad hoc networks. Experimental results show that, even for a simple three-node configuration, an actual ad hoc network suffers from high false positives; these results are validated by Markov and probabilistic models. However, this false positive problem cannot be observed by simulating the same network using popular ad hoc network simulators, such as ns-2, OPNET or Glomosim. To remedy this, a probabilistic noise generator model is implemented in the Glomosim simulator. With this revised noise model, the simulated network exhibits the aggregate false positive behavior similar to that of the experimental testbed. Simulations of larger (50-node) ad hoc networks indicate that monitoring-based intrusion detection has very high false positives. These false positives can reduce the network performance or increase the overhead. In a simple monitoring-based system where no secondary and more accurate methods are used, the false positives impact the network performance in two ways: reduced throughput in normal networks without attackers and inability to mitigate the effect of attacks in networks with attackers.

Index Terms—Mobile ad hoc networks, intrusion detection, passive monitoring, false positives, analytical models, noise modeling, performance analysis.

1 Introduction

A mobile ad hoc network (MANET) is a collection of wireless devices moving in seemingly random directions and communicating with one another without the aid of an established infrastructure. To extend the reachability of a node, the other nodes in the network act as routers. Thus, the communication may be via multiple intermediate nodes between source and destination. Since MANETs can be set up easily and inexpensively, they have a wide range of applications, especially in military operations and emergency and disaster relief efforts [9]. However, MANETs are more vulnerable to security attacks than conventional wired and wireless networks due to the open wireless medium used, dynamic topology, distributed and cooperative sharing of channels and other resources, and power and computation constraints [28].

Intrusion detection systems (IDSs), which attempt to detect and mitigate an attack after it is launched, are very important to MANET security. Several monitoring-based intrusion detection techniques (IDTs) have been proposed in literature [20], [7], [21], [2]. In a monitoring-based IDT, some or all nodes monitor transmission activities of other nodes and/or analyze packet contents to detect and mitigate active attackers. Intuitively, it is easy to see that monitoring-based intrusion detection is not likely to be accurate for ad hoc networks due to varying noise levels and varying signal propagation characteristics in different directions. An IDT uses additional mechanisms such as trust values for nodes before considering nodes to be suspicious. Even with such additional mechanisms, monitoring neighbors' transmissions is the key technique that triggers the detection process for many IDTs. Most evaluations of IDTs are based on small testbed configurations, or simulations which do not incorporate any realistic environmental noise models. More significantly, there are neither reports on the extent of the false positive problem nor on the quantification of the effectiveness of monitoring.

In this paper, we quantify false positives and analyze their impact on the accuracy of monitoring-based intrusion detection. We use a combination of experimental, analytical, and simulation analyses for this purpose. First, using a linear chain of three off-the-shelf wireless routers, we show that a sender of data packets falsely suspects, based on the monitoring of transmission activities in its radio range, its next hop of not forwarding its packets (though 98 percent of its packets are delivered to its destination). We validate the experimental results by deriving a Markov chain to model monitoring and estimate the average time it takes for a sender to suspect its next hop. However, this phenomenon cannot be observed using the commonly used simulators such as ns-2, Glomosim or OPNET since they do not implement realistic models of environmental radio noise and thus cannot simulate the false positives that are seen in an actual network. To remedy this deficiency, we use a previously proposed probabilistic noise model based on the generalized extreme value (GEV) distribution to model the noise levels seen in our experiments [23]. We incorporate the GEV noise model in the Glomosim simulator and show that net impact of false positives seen in the experimental testbed can now be recreated reasonably accurately with simulations. Finally, we use the simulator fortified with the

- *R.V. Boppana is with the Department of Computer Science, The University of Texas at San Antonio, One UTSA Circle, San Antonio, TX 78249. E-mail: boppana@cs.utsa.edu.*
- *X. Su is with Microsoft Corporation, One Microsoft Way, Redmond, WA 98052. E-mail: kevsu@microsoft.com.*

Manuscript received 11 Mar. 2009; revised 27 May 2010; accepted 12 July 2010; published online 28 Oct. 2010.
For information on obtaining reprints of this article, please send e-mail to: tmc@computer.org, and reference IEEECS Log Number TMC-2009-03-0084. Digital Object Identifier no. 10.1109/TMC.2010.210.

1536-1233/11/$26.00 © 2011 IEEE Published by the IEEE CS, CASS, ComSoc, IES, & SPS

noise model to simulate large MANETs to study the impact of noise on intrusion detection. Our results indicate that monitoring-based intrusion detection has very high false positives, which impact its capability to mitigate the effect of attacks in networks with attackers.

The rest of the paper is organized as follows: Section 2 describes the effect of false positives in monitoring using experiments on a three-node testbed. Section 3 presents analytical models to validate the experimental results. Section 4 presents the measurement and modeling of background noise for wireless devices. Section 5 incorporates proposed GEV noise model and evaluates monitoring-based approaches in large networks. Section 6 presents related work and Section 7 concludes the paper.

2 Testbed Evaluation of False Positives

In monitoring-based intrusion detection, each node monitors the forwarding behavior of its neighboring nodes. In most cases, a node only monitors its next hop in a route. Consider a three-node segment of a route (with at least two hops) being used to send data packets. If the three nodes are denoted as node 1 (source or the node closer to source), node 2, and node 3 (destination or the node closer to destination), then node 2 is the next hop of node 1 and node 3 is the next hop of node 2. When node 1 transmits a data packet to node 2, it expects to hear node 2's transmission of this packet to node 3 within some specified amount of time. If the fraction of packets not overheard by node 1 exceeds a specified threshold, then node 1 concludes that node 2 is dropping too many data packets and suspects it to be a malicious node.

For monitoring purposes, node 1 keeps track of a window of packets that it sent recently to its next hop. Two types of windows can be used to keep track of monitoring: *fixed* window or *sliding* window. Let W be the monitoring window size. Also, assume that each packet is given a sequence number, starting at 1. Let j be the sequence number of the most recent packet sent to the next hop. With fixed window monitoring, packets numbered $\lfloor (j-1)/W \rfloor W + 1, \ldots, j$ are monitored. The size of the monitoring window varies from 1 to W. With sliding window, packets $j - W + 1, \ldots, j$ for $j > W$ or $1, \ldots, j$, for $j \leq W$, are monitored. Both types of windows are illustrated in Fig. 1.

Let us consider a detection scenario with a threshold of T; so if $L = \lceil WT \rceil$ packets are not overhead within the current window, then the next hop is suspected. To understand the similarities and differences between the fixed and sliding windows, let us assume that noise does not impact the overhearing of transmissions within a node's radio range. In such a scenario, a malicious node can drop up to $L - 1$ packets out of W on the average without risking suspicion by neighbors. However, the temporary drop rates can be different. For example, a malicious node can drop as many as $L - 1$ packets at the end of one window and another $L - 1$ at the beginning of the next window and still not be suspected when fixed windows are used for monitoring. The sliding window approach is free of this deficiency since in any consecutive W-transmitted packets, a malicious node may drop at most $L - 1$ packets without risking suspicion by neighbors. Therefore, with the fixed windows approach, a malicious node can afford to drop packets at a faster rate, at times. The drawback of the sliding windows approach is that it can lead to higher false positives in noisy environments.

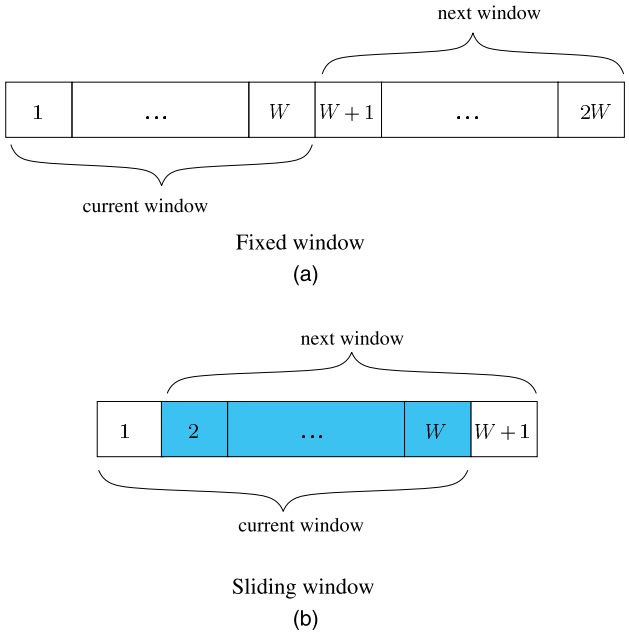

Fig. 1. (a) Fixed window and (b) sliding window illustration. W is the window size.

2.1 Testbed Experiments

To understand the extent of false positives in monitoring, we used a wireless network testbed of three Linksys wrt54g Wi-Fi routers [10]. The wrt54g routers have a built-in four-port 100 Mbps Ethernet switch, an 801.11g access point, two standard omnidirectional antennas, a 200 MHz MIPS processor, and 16 MB of RAM and 4 MB of flash memory, which serves as the disk memory. We reprogrammed the routers using OpenWrt Linux [22], [4]. This testbed was set up as a linear chain in a long corridor in a building with adjacent routers 20′ apart. All three routers use the same ssid (which is different from the other Wi-Fi devices in the building-wide 802.11b/g production network) so that they can communicate among themselves only. To minimize the interference, these three routers use a different (noninterfering) channel from those used by other access points. Also, to minimize interference from moving objects and signals from cell phones, we carried out our experiments early mornings from 2:00 am to 5:00 am.

One end router (denoted as node 1) sends packets to the other end router (node 3) via the intermediate router (node 2). We use static routes in node 1 and node 2 to ensure that the next hop for packets transmitting from node 1 is node 2 and the next hop for packets transmitting from node 2 is node 3. RTS/CTS handshake is used to reduce frame collisions due to the hidden terminal problem. Node 1 is set to promiscuous mode and monitors (overhears) transmissions from node 2 to node 3.

In each experiment, node 1 transmits at a rate of 200 Kbps (fifty 500 byte packets/s) for up to 80 seconds. A single CBR over UDP connection is used. Node 2 transmits every packet it receives from node 1 to node 3.

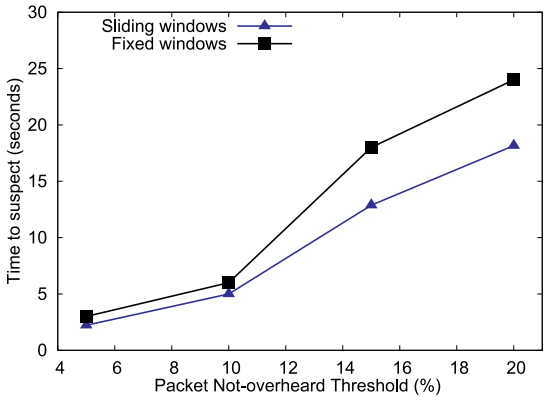

Fig. 2. Time to suspect next hop in monitoring-based approach in a three-node wireless testbed. $W = 150$, which corresponds to a 3 second interval for the packet rates used.

Every node records the ID of each packet it receives, transmits, or overhears. The packet trace from each router is sent to a desktop machine via the Ethernet connection of the routers. After the experiment, we analyzed the packet traces obtained from the three nodes. We removed the traces for the first 500 packets, which were considered to be part of the network warmup. With the MAC level ACK mechanism in the 802.11 protocol, node 1 can determine if a packet it transmitted is received successfully by node 2. Therefore, we considered only the packets that were successfully received by node 2 in our analysis of false positives.

We used these preprocessed packet traces to compute the percentage of packets received by node 2, but not overheard by node 1. If l is the number of packets in the monitoring window that is not overheard, then

$$q_w = \frac{l}{W} \tag{1}$$

is the fraction of successfully transmitted packets, but not overheard for the current window. (Though the size of the window varies from 1 to W when fixed window monitoring is used, using W rather than the current window size in the denominator results in the correct calculation of q_w.) If $q_w \geq T$ (equivalently, $l \geq \lceil WT \rceil$), where T is the threshold to suspect next hop, then node 1 suspects node 2 of dropping data packets. An IDT uses additional mechanisms such as trust values to actually suspect the nodes. Even in such cases, not overhearing is a key event that triggers the detection process.

We ran the experiments 24 times since the noise levels change frequently and unpredictably. For each experiment, given a window size, W, and threshold, T, we determined the time node 1 takes to suspect node 2. If j is the number of packets sent by node 1 at the point where q_w exceeded threshold, we can calculate the time to suspect the next node as $\frac{j}{\lambda}$, where λ is based on the f sending rate. Fig. 2 gives the average time it takes for node 1 to suspect node 2 as malicious sliding window monitoring. Even though the overall packet delivery ratio is about 98 percent and node 2 transmits all the packets it receives, node 1 suspects it within a few seconds, even with high threshold values ($T > 10\%$).

The three-node testbed is small, nodes are stationary, and only one connection between the end nodes with static routes is used to eliminate routing overhead and contention among the test nodes. Since there is only one active connection, there is no interference noise from other node transmissions within the studied network. All of the noise seen by the nodes is generated by external sources in the environment surrounding the nodes. Though all nodes are normal, the environmental (background) noise causes node 1 not to overhear some of node 2's transmissions. Even though the overall packet delivery ratio is about 98 percent, node 1 suspects node 2 within a short period of time. However, this phenomenon cannot be observed using the commonly used simulators, such as ns-2, Glomosim, or OPNET, since they use a constant background noise as the default noise model and do not implement realistic models of environmental radio noise. This points out the inadequacy of the evaluations of monitoring-based detection techniques using simulators. Therefore, it is important to understand the impact of noise on monitoring techniques. To this extent, we develop analytical models to validate the experimental results and to study the effectiveness of monitoring.

2.2 Additional Notes and Discussion Regarding the Experiments

We have conducted a large number experiments, though the data and the graphs we present are based on 24 experiments. We varied transmission (Tx) power using the *wl* program that came with the driver supplied by the router manufacturer. We used three settings: 32, 64, and 128 mW. We also varied the distance between the routers initially, but we choose the power setting of 32 mW and 20′ spacing between routers to ensure high packet delivery rate (98 percent in our experiments). Our objective is to show that, in a normal scenario with very few actual packet losses, monitoring can be highly error prone.

The overheard rate may be impacted by noise as well as multipath effects due to the long corridor we used for the experiments. The multipath effects should be analyzed more carefully by varying the distances between the routers. The experiments we conducted do not separate the multipath effects from noise. However, as we show in Section 4.2, the simulations based on the noise modeling we proposed and the standard open space radio signal propagation model used by the Glomosim simulator [31] match the experimental results closely. Though we do not directly show, our simulations seem to indicate that multipath effects are not a significant factor in our experiments. Also, studies have shown that radio noise in a typical work environment is very high [8]. Our own measurements (Section 4) confirm this. For this reason also, we believe that noise rather than multipath effects is the significant factor in the experiments we reported. We believe that the multipath effects are more significant at higher Tx power levels. This was confirmed by our experiments for Tx power settings of 64 and 128 mW, which resulted in higher not-overheard rates. We did not use this data in our analysis.

Though the our testbed is small, we believe that it is sufficient to show (a counter example) that monitoring is inaccurate. If monitoring is not effective in a three-node network, it likely to be even less effective in a larger MANET where there is interference due to transmissions by other nodes which adds to the background noise.

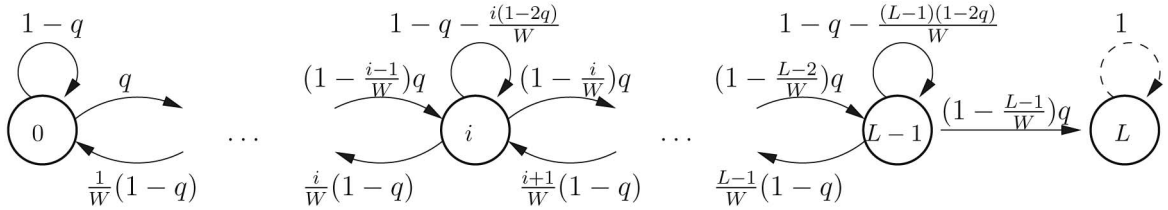

Fig. 3. A finite Markov chain with an absorbing state. q denotes not-overheard probability.

3 Analytical Models

We now present an analytical model to validate the experimental results. Let t_i, r_i, and o_i denote, respectively, the number of packets transmitted, received from previous hop, and overheard retransmissions of next hop by node i, for $i = 1, 2, 3$. It is clear that $r_1 = o_2 = t_3 = o_3 = 0$ for the three-node setup we used in the experiments. Also, $t_1 \geq r_2 \geq t_2$ and $o_1 \leq t_2$. If node 2 is not malicious and no packets are lost due to congestion (which is the case in our experiments), then $r_2 = t_2$. We calculate the *overall not-overheard* rate due to environmental noise, denoted q, as follows:

$$q = \frac{r_2 - o_1}{r_2}, \tag{2}$$

$$\begin{aligned} p_{i,i-1} &= \mathcal{P}[\text{The oldest packet in current window is not overheard } \cap \text{ The newest packet in next window is overheard}|\text{current state} = s_i] \\ &= \mathcal{P}[\text{The oldest packet in current window is not overheard}|\text{current state} = s_i] \cdot \mathcal{P}[\text{The newest packet in next window is overheard}|\text{current state} = s_i] \\ &= \frac{i}{W}(1-q), \quad if \quad 0 < i < L, \end{aligned} \tag{3}$$

$$\begin{aligned} p_{i,i+1} &= \mathcal{P}[\text{The oldest packet in current window is overheard} \cap \text{The newest packet in next window is not overheard}|\text{ current state} = s_i] \\ &= \mathcal{P}[\text{The oldest packet in current window is overheard}|\text{ current state} = s_i] \cdot \mathcal{P}[\text{The newest packet in next window is not overheard}|\text{current state} = s_i] \\ &= \left(1 - \frac{i}{W}\right)q, \quad if \quad 0 \leq i < L. \end{aligned} \tag{4}$$

It is noteworthy that node 1 knows r_2 due to MAC level ACKs from node 2. The not-overheard rate can also be considered as the probability that a packet received by node 2 was not overheard by node 1. The not-overheard rate q is a key parameter in the development of the analytical model.

3.1 Sliding Window Model

We model the state of sliding-window-based monitoring using a discrete-time Markov chain. More specifically, we use the number of not-overheard packets in the monitoring window as the state of the monitoring by node 1. The window slides to the right with each packet received by node 2. Therefore, packet receptions of node 2 are the time steps in the Markov chain.

The discrete-time Markov chain has $L + 1$ states, where $\frac{L-1}{W} < T \leq \frac{L}{W}$, as shown in Fig. 3. The state i, denoted as s_i, indicates the case where i packets in the current window are not overheard by node 1. State s_0 denotes the state where all of the W packets in the current window are overheard. State s_L indicates the state where L of the most recent W packets is not overheard, which means the fraction of not-overheard packets is beyond the threshold to suspect the monitored node. The purpose of the Markov model is to determine analytically the expected time to suspect its next hop by a monitoring node. Therefore, s_L is an absorbing state. Such Markov models are commonly used to analyze the expected time to encounter a bug in a software system [27].

Given that the Markov chain starts in state s_0, the average number of steps (packet transmissions) it takes to reach state s_L indicates the time it takes to suspect the next node.

To complete the Markov model, we need to derive the state transition probabilities. Let $p_{i,j}$ denote the transition probability from state s_i to state s_j, i.e., the probability that the number of packets not overheard in next sliding window will be j given the value i in the current sliding window. Only transitions $s_i \to s_{i+1}$ for $0 \leq i < L$, $s_i \to s_{i-1}$ for $0 < i < L$, and $s_i \to s_i$ for $0 \leq i \leq L$ are feasible since with any new transmission, the number of not-overheard packets can increase by 1, decrease by 1, or remain the same. So $p_{i,j} = 0$, if $|i - j| \geq 2$. Since s_L is an absorbing state, $p_{L,j} = 0$, for $j \neq L$. Assuming that the not-overheard packets in a sliding window are uniformly distributed, $p_{i,i-1}, p_{i,i+1}$, and $p_{i,i}$ are given in (3), (4), and (5).

$$p_{i,i} = \begin{cases} 1 - p_{i,i-1} - p_{i,i+1} & \text{if } 0 < i < L \\ \quad = 1 - \frac{i}{W}(1-q) - \left(1 - \frac{i}{W}\right)q & \\ \quad = 1 - q - \frac{i(1-2q)}{W} & \\ 1 - q & \text{if } i = 0 \\ 1 & \text{if } i = L. \end{cases} \tag{5}$$

The transition probability matrix of the Markov chain is given by

$$P = \begin{pmatrix} p_{0,0} & p_{0,1} & 0 & \cdots & \cdots & 0 & 0 \\ p_{1,0} & p_{1,1} & p_{1,2} & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & \cdots & 0 & p_{L-1,L-2} & p_{L-1,L-1} & p_{L-1,L} \\ 0 & \cdots & \cdots & \cdots & \cdots & 0 & 1 \end{pmatrix}.$$

It can be partitioned so that

$$P = \left(\begin{array}{c|c} Q & \mathbf{C} \\ \hline \mathbf{0} & 1 \end{array}\right) \tag{6}$$

where Q is $(L-1)\times(L-1)$ substochastic matrix describing the probabilities of transition only among the transient states. $\mathbf{C}$ is a column vector and $\mathbf{0}$ is a row vector of $(L-1)$ zeros.

Now, the n-step transition probability matrix P^n has the form

$$P^n = \left(\begin{array}{c|c} Q^n & \mathbf{C}' \\ \hline \mathbf{0} & 1 \end{array}\right) \tag{7}$$

where $\mathbf{C}'$ is a column vector whose elements will be of no further use. The (i,j) entry of matrix Q^n denotes the probability of being in state s_j after exactly n steps starting from state s_i. For the finite Markov chain with an absorbing state, the matrix $I-Q$ has an inverse, $M=(I-Q)^{-1}$. Let t_i be the expected number of steps before the chain enters the absorbing state, given that the chain starts in state s_i. Let t be the column vector whose $(i+1)$th entry is t_i. Then, $t=Mv$, where v is a column vector, all of whose entries are 1. Note that t_0 is the expected number of steps before entering the absorbing state when the chain starts from state s_0. Therefore, for the experimental network, t_0 denotes the number of packets that node 1 transmits before it suspects that node 2 as malicious node. If the average packet sending rate by node 1 is λ, the average time taken for node 1 to suspect node 2 will be $\frac{t_0}{\lambda}$.

For each experimental data set, we calculated the not-overheard rate q using (2) from the end of warmup to the instant of suspecting the next hop and compared the time to suspect value obtained from the analysis of experimental data with the model's estimate for the same q value. Fig. 4 gives the time to suspect values from experiments and the model as a function of q. Once again, we used window size $W=150$. These figures show that experimental results match the results from the Markov model, especially for thresholds $\leq 10\%$.

3.2 Fixed Window Model

Let X denote the random variable for the number of not-overheard packets in a fixed window of size W and q denote the overall not-overheard rate. Then,

$$\mathcal{P}[X=i] = \binom{W}{i} q^i (1-q)^{W-i}.$$

In a fixed window, the probability that less than $L=\lceil WT\rceil$ packets are not overheard is given by

$$\mathcal{P}[X<L] = \sum_{i=0}^{L-1} \binom{W}{i} q^i (1-q)^{W-i}.$$

Then, the average number of fixed windows that need to be checked before a fixed window has L or more packets not overheard is

$$N = \frac{1}{1-\mathcal{P}[X<L]} = \frac{1}{1-\sum_{i=0}^{L-1}\binom{W}{i} q^i (1-q)^{W-i}}. \tag{8}$$

Since the monitoring node checks if q_w, window not-overheard rate given by (1), exceeds the threshold even before the current window is full, the average time taken for node 1 to suspect node 2 can be calculated as

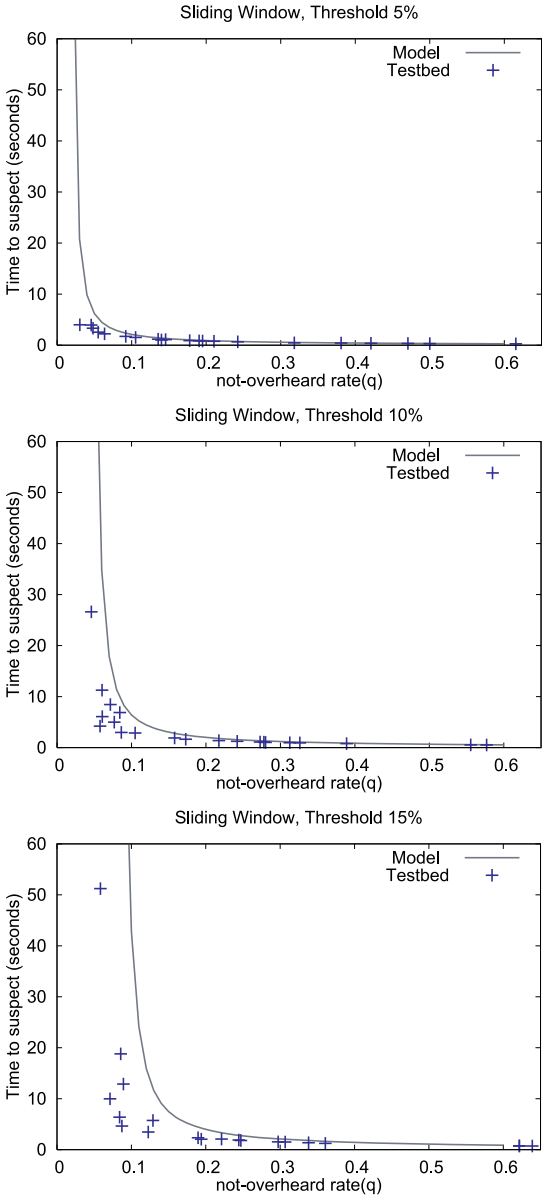

Fig. 4. Comparison of time to suspect values from experiments and Markov model for sliding window monitoring.

$[(N-1)\cdot W + R_{fw}]/\lambda$, where R_{fw} is the average size of the current fixed window when the threshold is reached. R_{fw}, estimated using a truncated negative binomial distribution, is given by

$$R_{fw} = \frac{\sum_{k=L}^{W} k\cdot\left[\binom{k-1}{L-1} q^L (1-q)^{k-L}\right]}{\sum_{k=L}^{W}\left[\binom{k-1}{L-1} q^L (1-q)^{k-L}\right]}. \tag{9}$$

Once again, we compare the time to suspect values from the experiments with those given by the model for the same q value. The results are given in Fig. 5.

A visual inspection of Figs. 4 and 5 indicates that the analytical models and experimental results match closely. For a more rigorous evaluation, we calculated the root-mean-square error (RMSE), a commonly used statistical

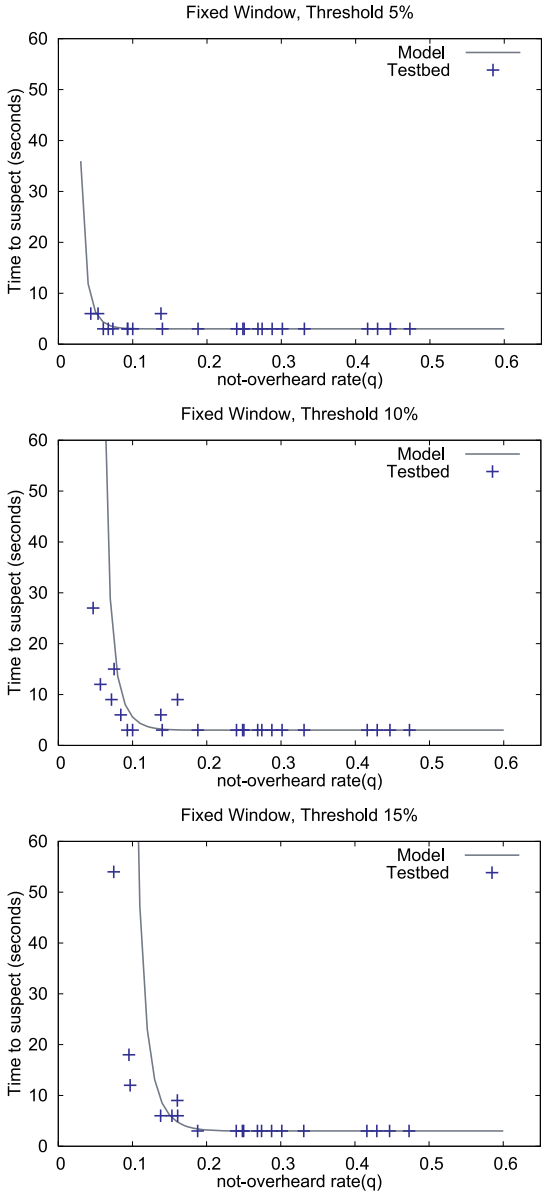

Fig. 5. Comparison of time to suspect values from experiments and model for fixed window monitoring.

measure, to compare the differences between the experimental results and the corresponding model values (see Fig. 6). Smaller RMSE values indicate that experimental results match those from models better. When the threshold is 10 percent and sliding window is used, there is about 3 second difference between the time to suspect obtained from experiments and those estimated from Markov model.

Given that monitoring is imperfect and environmental noise could increase false positives, it is surprising that none of the published results on monitoring-based intrusion detection techniques analyzed the impact of noise. Also, to the best of our knowledge, there are no extensive evaluations of monitoring techniques using testbeds (with 10 seconds of nodes), and most large network evaluations were done using simulations. This points out a major inadequacy of the existing simulators for ad hoc networks: the lack of a reasonable background noise model. In the next section, we develop a parameterized noise model that

Threshold	5%	10%	15%
Fixed window	2.4653	4.8528	7.5718
Sliding window	0.4499	3.0180	7.1270

Fig. 6. RMSE for the values obtained from experiments and the analytical models.

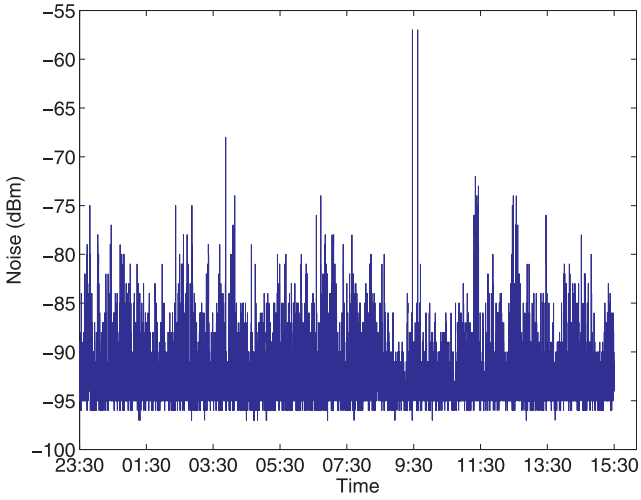

Fig. 7. Background noise changes for 16 hours (11:30 pm to 3:30 pm the next day).

can be incorporated in the current simulators to simulate environmental noise.

4 Noise Model for Simulators

In this section, we describe a parameterized noise model that we originally developed in an earlier work to study the impact of noise on the performances of routing protocols [23]. We used an expanded testbed of eight Linksys wrt54g Wi-Fi routers to measure the background noise. We obtained the noise levels using the *wl* program that came with the driver supplied by the router manufacturer. This noise information is sent to a specified desktop machine via the Ethernet. Due to clock resolution, each router could provide the noise level it sees once in every 100 ms. Fig. 7 shows the noise data sampled from one of the eight routers in the testbed. The noise levels are much higher than the default ambient noise levels (for example, -100.97 dBm at 290 K temperature in Glomosim) used in current simulators. (The current simulators model interference from other nodes' transmissions, whereas the ambient noise is generated by independent external sources in the environment surrounding the node.) Also, the noise fluctuates and at times reaches very high values. The noise data gathered from other routers have the same distribution and nearly identical histogram charts.

4.1 GEV Noise Model

We used MATLAB to analyze and model the noise levels captured in our measurements. MATLAB [26] has an extensive library of distributions including Gaussian, gamma, and lognormal. In such diverse fields as image processing, architectural acoustics, and electronic music, it is often assumed that noise conforms to Gaussian distribution. But, we found that neither Gaussian nor any of the

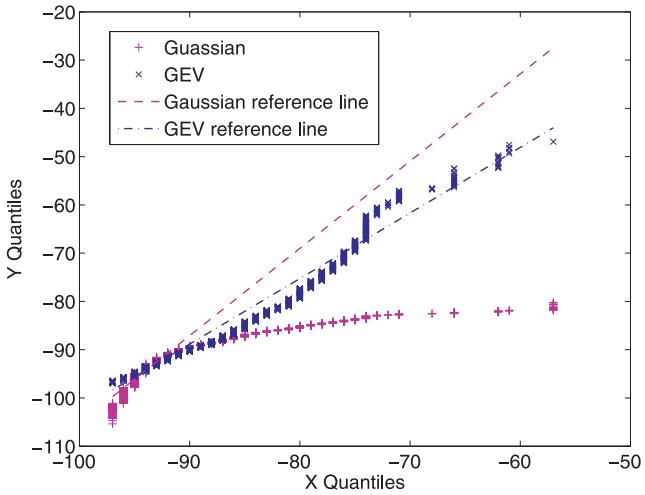

Fig. 8. Quantile-Quantile plot between generated data and the empirical data.

commonly used distributions, such as gamma and lognormal model, the noise levels seen by wireless routers accurately. Further investigation revealed that the GEV distribution [14] models this background noise fairly accurately. If X is a GEV random variable, then its cumulative distribution function (CDF) and probability density function (PDF) are given as follows:

$$F(x;\mu,\sigma,\xi) = \exp\left\{-\left[1+\xi\left(\frac{x-\mu}{\sigma}\right)\right]^{-1/\xi}\right\} \quad \text{for } 1+\xi\left(\frac{x-\mu}{\sigma}\right) > 0, \tag{10}$$

$$f(x;\mu,\sigma,\xi) = \frac{1}{\sigma}\left[1+\xi\left(\frac{x-\mu}{\sigma}\right)\right]^{-1/\xi-1} \exp\left\{-\left[1+\xi\left(\frac{x-\mu}{\sigma}\right)\right]^{-1/\xi}\right\} \quad \text{for } 1+\xi\left(\frac{x-\mu}{\sigma}\right) > 0, \tag{11}$$

where μ is the location parameter, the scale parameter is $\sigma > 0$, and the shape parameter is ξ. The shape parameter ξ governs the tail behavior of the distribution.

We also drew the *quantile-quantile* (Q-Q) plot of the empirical data and the corresponding GEV random variates. If the theoretical distribution (GEV, in this case) accurately models the empirical data (sampled noise, in our case), then the Q-Q plot would be linear [13]. Fig. 8 shows that the points from GEV distribution fall closer along their reference line than the points from Gaussian distribution. Therefore, GEV distribution models the measured noise data better than the Gaussian distribution. The estimated parameters of GEV distribution for the sampled data are: $\mu = -93.768$ dBm, $\sigma = 1.579$, and $\xi = 0.179$. The sampled data from other routers can also be modeled using GEV distribution with slightly different parameter values. This is to be expected since the environmental noise changes slightly for different labs or offices even on the same floor of a building. See [23], which introduced the noise model, for more details.

We incorporated the GEV noise model with the default parameters $\mu = -93.768$ dBm, $\sigma = 1.579$, and $\xi = 0.179$, which correspond to the empirical data, in Glomosim simulator. The GEV random variates are generated using the inverse transform technique [13]. The default noise model in Glomosim is a constant value -100.97 dBm, corresponding to 290 K temperature. When GEV noise model is used, the GEV random variate for the background noise level for each node is determined every 1 ms. Each node has a different random seed, which ensures that the noise levels generated are independent.

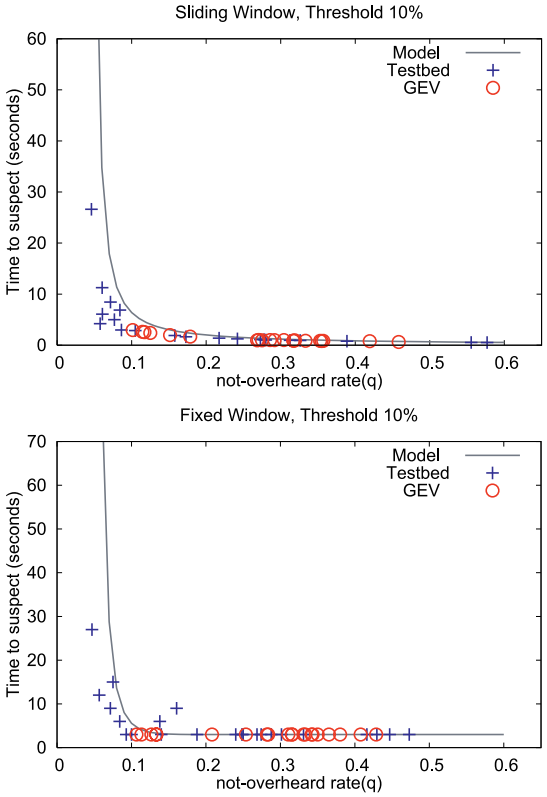

Fig. 9. Time to suspect comparison between testbed, model, and Glomosim simulation with GEV noise model. Window size $W = 150$.

4.2 Three-Node Network Simulations

We simulated the three-node experimental network using the Glomosim simulator. When the default noise model is used, node 1 never suspects node 2. However, when GEV noise model is used, node 1 tends to suspect node 2. The time to suspect value depends on the q value, which depends on the GEV parameters as seen in the experiments. In GEV, ξ only affects the tail behavior of the distribution slightly, and σ has more impact on the distribution. So, we adjusted σ to simulate network environments with different q values.

We compared the time to suspect values obtained from simulations with those from the experiments and the analytical model estimates. Owing to space considerations, we show, in Fig. 9, the comparisons for the case of 10 percent threshold only.

The RMSE calculations, given in Fig. 10, for the simulated and analytical model values indicate that there is a close agreement between the analytical models and the simulations.

Threshold	5%	10%	15%
Fixed window	0.2647	0.2193	1.8623
Sliding window	0.0002	0.0279	2.8541

Fig. 10. RMSE for difference between simulation results and corresponding values estimated from models.

GEV model is similar to the naive sampling indicated by Lee et al. [15]. They propose the closest-fit pattern matching (CPM) approach to generate noise from sampled noise traces to efficiently and accurately simulate packet delivery. Although the CPM technique captures the autocorrelation effects better, we choose GEV model for the following reasons: 1) GEV model is a simple parametric model which can be easily adjusted to simulate different background noise profiles, while CPM requires a new noise trace files in each case; 2) CPM is computationally expensive; 3) GEV model gives reasonably accurate results based on our evaluation of simulation and experimental results.

Another method to verify the proposed noise model is to collect the noise level at nodes 1 and 2 during the experiment and use them for simulations to see if the not-overheard rate q and time to suspect values match those from the corresponding experiment. If an experiment duration is t seconds, $t \leq 80$ seconds, the number of noise samples collected will be 80,000 or less assuming 1 ms sampling interval. This may be enough to develop an accurate parametric model. However, our approach was to develop a general noise model based on samples collected for 16 hours, and see if the simulator fortified with the noise model produces results that match the experimental results.

5 Simulation of Mobile Ad Hoc Networks

We used the Glomosim simulator, v2.03 [31] to evaluate the effectiveness of monitoring in larger mobile ad hoc networks using both GEV noise model (with parameters $\mu = -93.768$ dBm, $\sigma = 1.579$, and $\xi = 0.179$ that give the default $q = 0.1$) and the Glomosim default noise model, which is a constant noise of -100.47 dBm ($q = 0$). The actual not-overhead rate is higher due to interference from competing transmissions in an ad hoc network. Each node maintains a monitoring window for each traffic flow (connection) though it. In each traffic flow, each data packet sent from the source node is assigned an increasing ID. Only when current node overhears next hop forwarding packets j, it will consider packets with ID between i and j as not-overheard, where i is ID of the last overheard packet and $i < j$. Therefore, it can avoid false positives due to random back offs at the MAC layer.

We implemented the Watchdog intrusion detection technique (denoted, WD) proposed in [20] as a representative monitoring-based IDT. Following the description given in [20], our implementation has three components: *watchdog*, *pathrater*, and sending extra route request when all routes contain one or more suspicious nodes. In the watchdog component, each node that sends or forwards data packets monitors its next hop. When a node suspects its next hop, it will sends an ALARM message to the source node (if it is not the source). When a route break occurs, the monitoring windows in the broken route path are cleared.

Number of Nodes	50
Node Speed	[1-19] m/s
Node Mobility	Modified Random Waypoint
Pause Time	0 second
Field Size	1500 m $\times$ 300 m 2200 m $\times$ 440 m
Warmup time	200 sec.
Total simulation time	1800 sec.
Attack start time	600 sec. (if used)
Radio Range	250 m
MAC	802.11
Number of Traffic Pairs	10
Traffic Load	100 Kbps (CBR/UDP)
Routing Protocol	DSR
Data Packet Payload	500 bytes
Link BW	2 Mbps
Noise Models:	
Glomosim default	-100.97 dBm (constant)
GEV noise model:	
μ	-93.768 dBm
σ	1.579
ξ	0.179
Monitoring:	
Threshold, T	10%
Window type	Sliding and fixed
Window size, W	150

Fig. 11. Simulation parameters.

In the pathrater component, nodes that are not suspected are given a small positive value, less than 1, as their initial rating, which is increased gradually with passage of time. When an alarm message is received by the source node of a route, it will assign a rating of -100 to the suspected node. The rating of a path is the average of the ratings of the nodes on the path. The source chooses the highest rated path if there are multiple positive paths to the same destination. If all paths to its destination have negative ratings, then a new route discovery is initiated (the third component of the IDT) to find a path with positive rating. Although WD is a simple IDT, its primary element—monitoring—may be used as the key step to initiate the detection process in more elaborate IDSs.

We used both sliding and fixed window monitors in our simulations since the type of window used in [20] is not specified. However, both produced nearly identical results (often, the curves for both cases are superimposed on each other when plotted). Therefore, we present the results for the sliding window case only.

The simulation parameters are listed in Fig. 11. With 50 nodes, the node densities (ρ, the average number of nodes in a radio transmission area) are about 10 for the larger fields and 22 for the smaller fields. We chose long corridor type fields so that routes are likely to have multiple hops. (If one-hop paths are used most of the time or if the network is disconnected most of the time, then the impact of the attacks and the effectiveness of Watchdog IDT cannot be seen clearly.)

In order to avoid packet losses due to congestion, we only used 100 kbps traffic load. We use the following performance metrics to evaluate the effectiveness of monitoring:

- *Number of nodes suspected.* The total number of nodes suspected by one or more nodes in the network.
- *Total false positives.* The total number of times that normal nodes are suspected.

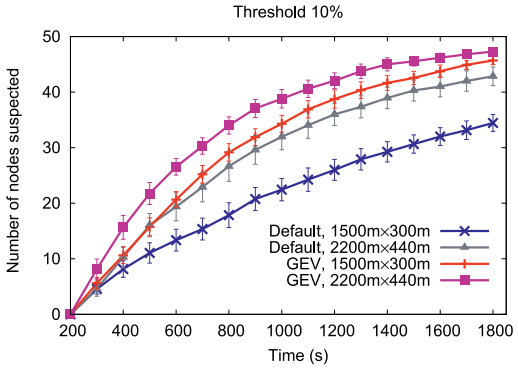

Fig. 12. Number of normal nodes to be suspected in normal ad hoc networks.

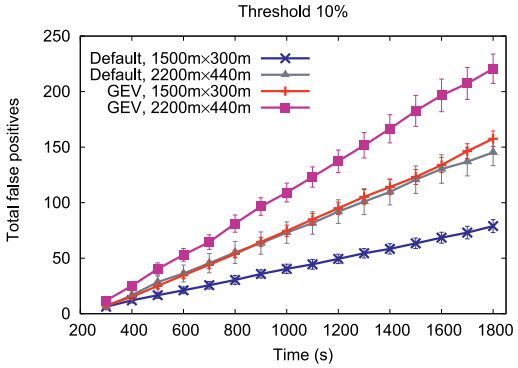

Fig. 13. Total false positives in normal ad hoc networks.

All simulations were run for 1,800 seconds with 200 seconds first used for warmup; and the attackers, in the simulations with attacks, start dropping packets at 600 seconds. Each configuration was repeated 20 times and the results were averaged; the 95 percent-level confidence intervals are indicated for all data points.

5.1 False Positives in Normal Mobile Ad Hoc Networks

First, we ran a set of simulations to see the extent of false positives in MANETs. We used only monitoring of next hops; there were no malicious nodes in these simulations. Figs. 12 and 13 give the number of normal nodes suspected, and total false positives, respectively, as a function of simulation time in both high-density and low-density networks with threshold $T = 10\%$. When GEV noise model is used, nodes are suspected much faster and more false positives occur. If the simulation is run for long enough time, all nodes in the network will be suspected. Even when default constant background noise is used, there are many false positives due to interference noise from competing transmissions. It is interesting to note that false positives are higher in low-density networks than in high-density networks though the interference noise is likely to be less in the former networks. The reason is, in low-density networks, the hop distances are larger and signals overheard during monitoring are weaker correspondingly. Also, since there are more hops in each route in the low-density network, there are more chances that nodes will be suspected. Although fewer false positives occur when the threshold is higher (e.g., 15 percent), malicious nodes can take advantage of it and drop more packets without being detected. Therefore, in the remaining of this paper, we choose 10 percent as the detection threshold.

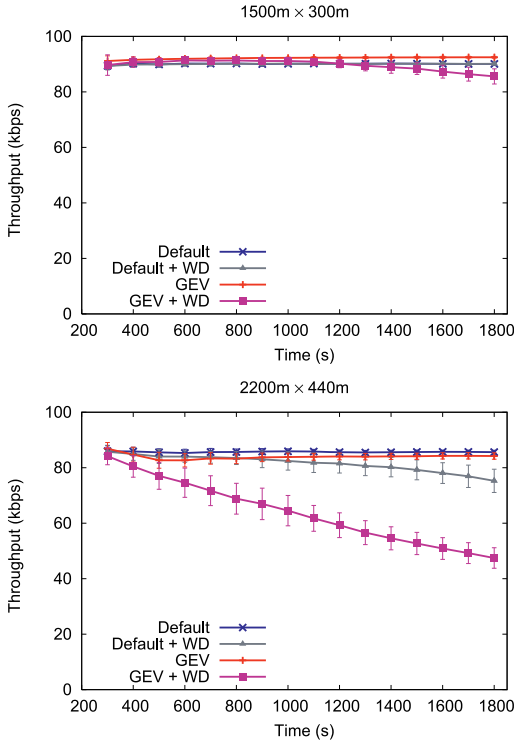

Fig. 14. Throughput in normal networks.

5.2 Impact of Intrusion Detection Technique on Normal Networks

There are too many false positives when monitoring is used in normal mobile ad hoc networks, especially when the background noise is simulated using the GEV noise model. However, it is not clear if the false positives have any impact on the network performance: since there may be multiple paths between a source and its destination, when a node is suspected, an alternate path that does not involve the node may be used without any loss of performance. Therefore, in this set of simulations, we used the overall network throughput as the performance metric. We measured the network throughput with and without GEV noise model. Then, we turned on the Watchdog IDT (explained above), reran the same configurations and measured the network throughput.

We measured the number of delivered data bytes every 100 seconds after the warmup time. The network throughput at any time is given by dividing the total bytes delivered up to that point since warmup by the time elapsed since the warmup.

Fig. 14 shows impact of Watchdog IDT (denoted, WD) on the network throughput in both high-density and low-density networks without attacks. In a high-density network, WD does not affect the network throughput significantly since sources can find alternate paths to get around the false positives. But in low-density networks, due to very high false positives and due to relatively fewer alternate paths, WD hurts the network performance, especially when GEV background noise model is used.

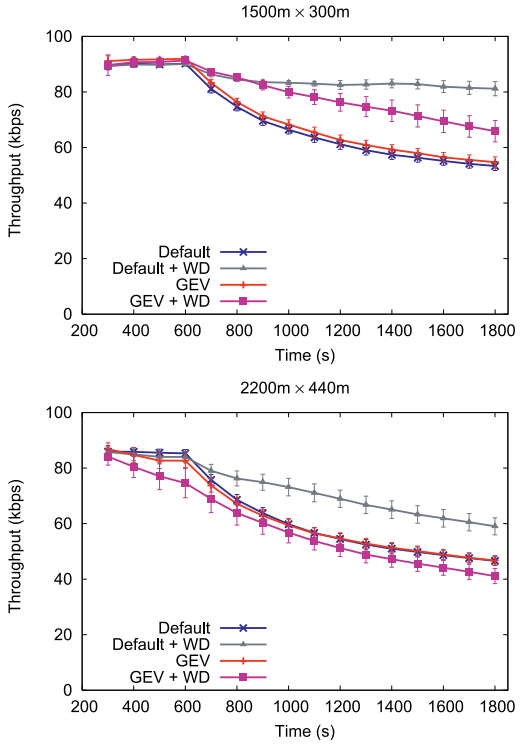

Fig. 15. Throughput in networks with attackers. Ten nodes are malicious nodes which drop all received data packets.

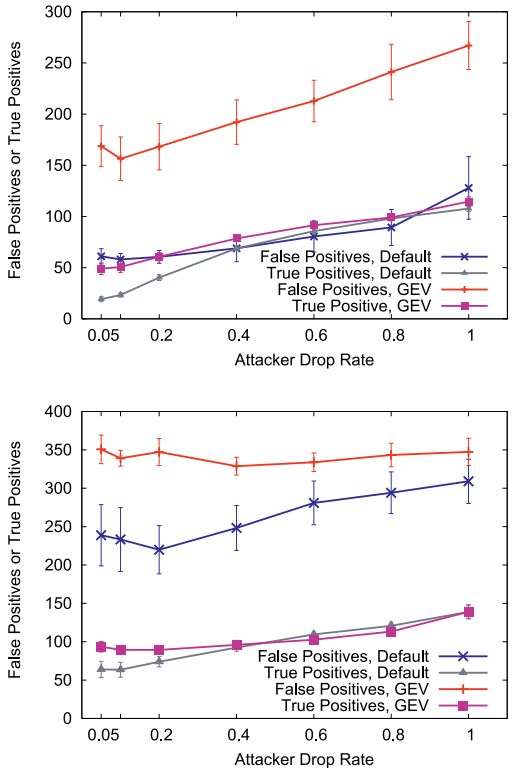

Fig. 16. Comparison of false positives and true positives with 10 attackers in networks with field sizes $1{,}500 \times 300$ m^2 (top) and $2{,}200 \times 440$ m^2 (bottom).

5.3 Effectiveness of Intrusion Detection Technique in Networks with Attackers

Next, we evaluated the effectiveness of WD when there are attackers, who participate in the route discovery process as normal nodes but drop all received data packets. We simulated the case where 10 of the network nodes are malicious and drop all data packets. This is the attack model used in [20]. It is noteworthy that it is much harder to detect malicious nodes when they drop selectively. Therefore, the simulated attack presents an easier challenge for an IDT.

Fig. 15 shows network throughput when the 10 malicious nodes drop all received data packets starting at simulation time of 600 seconds. Without WD, the network throughput degrades to 40 percent in the high-density network, and to 46 percent in the low-density network. In the high-density network, with WD active, the network throughput is improved from 53.3 to 90.1 kbps with default background noise, and from 54.7 to 65.8 kbps with GEV background noise. In the low-density network, however, WD does not mitigate the impact of the attacks, especially when GEV noise model is used.

To further understand the throughput behavior, we looked at the total false positives and true positives for different packet drop rates (see Fig. 16). The two networks differ significantly in the number of true and false positives. First, let us consider the dense network with $1{,}500 \times 300$ m^2 field. The number of false positives is larger than the number of true positives when drop rate is low (5 to 20 percent), and false positives are close to true positives when for 40 to 100 percent drop rates. It is difficult to differentiate malicious nodes from normal nodes, especially when the drop rate is low. As shown in Fig. 15, WD is effective for this case and mitigates packet dropping reasonably well (the throughput drop is arrested near the end of simulation). Under the GEV noise model, the number of false positives is two to three times the number of true positives. Consequently, WD fails to mitigate packet dropping effectively.

Let us consider the sparse network, $2{,}200 \times 440$ m^2. Even with the default background noise model, the number of false positives are about 2.5 times that of true positives. Because of the lower node density, the distances between consecutive nodes in a path are likely to be closer to the end of the nominal radio range; therefore, the interference noise, which is modeled by the simulator, from competing transmissions by other nodes in the network is a significant factor. When GEV noise model is used, the number of false positives are even higher—about 3.5 times the true positives. Consequently, WD has an adverse impact on the network performance when GEV noise model is used: the throughput is slightly higher when WD is not used even when the attackers drop packets.

5.4 Discussion

The simulation results for Watchdog IDT in [20] appear to differ significantly from those presented in this paper for several reasons. Marti et al. [20] use a square field with high node density ($\rho > 20$). Therefore, the paths used in their simulations are mostly short paths with many alternative paths to bypass suspected nodes. Also, they simulated their networks with default noise model and for only 200 seconds after the attacks are launched. As the network operates

longer, the performance deteriorates significantly, especially when the environment is noisy.

A typical IDS will complement monitoring with additional techniques to reduce false positives. Even in such systems, monitoring because of the ease and simplicity of its implementation is likely to be the early warning indicator, which triggers more accurate and expensive secondary methods. Obviously, there is an overhead or cost associated with false positives. If an early warning system is prone to a large number of false positives, then there is a significant cost incurred even if nodes are not falsely suspected or labeled as malicious. This cost can be the loss of throughput, more control packet transmissions, increased energy consumption, or larger packet delays.

To quantify this cost, we used WD. Of course, WD does not have a secondary warning system, and the network throughput captures the effects of false positives due to monitoring completely, especially in a normal network with no attacks. In this sense, our use of WD may be considered as that of an aggregation technique that combines the cost of false positives due to monitoring into a quantifiable throughput loss.

In a more practical system in which a simpler early warning technique triggers a more expensive and accurate detection method, these false positives may lead to another type of cost. If the most common case is no attacks, then the high overhead caused by false positives may reduce use of the IDS with users opting to manually turn it on when high security is needed or to run it only on nodes with sufficient energy, computational, and network resources.

If monitoring is used simultaneously with other techniques, to reduce the lead time, then the high false positive rate due to monitoring is likely to pollute the data used for detection. We have not evaluated this effect in this study.

It is possible to reduce the number of false positives due to monitoring by having higher threshold values, allowing a node to exceed the not-overheard threshold multiple times before labeled as suspicious, or both. This will likely mitigate the false positive problem in normal networks without attacks (the most common case). However, such a design reduces the true positive rate, and the network becomes less responsive and less resilient to attacks. In the unlikely event of an attack, the detection will take a long time since malicious nodes can exploit the detection rule by alternating between normal and attack modes and stay undetected for long periods.

6 Related Work

Many IDTs for MANETs have been proposed in literature. They can be classified as: signature-based detection, anomaly detection, and specification-based detection. A survey of intrusion detection techniques is given in [25]. Based on how the data needed for intrusion analysis are gathered, IDTs for MANETs can be divided into three approaches: monitoring-based, probing-based, or explicit feedback among intermediate nodes in routes. (Explicit feedback among end nodes is commonly used for both security and performance tuning. We do not specifically review the literature on this technique.)

Watchdog and pathrater [20] are the first monitoring-based technique proposed for ad hoc networks. In this approach, nodes monitor transmission activities of neighboring nodes and analyze packet contents to detect and mitigate an attack after it is started. When a node suspects its next hop, it will send an alarm message back to source node. Pathrater is used to punish suspicious nodes by not including them in routing. However, monitoring-based intrusion detection is not likely to be accurate for ad hoc networks due to varying noise levels, varying signal propagation characteristics in different directions, and interference due to competing transmissions within the network. In this paper, we showed monitoring gives very high false positives when environmental noise effects are considered. We tried to complement the existing results by quantifying the benefits and overheads of watchdog in more realistic noise conditions.

The Watchdog technique has been extensively studied for its deficiency, false positives and has been modified or supplemented it with other mechanisms to make it more accurate. Specific results include CONFIDANT [7], [5], CORE [21], and LARS [12]. These results use different policies to propagate monitored information (trust) to others in order to mitigate misbehavior and enforce cooperation. In particular, Buchegger and Le Boudec [5] present a Bayesian approach to assign trust and reputation ratings the CONFIDANT system. Their simulation results (with the default noise model of a constant value) show that incorporating secondary trust information gathered from other nodes with the primary trust information directly gathered (by monitoring) can significantly speed up the detection of misbehaved nodes. The effectiveness of these approaches needs to be carefully evaluated with more realistic noise simulation models or experiments.

There are several other papers on using a reputation/trust system for MANETs [11], [19], [18]. Luo et al. [19] describe a localized trust model in which multiple nodes are collaboratively used to provide authentication services. Eschenauer et al. [11] describe a trust framework which encompasses Pretty Good Privacy (PGP) [32] like trust models. Liu et al. [18] present a dynamic trust model to address packet drops by selfish and malicious nodes. In general, a trust system requires propagation and dissemination of trust. Also trust evidence must be distributed redundantly to handle the unreliable connectivity in MANETs [11]. Trust propagation is complex, not well understood in the context of ad hoc networks, in which trust collection and dissemination may be incomplete and problematic and has high computational requirements (e.g., collaborative authentication [19]) and communication overhead (requiring localized or limited distance network floods [18]).

Buchegger et al. [6] modified passive monitoring with a passive acknowledgment (PACK) mechanism in which a node resends a not-overheard packet multiple times. Using a three-node testbed, they show that PACK loses significantly fewer packets than a nonmonitoring-based approach. However, the retransmission overhead is not analyzed. PACK requires additional book keeping, such as timers and retransmissions. Based on our result that noise makes

overhearing unreliable, it is likely that many of these retransmissions are redundant.

In probing-based approaches [1], [29], [30], [16], nodes query other nodes and receive their reception and transmission of data. Analyzing this information, they can detect intruders. However, probing-based approaches have different issues. First, it will incur more delay to detect malicious nodes since an anomalous activity needs to be suspected/identified prior to probing for relevant data from other nodes. Second, malicious nodes can give false probe data to avoid detection. Third, malicious nodes can also collude to *avoid detection, or* frame up legitimate nodes, or deceive legitimate nodes to send incorrect information.

The explicit feedback approach by Liu et al. [17] requires downstream nodes (toward the destination) send explicit ACKs to upstream nodes two hops away from them. This achieves the intended effect of monitoring with explicit ACK packets between nodes two hops away from each other. This method overcomes the limitations of monitoring at the cost of additional packet transmissions, book keeping, and computational overhead. We have used a similar approach, denoted p-hop crosscheck, $p \geq 2$, to detect control packet falsification in on-demand route discoveries in another paper [24]. In addition to the computational and book-keeping overhead, this approach works only for isolated and noncolluding malicious nodes. For example, if there are two malicious nodes in a row in a route that always send the anticipated feedback ACKs upstream and ignore any ACKs from downstream nodes, then the 2ACK scheme for data packets as well as the two-hop crosscheck for control packets do not work [24].

This paper is based on our earlier conference paper [3]. Compared to the conference version, this paper presents the fixed window model, additional analysis for different threshold values, more details on RMSE analysis, description of the GEV noise model, and significantly more simulation results including the simulation analyses of a second network with higher node density. The GEV noise model and the experiments to capture the noise samples are originally described in another paper [23], which investigates the impact of noise on the performances of routing protocols. This paper uses the noise model to investigate passive monitoring and its effectiveness.

7 Conclusions

Several monitoring-based intrusion detection techniques proposed in literature rely on each node passively monitoring the data forwarding by its next hop to mitigate packet dropping attacks by insider nodes. Though monitoring-based intrusion detection is not likely to be accurate for ad hoc networks due to varying noise levels, varying signal propagation characteristics in different directions, and interference from competing transmissions, there are no specific studies on the impact of noise on false positives and the impact of false positives on network performance.

In this paper, we presented quantitative evaluations of false positives in monitoring-based intrusion detection for ad hoc networks. We showed that, even for a simple three-node configuration, an actual ad hoc network suffers from high false positives. We validated the experimental results using discrete-time Markov chains and probabilistic analysis. However, this problem of false positives cannot be observed by simulating the same three-node network using popular ad hoc network simulators such as ns-2 with mobility extensions, OPNET or Glomosim, because they do not simulate the noise seen in actual network environments. To remedy this, we developed a parameterized noise model based on GEV distribution function. With the noise model incorporated in the Glomosim simulator, we showed that the three-node network simulation reveals the same false positive patterns that the experimental network produced and the analytical models predict.

We used the simulator fortified with the GEV noise model to study the impact of monitoring-based intrusion detection on larger ad hoc networks. Our results indicate two potential problems with monitoring-based IDT: 1) IDT may reduce performance of a normal network, especially when the network is not dense, and 2) IDT may not improve the network throughput since any mitigation of packet dropping by malicious nodes is offset by suboptimal paths used owing to false positives.

The IDT we evaluated is a simple one and depends primarily on monitoring. A more elaborate IDT may use additional mechanisms such as trust values of nodes and cross-checking other nodes monitoring data before actually suspecting a node. However, even in such techniques, monitoring may be used as the key step to initiate the detection process. This can increase the overhead of intrusion detection and may deter its use. In light of that our results indicate a fundamental problem with monitoring-based IDTs: the key technique used is unreliable, and any detection process based on it is likely to be error prone.

In future, we intend to investigate the effectiveness of probing techniques in the presence of colluding attackers. We also would like to develop new intrusion detection techniques that avoid the problems of passive monitoring.

Acknowledgments

The authors would like to thank the reviewers for several comments and suggestions, which led to more detailed discussions of the experiments and simulations in Sections 2.2 and 5.4 and improved the paper. X. Su was at the University of Texas at San Antonio when the research described in this paper was conducted. This research was partially supported by NSF grant 0551501, AIA grant F30602-02-1-0001, and a grant from the DMEA.

References

[1] B. Awerbuch, D. Holmer, C. Nita-Rotaru, and H. Rubens, "An On-Demand Secure Routing Protocol Resilient to Byzantine Failures," *Proc. ACM WiSe,* pp. 21-30, Sept. 2002.

[2] S. Bansal and M. Baker, "Observation-Based Cooperation Enforcement in Ad Hoc Networks," Research Report cs. NI/0307012, Standford Univ., 2003.

[3] R.V. Boppana and X. Su, "An Analysis of Monitoring Based Intrusion Detection for Ad Hoc Networks," *Proc. IEEE Globecom: Computer and Comm. Network Security Symp.,* Dec. 2008.

[4] R.V. Boppana and S. Desilva, "Evaluation of a Stastical Technique to Mitigate Malicious Control Packets in Ad Hoc Networks," *Proc. Int'l Symp. World of Wireless Mobile and Multimedia Networks (WoWMoM)/Workshop Advanced Experimental Activities on Wireless Networks and Systems,* pp. 559-563, 2006.

[5] S. Buchegger and J.Y. Le Boudec, "A Robust Reputation System for Mobile Ad-Hoc Networks," *Proc. Workshop Economics of Peer-to-Peer Systems (P2PE '04),* 2004.
[6] S. Buchegger, C. Tissieres, and J.Y. Le Boudec, "A Test-Bed for Misbehavior Detection in Mobile Ad-Hoc Networks – How Much Can Watchdogs Really Do?" *Proc. IEEE Workshop Mobile Computing Systems and Applications (WMCSA '04),* 2004.
[7] S. Buchegger and J.-Y. Le Boudec, "Performance Analysis of the Confidant Protocol: Cooperation of Nodes Fairness in Dynamic Ad-Hoc Networks," *Proc. IEEE/ACM MobiHoc,* 2002.
[8] R. Burchfield, E. Nourbakhsh, J. Dix, K. Sahu, S. Venkatesan, and R. Prakash, "RF in the Jungle: Effect of Environment Assumptions on Wireless Experiment Repeatability," *Proc. IEEE Int'l Conf. Comm. (ICC '09),* pp. 1-6, 2009.
[9] I. Chlamtac, M. Conti, and J.J.-N. Liu, "Mobile Ad Hoc Networking: Imperatives and Challenges," *Ad Hoc Networks,* vol. 1, no. 1, pp. 13-64, 2003.
[10] Cisco Systems Inc., Linksys WRT54G v2.2 Wireless-G Broadband Router, http://www.linksys.com, 2004.
[11] L. Eschenauer, V.D. Gligor, and J. Baras, "On Trust Establishment in Mobile Ad-Hoc Networks," *Proc. Security Protocols,* pp. 47-66, 2003.
[12] J. Hu, "Cooperation in Mobile Ad Hoc Networks," Technical Report TR-050111, Dept. of Computer Science, Florida State Univ., 2005.
[13] R. Jain, *The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling.* John Wiley & Sons, 1991.
[14] M.R. Leadbetter, G. Lindgreen, and H. Rootze, *Extremes and Related Properties of Random Sequences and Processes.* Springer-Verlag, 1983.
[15] H. Lee, A. Cerpa, and P. Levis, "Improving Wireless Simulation through Noise Modeling," *Proc. ACM Int'l Conf. Information Processing in Sensor Networks (IPSN '07),* pp. 21-30, Apr. 2007.
[16] K. Liu, J. Deng, P.K. Varshney, and K. Balakrishnan, "An Acknowledgement Based Approach for Detection of Routing Misbehavior in Manets," *IEEE Trans. Mobile Computing,* vol. 6, no. 5, pp. 488-502, May 2007.
[17] K. Liu, J. Deng, P.K. Varshney, and K. Balakrishnan, "An Acknowledgment-Based Approach for the Detection of Routing Misbehavior in MANETs," *IEEE Trans. Mobile Computing,* vol. 6, no. 5, pp. 488-502, May 2007.
[18] Z. Liu, A. Joy, and R. Thompson, "A Dynamic Trust Model for Mobile Ad Hoc Networks," *Proc. 10th IEEE Int'l Workshop Future Trends of Distributed Computing Systems (FTDCS '04),* pp. 80-85, 2004.
[19] H. Luo, P. Zerfos, J. Kong, S. Lu, and L. Zhang, "Self-Securing Ad Hoc Wireless Networks," *Proc. Seventh IEEE Symp. Computers and Comm. (ISCC '02),* 2002.
[20] S. Marti, T.J. Giuli, K. Lai, and M. Baker, "Mitigating Routing Misbehavior in Mobile Ad Hoc Networks," *Proc. ACM MobiCom,* pp. 255-265, Aug. 2000.
[21] R. Molva and P. Michiardi, "Core: A Collaborative Reputation Mechanism to Enforce Node Cooperation in Mobile Ad Hoc Networks," *Proc. IFIP Comm. and Multimedia Security Conf.,* 2002.
[22] Seattle Wireless Project, http://www.seattlewireless.net, 2010.
[23] X. Su and R.V. Boppana, "On the Impact of Noise on Mobile Ad Hoc Networks," *Proc. ACM Int'l Wireless Comm. and Mobile Computing Conf. (IWCMC '07),* pp. 208-213, 2007.
[24] X. Su and R.V. Boppana, "Crosscheck Mechanism to Identify Malicious Nodes in Ad Hoc Networks," *Security and Comm. Networks,* vol. 2, no. 1, pp. 45-54, 2009.
[25] B. Sun, L. Osborne, Y. Xiao, and S. Guizani, "Intrusion Detection Techniques in Mobile Ad Hoc and Wireless Sensor Networks," *IEEE Wireless Comm. Magazine,* vol. 14, no. 5, pp. 56-63, Oct. 2007.
[26] The MathWorks Inc., MATLAB, version 7.1.0.183 (R14), http://www.mathworks.com/products/matlab, 2010.
[27] K.S. Trivedi, *Probability and Statistics with Reliability, Queuing, and Computer Science Applications,* second ed. John Wiley & Sons, 2001.
[28] H. Yang, H. Luo, F. Ye, S. Lu, and L. Zhang, "Security in Mobile Ad Hoc Networks: Challenges and Solutions," *IEEE Wireless Comm.,* vol. 11, no. 1, pp. 38-47, Feb. 2004.
[29] W. Yu, Y. Sun, and K.J.R. Liu, "HADOF: Defense against Routing Disruption in Mobile Ad Hoc Networks," *Proc. IEEE INFOCOM,* 2005.
[30] W. Yu, Y. Sun, and K.J.R. Liu, "Stimulating Cooperation and Defending against Attacks in Self-Organized Mobile Ad Hoc Networks," *Proc. Second Ann. IEEE CS Conf. Sensor and Ad Hoc Comm. and Networks (SECON '05),* 2005.
[31] X. Zeng, R. Bagrodia, and M. Gerla, "GloMoSim: A Library for Parallel Simulation of Large-Scale Wireless Networks," *Proc. Workshop Parallel and Distributed Simulation,* pp. 154-161, 1998.
[32] P. Zimmerman, *PGP User's Guide.* MIT, 1994.

Rajendra V. Boppana received the BTech degree in electronics and communications engineering from the Karnataka Regional Engineering College, Mysore University, India, the MTech degree in computer technology from the Indian Institute of Technology, Delhi, and the PhD degree in computer engineering from the University of Southern California. Since 1991, he has been a faculty member in the Department of Computer Science at the University of Texas at San Antonio. His research interests are in parallel and distributed computing, performance evaluation, and wireless networks and security. He has published extensively and served on the program committees of several conferences in these areas. He is a senior member of the IEEE.

Xu Su received the BE degree in computer science and applications from Yanshan University, Qinhuangdao, China, in 1997, the MS degree in computer science from Wright State University, Dayton, Ohio, in 2002, and the PhD degree in computer science from the University of Texas at San Antonio in 2009. He has been working for Microsoft Corporation since October 2008. He is a conference web cochair for the Fifth International ICST Conference on Communications and Networking in China (ChinaCom 2010). He served on the technical program committee for the First International Conference on Next Generation Wireless Systems (NGWS 2009) and the IEEE International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob 2008). He has served as reviewer for 18 major international conferences and journals. He was listed in Marquis' *Who's Who in America*, 2009. He received a Presidential Dissertation Award from the University of Texas at San Antonio in 2008. He received a National Achievement Award in Science and Technology Progress by the Ministry of Education, Second Class, China, in 2004. His research interests include antispam, wireless network security, mobile ad hoc networks, wireless sensor networks, and networked multimedia. He is a member of the IEEE and the ACM.

▷ **For more information on this or any other computing topic, please visit our Digital Library at** www.computer.org/publications/dlib.