Passive and Active Analysis in DSR-Based Ad Hoc Networks

Tae Dempsey, Gokhan Sahin*, and Y.T. (Jade) Morton

Miami University
Department of Electrical and Computer Engineering
Oxford OH 45056, USA
{dempsetl,sahing,mortonyt}@muohio.edu

Abstract. Security and vulnerabilities in wireless ad hoc networks have been considered at different layers, and many attack strategies have been proposed, including denial of service (DoS) through the intelligent jamming of the most critical packet types of flows in a network. This paper investigates the effectiveness of intelligent jamming in wireless ad hoc networks using the Dynamic Source Routing (DSR) and TCP protocols and introduces an intelligent classifier to facilitate the jamming of such networks. Assuming encrypted packet headers and contents, our classifier is based solely on the observable characteristics of size, inter-arrival timing, and direction and classifies packets with up to 99.4% accuracy in our experiments. Furthermore, we investigate active analysis, which is the combination of a classifier and intelligent jammer to invoke specific responses from a victim network.

Keywords: ad hoc networks, dsr, classification, algorithms, jamming.

1 Introduction

Wireless ad hoc networks have fundamentally altered today's battlefield, with applications ranging from unmanned air vehicles to randomly deployed sensor networks. This, in conjunction with the proliferation of standard and non-standard architectures and algorithms, has led to the development of countless protocols to support those applications. Additionally, the widespread use of encryption and other anti-sensing techniques has made it increasingly more difficult for network researchers to characterize wireless ad hoc networks.

Security and vulnerabilities in wireless ad hoc networks have been considered at different layers, and many attack strategies and counter-measures have been proposed. One of the most common types of attacks considered is denial of service

* G. Sahin is the corresponding author. This work has been supported by the Dayton Area Graduate Studies Institute (DAGSI) through the Air Force Research Lab/DAGSI Research Fellowship Award 401374 awarded by the state of Ohio. A preliminary version of part of this work appeared in IEICE Transactions on Information and Systems, May 2009 special issue on Information and Communication System Security [2].

J. Zheng et al. (Eds.): ADHOCNETS 2009, LNICST 28, pp. 623–638, 2010.

(DoS), where a jammer may completely or partially prevent communication in the network. Although ad hoc networks are particularly prone to DoS from internal attacks, we are interested in DoS from external attackers due to the assumed use of encryption techniques.

This paper considers the intelligent sensing and jamming problem in a wireless ad hoc network. In this paper, we interchange the terms jammer and classifier, as they are the components of the same entity. We assume that all traffic is encrypted including the headers, and that a jamming node in the vicinity of the network attempts to classify the traffic (i.e., determine the type of the packets), and disrupt the operation of the network by selectively jamming the packets that would inflict the most damage. Throughout this paper, we will assume a layered sensing architecture [1], where lower layer (physical and MAC layer) sensors provide the jammer with key information regarding the traffic observed, such as the packet sizes, packet inter-arrival times, and direction. We define direction as a function of a received packet, and two packets are said to be traveling in the same direction if they were emitted from the same source node. The jamming node can then use these observations, historical traffic data, and key characteristics of known protocols to classify packets, and possibly engage in targeted jamming. Our contributions are twofold. First, we investigate the effectiveness of intelligent jamming of packets in a wireless ad hoc network using the Dynamic Source Routing (DSR) protocol [4] through a test scenario in ns-2 [6], the canonical network simulator. More specifically, we aim to quantify how intelligent a classifier should be to properly facilitate a jammer. Secondly, we develop algorithms for the automatic detection and classification of both DSR and TCP packets in encrypted wireless ad hoc networks. We assume both packet contents and header information are encrypted and thus our characterization of DSR and TCP is based solely on observable packet characteristics of size, inter-arrival timing, and direction.

The remainder of this paper is organized as follows: Section 2 describes the motivation for our work, previous research, and our contributions. Section 3 illustrates our network model and related assumptions. In Section 4 we detail the impact of packet classification accuracy on the effectiveness of intelligent jamming. We then describe our classifier for the so-called historical analyzer in Section 5 with the respective simulation results in Section 6. In Section 7 we discuss the utilization of intelligent jamming to actively analyze a network. Finally we summarize our work and explain future directions in Section 8.

2 Motivation, Previous Work, Contributions

2.1 Motivation

Automatic sensing and classification can, in turn, facilitate jamming of a network through the selection of the most critical packet types and packet flows, providing opportunities to perform stealth denial of service attacks by allowing the network to barely operate. Although it is possible to jam packets in a victim network by

simply emitting a strong interfering signal, there would be two significant drawbacks. First, such a strong signal could easily deplete the jammer node's energy resources, and second, the victim network could easily detect the signal and take anti-jamming measures. Alternatively, it has been shown that if a jammer could first identify the network protocols and jam only packets that could inflict the most damage, jamming could be significantly more efficient. Additionally, a jammer could adjust the level and frequency of its jamming in order to avoid being detected by the victim network. Through the use of intelligent sensing and jamming, significant jamming gains could be achieved in an enemy network.

Note that in order to realize these gains, the jammer would need to be able to classify the packets. In this paper, we first demonstrate the potential of intelligent jamming in a DSR ad hoc network, and then develop algorithms for historical analysis (classification of observed packets) in such a network in the presence of TCP traffic. It is important to note that we do not address the problem of online classification, which is the classification of the packets in real-time as they are being transmitted, in this paper. The focus of our historical analysis algorithms is rather on providing reliable reference data that could later be used as an input for the online classification algorithms.

DSR was selected as a protocol to be characterized because of its popularity among mobile ad hoc network routing protocols. TCP was selected as the communication protocol due its ubiquity among a majority of existing communication networks. Therefore, we believe that TCP traffic with DSR routing constitutes an important step for our methods, although we plan to expand our work to include other ad hoc routing protocols and traffic types as well.

2.2 Previous Work

Previous work for the classification of packets in encrypted ad hoc networks has been done for ad hoc networks using the Ad Hoc On Demand Distance Vector (AODV) routing protocol and TCP [1].

It has been shown that size plays an important role in identifying packets in encrypted wireless ad hoc networks. The variability of packet sizes in networks allows specific sizes to be correlated with certain packet types. Brown [1] developed a probabilistic size-only classifier to classify packets in AODV/TCP networks. The key idea here is that certain packet types tend to be associated with particular packet sizes, even though there may be some additional variations from network to network. These size variations can be modeled as probability distributions with peaks around the most typical values for each packet type. However, the main disadvantage to using the size-only classifier is the severe dependence on initial seed values for the probabilistic model. The authors in [1] attempt to work around this problem by implementing a historical timing and sequence analyzer that would in turn update the probability distribution for the size-only classifier. However, this updating mechanism still has partial reliance on size metrics, posing a recursive dependence on the initial size seed values. In this paper we present a classification algorithm that can be used to seed an online size classifier.

The authors in [1] develop a historical classifier that analyzes a histogram of packets for specific sequences based on inter-arrival timing of packets and their respective sizes. The results from their historical classifier are then used to update their size-only classifier, but are invalid given a more general traffic scenario. To the best of our knowledge, their algorithms do not explicitly consider networks utilizing a variable size congestion window for TCP, limiting the senders in their network model to transmitting only a single data packet a time. We address this issue in the classification algorithm we present later in this paper.

Additionally, there has also been some work on application layer packet classification in wired networks [8] using techniques based on Hidden Markov Models.

2.3 Contributions

The contributions of this paper can be summarized as follows. First, we present the first study of intelligent sensing and jamming in an encrypted DSR/TCP ad hoc network and quantify the impact of classification accuracy on the effectiveness of intelligent jamming. Our work also develops a classifier (historical analyzer) for encrypted DSR/TCP wireless ad hoc networks that is capable of handling a fluctuating congestion window with an accompanying dynamic DATA-ACK round trip time estimator to seed the classifier. This is accomplished through a pooling approach in conjunction with the use of direction information in classifying the observed packet sequences, in addition to the use of size and inter-arrival times proposed in [1].

3 Network Model

The network described in this section is the network configuration and topology used in simulations throughout this paper.

We consider a layered view of sensing/jamming as in [1], where each layer provides services to the layer above and makes use of the services from the layer below. We are particularly interested in the network and transport layers, where an external attacker is able to observe packet sizes and inter-arrival times.

We utilize the network simulator ns-2 to run our simulations and use the ns-2 generated traffic traces as input for our classifier algorithms. Our network consists of two mobile ad hoc nodes with DSR as the routing protocol to maintain connectivity. There is a third mobile ad hoc node, denoted as the tap node (or jamming node), which passively listens to the network and collects information related to the packets heard in the air, saving the observed packet information (size, inter-arrival time, and direction) for use by a historical analyzer. All of the nodes are equipped with standard 802.11 wireless devices and use RTS/CTS to establish access to the wireless medium. It is also assumed that the tap node is also capable of jamming packets in the air before they are delivered at the target.

Because of the layered view of sensing, we are assuming that the MAC layer is capable of distinguishing between MAC layer control packets (i.e. RTS/CTS)

and network or transport layer packets, and the MAC layer packets are filtered so that the only packets recorded are network layer and transport layer packets. The details of the actual mechanisms for distinguishing MAC level packets in a completely encrypted wireless network are beyond the scope of this paper.

As with previous work ([1],[8]), we also assume for simplicity that, we are dealing with one connection at a time in the network. Simultaneous connections create a situation in which the tap node would be observing packets from one connection overlapping with packets from another connection. In the presence of multiple connections, we would first need to demultiplex the traffic flows before applying our classifier.

4 Intelligent Jamming

Recall that intelligent jamming is the selective jamming of the packets that could inflict the most damage in a network. In this section, we explore the effectiveness of such an intelligent jammer under various packet classification accuracy assumptions. Our goal is to determine how intelligent, or accurate, a classifier must be to maximize the effectiveness of a jammer for our network. We intend to present simulation results for various scenarios, where the jammer utilizes a naive packet classifier with varying degrees of accuracy. The results will illustrate the effectiveness of the jammer as measured by the delay between packet transmissions and deliveries.

We will be simulating our network model where a source node attempts to communicate with a destination node and the tap node attempts to jam communications. To better understand the jamming scenario we intend to use, a fundamental understanding of the DSR protocol is required. When a source node A wishes to transmit a packet to a destination node B and A does not have a route to B, then A will initiate a process known as Route Discovery to find such a route to B. During this process, A will broadcast a Route Request (RREQ) to its neighbors and this request will be forwarded until it reaches the intended destination at B. Node B will then reply with a Route Reply, which contains the full route from the source at A to the destination at B. However, in the event that the request does not reach B or the reply does not reach A, A will timeout, wait a certain back-off period, and then retransmit a RREQ up to 16 total times before giving up and assuming there is no route to B.

Using this knowledge, we intend to jam packets during the Route Discovery process as the source will attempt to establish a single connection to its destination. Note that we are only interested in jamming packets transmitted by the source and so replies transmitted by the destination would be successfully received by the source. Simulating this scenario in ns-2 involved the addition of a naive probabilistic packet classifier to the mobile nodes in which packets were either classified correctly or incorrectly based on randomly sampling a probability distribution consistent with the preset classifier accuracy. Packet jamming was then simulated by dropping packets at the MAC layer.

Thus, with a perfectly accurate classifier, we expect that all of the 16 DSR RREQ packets would be jammed and consequently result in connection failure.

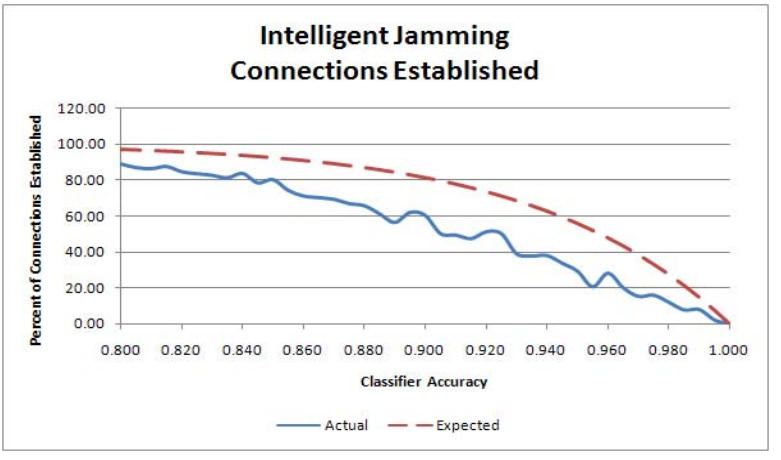

Fig. 1. Percent of connections established during intelligent jamming simulations

However, as the accuracy of the classifier decreases, the chance that a DSR RREQ packet is allowed to pass greatly increases. The probability that at least 1 DSR RREQ packet is misclassified (and Route Discovery succeeds) is plotted in Figure 1 as the dashed line. Thus, with a 95% accurate classifier, there is a 60% chance that at least one RREQ out of the 16 will reach the target. Attempting to block a DSR Route Discovery with a 90% accurate classifier proves to be virtually futile, as there is an 81.5% chance that at least one RREQ will reach the target. We note, however, that it is possible to cause significant delays in the network even with relatively low classification accuracy.

We ran simulations for classifier settings ranging from 80.0% accurate to 100.0% accurate, incrementing by 0.5% accuracy. 1000 simulations were run at each interval and the average delay before a connection was established was recorded in seconds. Each simulation exhibited the same traffic pattern to provide control over the experiment, where each simulation was ran for 100 seconds and consisted of an FTP connection starting at time 0.0 seconds and ending at 60.0 seconds. Upon starting the simulation, the source node attempts to establish connectivity with its destination node. Because the simulation and traffic both start at time 0.0, the source has no knowledge of any routes and thus has no route to its destination and must initiate DSR Route Discovery to find such a route. At this time, the tap node attempts to listen for and jam any perceived DSR packet that is being transmitted over the wireless medium from the source to the destination.

The average percentage of overall connections that is allowed to establish a connection is plotted in Figure 1 as the solid line for accuracy levels from 80.0 to 100.0. Taking into consideration Figures 1 and 2, it can be seen that there is no tradeoff between delay and accuracy or connections jammed and accuracy.

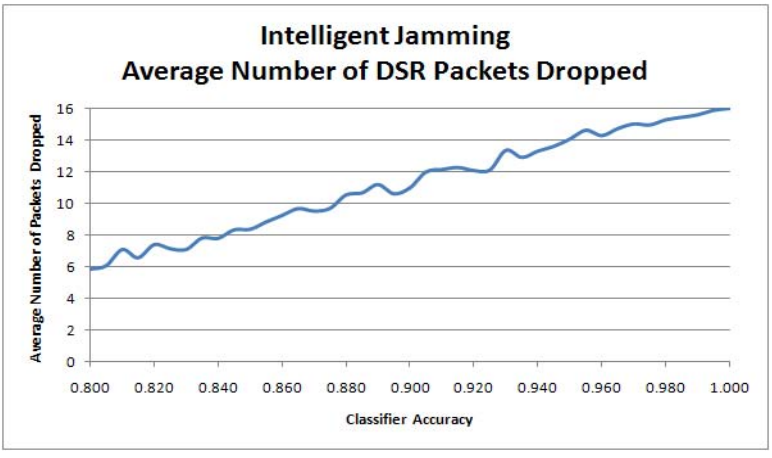

Fig. 2. Average number of DSR packets jammed before connection established

While increasing accuracy beyond 95% results in significant expected jammer performance, actual rate of increase in performance, observed from simulation, is not as significant. Additionally, significant increase in average connection delay is only observed with accuracy levels above 99%. At 95% accuracy, we observed an average connection delay of 36 seconds, no connection 71% of the time, and 80% of the time it took at least 14 DSR packets out of the 16 maximum allowed before a connection was established. Thus, we believe that a starting threshold as motivation for our algorithm accuracy of 95% is reasonable, though we intend to exceed this threshold.

We should note, however, that we are not establishing an absolute threshold for classifier algorithms in general for many reasons. The first reason being that our network model is concerned with two nodes, and that classifier algorithms in a network model consisting of many nodes and hops may operate similarly with lower accuracy or may require higher accuracy. The second reason is that it may be sufficient for a jammer to achieve an average connection delay that correlates with lower classifier accuracy. Thirdly, jammer restrictions may limit a jammer to jamming only a certain number of packets in a given time frame. Firm accuracy thresholds should be appropriately considered for other varying contexts.

5 Classification Algorithm

In this section we describe our method of classifying packets in the network based on various observable metrics. Due to the assumed encryption, the only available metrics from the packet characteristics are sizes and inter-arrival times. Additionally, a node may also distinguish packets from different senders/receivers based on information such as signal strength or angle of arrival, even though the

addresses in the packets are encrypted. Accordingly, we will also introduce the notion of direction as an input for our classifier.

Because the size-only classifier has heavy dependence on the seeded values, it has been suggested that no assumptions of the network conditions should be made and as a result the probability distributions for packet sizes should be initially flat [3]. Consequently, inter-arrival timing between packets would be the only metric to identify sequences, and from these results the seed values of size can be established. The obvious advantage of this classifier is that it allows packets to claim their distinctive peaks in the probability distributions without claiming unnecessary sizes and causing the conflict presented in the previous subsection. The timing-only classifier is typically used only after a sufficient number of packets have been collected in a histogram of packets. On the other hand, though, the disadvantage in this scheme is similar to that of the size classifier as the classifier has heavy dependence on initial seed values for the mean inter-arrival times for the packet sequences. Additionally, the authors in [1] do not address the general traffic model that allows a sender to transmit multiple packets at a time. Their algorithms assume the sender must wait for a data packet's ACK before transmitting the next data packet. Though [1] addresses the possible presence of intermediate packets as the result of co-mingled connections, intermediate packets in a single-connection environment are the result of a source node sending multiple data packets before receiving an ACK.

To resolve the issues presented in the previous work, we make no assumption of network conditions related to packet sizes and encryption levels, [3]. However, rather than relying on timing alone, we also make use of generic packet size characteristics. We use the dataline threshold from [1] in addition to other metrics, termed filters in our classifier. The term dataline simply refers to a size threshold that separates data packets from control packets, while filters are simply additional logic to either support or negate possible sequences within the packet stream.

Another additional filter that we use in our classifier is direction compatibility. The use of direction as a filter in the classification of packets is critical when considering a more general traffic model. Though we defer the details of handling this problem until the later, we illustrate the necessity of direction with the following observed sequence of packet sizes in bytes: 66, 66, and 66.

Assume for now that the time differences between the three 66-byte packets exactly match those of the TCP startup sequence while the packet sizes also match particularly well. The scheme in [1] would then classify these packets as the TCP startup sequence. However, consider now that we also take into account the direction of the packets, and observe that all of these consecutive packets actually originated from the same source node - this would contradict the classification of the TCP startup sequence, as each packet in the startup sequence travels in the opposite direction as the previous packet. As it turns out, these three packets are actually three consecutive ACK packets from previously transmitted data packets. Considering sequences alone using the proposed algorithms in [1]

would cause the classifier to perform much worse than the accuracy threshold previously determined in this paper.

Furthermore, using the knowledge that an ARP (Address Resolution Protocol) packet is the smallest packet in the network, we can search the histogram of packet sizes for the smallest packet size. Also, given that each data packet has an ACK and we have enough packets, we can search the histogram of the observed packet sizes for the size underneath the dataline that is most used, which would represent the most common ACK packet size. Once the ARP and ACK sizes have been estimated, we can use this information in leveraging sequence classifications. For example, if we are certain that the ARP size is correct then we know a series of packets cannot be an ARP sequence unless their sizes correspond with the estimated size. Likewise, the same can be done when encountering a packet matching the ACK size. In the case that a network utilizes multiple ACK sizes, extending the algorithm to identify multiple common ACK sizes is straight-forward.

5.1 Handling Fluctuating Congestion Window Sizes

As mentioned, neither [1], nor [3] explicitly handles varying congestion window sizes, limiting the sender's transmission rate. Here, we describe our initial solution to handling a varying congestion window size for a single-connection. The operation of the data packet pool, or more simply the pool, is based on TCP's behavior of sending data packets only after a connection has been established and not closing a connection until all ACK have been received at the source.

The pool is similar to a FIFO data structure with a fixed size limit and some other major differences. Unlike queues, the first element in the pool is pushed out as new elements are inserted. All elements that reside in the pool are also subject to being removed based on the amount of time spent in the pool. Both the pool-size limit and timeout limits are implemented to allow the pool to be self-maintained during classification.

As the classifier identifies data packets, it inserts those packets into the next available slot in the pool. Ideally, the first perceived ACK packet the classifier will now encounter corresponds to the first data packet, which resides in the first element of the pool. Thus, once the classifier encounters an ACK (based on the perceived size as described above), it consults the first element in the pool for its corresponding data packet. The time interval is calculated and then matched with the DATA-ACK sequence. If the match probability is high enough, then the packets are classified as the DATA-ACK sequence, and the data packet is removed from the pool. Otherwise, the data packet remains as the first element of the pool, and the perceived ACK packet is not classified and left as an ignored packet.

5.2 Cross-Protocol Detection

We have developed another initial enhancement to the classifier specific to DSR, which involves cross-protocol sequence detection. Occasionally it is helpful to

"reset" the network as [1] describes in order to start with fresh packet streams. If it is the case that the enemy network has been reset to the point where routes need to be re-established and ARP caches have timed out, then a new cross-protocol sequence can be detected that is as follows: DSR RREQ, ARP REQ, ARP REP, and DSR RREP. The impact of the detection and classification of more cross-protocol sequences should be further investigated.

5.3 Dynamic Mean Delay Calculation

As stated earlier, a major drawback to the size classifier is the potential inaccuracy of the seed mean values. Similarly, a major drawback to the time classifier is the potential inaccuracy of the seed mean delay values. We have developed an initial round trip time (RTT) estimation algorithm based on the pool concept to provide a better seed mean delay value for the DATA-ACK sequence. This enhancement is necessary since we are handling a fluctuating congestion window size, and longer ACK delays are acceptable. We will see later in this paper an alternative approach to dynamically calculating this mean using an active analysis approach.

6 Numerical Results

This section details the numerical results for our classification algorithm. Our classifier was tested against various packet streams using ns-2 generated traffic traces from a number of different simulations.

We ran four separate simulations in ns-2 to produce four varying traffic traces that could be used as input to the classifier algorithms. The goal was to produce traffic traces with enough packets to properly measure the performance of our algorithms. The first three simulations consisted of a single FTP connection between the two DSR nodes of our network model. Each of these three simulations was run for various lengths of time, ranging in duration between 20 and 40 second transfers, using a TCP congestion window of one. The fourth simulation was run, similarly with an FTP connection, with a 20 second transfer, but was allowed to have a varying TCP congestion window. The role of the fourth simulation is to determine how well our algorithm performs when faced with more general network conditions. The tap node plays no role in this simulation scenario.

Because we are utilizing data that was generated by our simulations, we know the actual packet types that we will be attempting to classify. Thus, measuring the performance of our algorithm is simply comparing the actual packet type with the guessed packet type emitted by our algorithm. A confusion matrix was constructed to illustrate the performance of our classifier and is shown in Figure 3.

Recall we developed a classifier that utilized only the metric of packet timing, as well as additional logic known as filters. These filters provide the capability of either confirming or negating possible sequences based on either direction and/or

Actual Packet Type \ Classified Packet Type	Data	Syn	Syn+Ack	Ack	Fin	ARP Req	ARP Rep	DSR Req	DSR Rep
Data	1246	0	0	0	0	0	0	0	0
Syn	0	7	0	0	3	0	0	0	0
Syn+Ack	0	0	7	3	0	0	0	0	0
Ack	0	0	0	1157	5	0	0	0	0
Fin	0	0	0	4	11	0	0	0	0
ARP Req	0	0	0	0	0	10	0	0	0
ARP Rep	0	0	0	0	0	0	10	0	0
DSR Req	0	0	0	0	0	0	0	10	0
DSR Rep	0	0	0	0	0	0	0	0	10

Fig. 3. Our algorithm results for DSR/TCP

other dynamically calculated thresholds. Additionally, we implement the concept of a data packet pool to handle more general traffic models. We also implemented an algorithm to analyze the history of packets and calculate an average round trip time to seed the DATA-ACK interval and standard deviation. The confusion matrix in Figure 3 shows the results of our timing with filters classifier. Initial seed values were pre-calculated from all four ns-2 simulation traces to provide best possible results.

Of the 2563 total packets, 2468 could be classified while 80 were either ignored or unclassified. The 80 packets that were ignored or unclassified are simply a result of the filters and are a good indication that the algorithm is not forcing infeasible classifications of packets, thus reducing the number of false positives. Note that not classifying some of the packets is acceptable, even desirable when the classification may be unreliable, since the function of the historical analyzer is to provide accurate reference data for the use of online classifiers, rather than classify all observed packets. Accordingly, excluding the unclassified packets, the classifier simulations above represent 99.4% accuracy.

Our timing with filters classifier is still vulnerable to failure when the interval seeds are inaccurate, but is more robust than the historical schemes in [1] since the DATA-ACK round trip time is calculated dynamically. Simulations that utilize the pool but not the dynamic interval algorithm result in overall classification of merely 51.4%. Note that even this rating is extremely inflated due to the high number of easy to classify data packets and is a clear indication that incorrect seed mean delay values are detrimental to the performance of the classifier.

Additional filters and metrics should be developed and added to the classifier to help refine the distinction between known sequences. Further work is being done to develop such additions.

7 Active Network Analysis

In Section 4 we introduced the concept of an intelligent jammer and discussed that it could be used to engage in targeted jamming against a victim network. In Section 5 we focused on developing classification algorithms to determine packet types based on observable packet metrics. In this section, we combine the

two concepts of jamming and classification to discuss how to analyze networks actively, by utilizing a jammer to invoke network behavior that can later be exploited by the classifier.

7.1 Controlling the Network

For a jammer to control a victim network, it must be able to dictate when and how the traffic flows between nodes in the network. The ability to control a network depends on the jammer's ability to target and jam specific packet types. Because of TCP congestion control, lost ACK packets will force a sender to retransmit data packets, severely impacting throughput and round trip times. Consecutive lost ACK results in the sender timing out, believing the connection is lost or broken, and relying on higher layers to restore the connection.

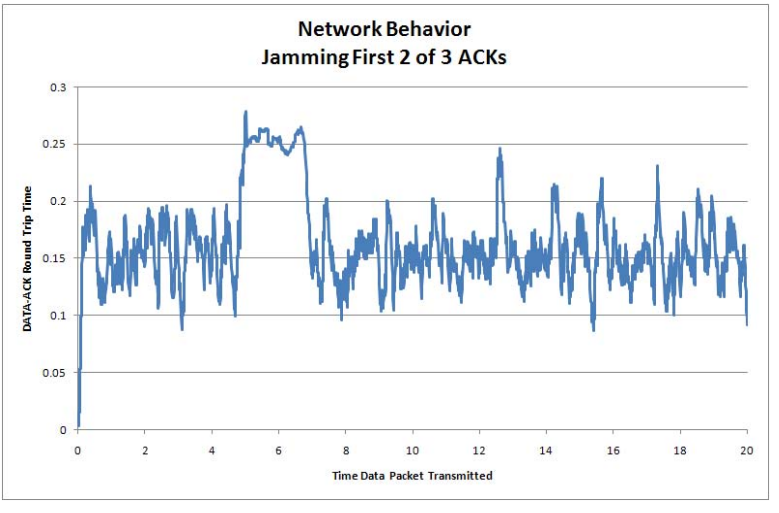

Fig. 4. Network behavior - Jamming first 2 of 3 TCP ACK packets

Active analysis can be used to intentionally introduce delay into the network. To illustrate this concept, we simulated a jamming scenario in which TCP ACK packets sent back to the sender were being actively jammed. More specifically, between the time of 5 and 7 seconds, every 1st and 2nd out of 3 ACKs were jammed. This resulted in a significant delay in the network over the 2 second jam period as the round trip time increased by 67%, from 150ms to 250ms, seen in Figure 4. While simply jamming an ACK forces TCP to retransmit a data packet and shrink its send rate, jamming two consecutive ACKs tricks TCP into thinking the network is severely congested and further slows the rate at which a sender can transmit. This results in a situation where the sender must now wait longer to retransmit the second data packet, causing the round trip time statistics to increase. Once the targeted jamming ceased, TCP thinks

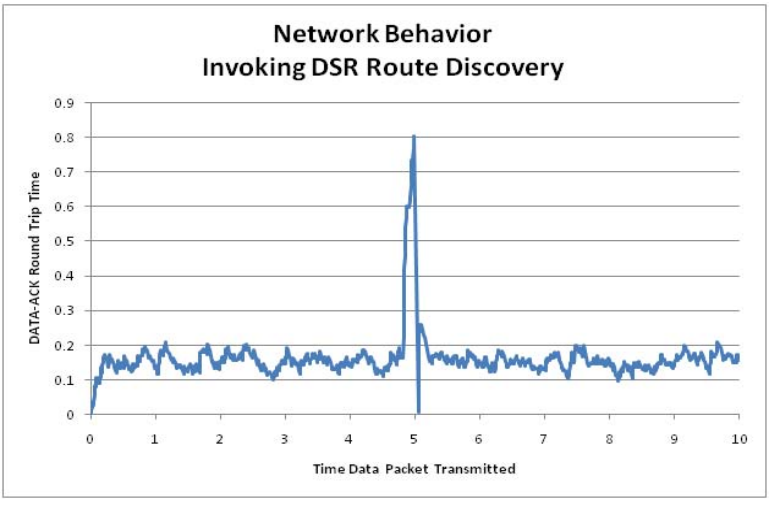

Fig. 5. Network behavior - Invoking DSR Route Discovery

congestion has returned to normal and increases the number of packets a sender can transmit, resulting in faster round trip times than during the jam period.

Active analysis can also be used to invoke specific responses from a network. To illustrate this concept, we ran a simulation in which the network was subjected to blanketed jamming. Recall that blanketed jamming, or brute-force jamming, means that all packets in the network are jammed, regardless of size, source, or time. While this is an inefficient approach and we concentrate on intelligent jamming in this paper, blanketed jamming for a small duration of time can invoke an interesting network response. In this simulation, we setup blanketed jamming for a period of 2 seconds. Unlike the targeted jamming scenario that resulted in increased network delay for a short period of time, the blanketed jamming scenario resulted in complete network disruption. Packet type information was recorded while jamming and we observed that four consecutive TCP DATA packets from the sender were jammed followed shortly by two DSR RREQ packets during the two second jam period. This means that TCP congestion control forced the sender into believing the connection was lost because it never received ACKs for the data packets, and relied upon DSR at the network layer to discover a new route.

Knowing that TCP gives up after four consecutive ACK time outs in ns-2, we setup another simulation in which the jammer targets four consecutive data packets and specifically invokes the DSR Route Discovery mechanism from the victim network. The jamming period was designated to begin at 5 seconds and end whenever the four data packets were jammed and the first DSR packet was sensed. By allowing the DSR Route Discovery mechanism to complete, the victim network can immediately re-establish communication and we can observe the delay caused by simply jamming four consecutive data packets. The network

disruption we observed was approximately 800ms for simply jamming the four data packets. The network behavior is plotted in Figure 5. From this point, with the DSR Route Discovery mechanism invoked, the intelligent jammer could even possibly engage in jamming DSR RREQ packets, similar to the simulations ran in 4 to introduce even more network delay.

Controlling a victim network is useful because with the ability to invoke the DSR Route Discovery mechanism, it allows our classifier to start recording packets from the beginning of a flow and ensure accurate packet and sequence classification. Additionally, it also allows the classifier to observe rare packet sequences. By invoking DSR Route Discovery, we reset the network and were able to observe both the DSR RREQ-REPLY and TCP Startup sequences, which can potentially provide an online classifier with more accurate training data.

7.2 Lomb Periodogram

In Section 5 we discussed that our improved classification algorithm was capable of dynamically estimating the RTT value for the TCP DATA-ACK sequence. However, there may be traffic scenarios in which the congestion window varies so much over time that an average estimation over the entire histogram is not an accurate portrayal of RTT for the DATA-ACK sequence. Thus, our classifier must be able to change its perception of the TCP DATA-ACK inter-arrival timing over time. One such solution would be to utilize a sliding window mechanism, in which the RTT value was constantly updated by only computing the estimate over say the last 10 packets, for example. A sliding window mechanism could be implemented using the data pool concept that we already utilize in our algorithm. However, the pool may become unreliable in situations where the window slides faster than it is able to detect DATA-ACK sequences. As a result, a more reliable method of obtaining a sliding RTT value needs to be implemented.

Fortunately, we can make use of techniques already developed for other applications. For example, [7] uses signal processing techniques to analyze traffic and determine transmission timing intervals and round trip times in wireless networks. The author in [7] utilizes a technique known as the Lomb Periodogram [5], which is a signal processing function capable of identifying significant frequency patterns within signals whose samples are unequally distributed. The author in [7] notes that in some experiments, significant peaks that occur in the Lomb Periodogram result correspond to the transmission intervals for UDP flows and round trip time for TCP flow. In other experiments, however, the significant peaks instead correspond to transmission intervals for TCP.

We conducted experiments with the Lomb Periodogram using the network model for this paper for maximum TCP congestion window sizes ranging from 1 to 5. We found that the Lomb Periodogram actually identified the TCP transmission intervals, as evidenced by the close correlation between the actual transmission intervals and the Lomb Periodogram result from Table 1.

Recall that with a TCP congestion window of one, a source must wait for a previously sent data packet to arrive before transmitting the next packet. This forced waiting behavior creates a base case in which the transmission interval

Table 1. Lomb periodogram experiment results

ns-2 Avg RTT	Avg TX Interval	Lomb Result	Max TCP Congestion Window
5.1ms	5.83ms	5.88ms	1
12.9ms	5.74ms	5.77ms	2
15.8ms	5.72ms	5.85ms	3
23.3ms	5.73ms	5.86ms	4
27.5ms	5.73ms	5.97ms	5

and round trip time converge to similar values. We are particularly interested in this case because it provides a robust approach to dynamically calculating RTT, despite the requirement of a single packet congestion window. A single packet window, however, can be achieved utilizing active analysis. By instructing an intelligent jammer to jam specific TCP packets and trick senders into reducing their send rates, the Lomb Periodogram could be very powerful in establishing an accurate round trip time threshold over time.

8 Conclusion

Automatic sensing and classification of packets is critical for supporting intelligent jamming in wireless ad hoc networks with encrypted traffic. In this paper, we have studied the impact of the classification accuracy on the performance of an intelligent jammer in a DSR/TCP ad hoc network. We found that in order to facilitate such intelligent jamming, we must provide the jammer a classifier that is at least 95% accurate in the classification of encrypted packets. We then developed an offline classifier that utilizes the metrics of size, inter-arrival timing, sequence, and direction in the process of identifying packet types and sequences. While our classifier is capable of handling a fluctuating congestion window and does not rely on a seed delay value for the DATA-ACK packet sequence, it still relies on seed delay values for the other sequences. We also utilize active analysis to provide our classifier with a robust DATA-ACK delay estimator, the Lomb Periodogram.

We are currently extending our work in various directions. We are investigating alternative approaches to the probabilistic modeling of the inter-arrival delays of packet sequences. Furthermore, we are looking into the development and application of other filters to refine our classification process. Our long-term goal is to extend our work to include other ad hoc protocols and traffic types.

References

1. Brown, T.X., James, J.E., Sethi, A.: Jamming and sensing of encrypted wireless ad hoc networks. In: MobiHoc 2006: Proceedings of the 7th ACM international symposium on Mobile ad hoc networking and computing, pp. 120–130. ACM, New York (2006)

2. Dempsey, T., Sahin, G., Jade Morton, Y.T.: Intelligent sensing and classification in DSR-Based ad hoc networks. IEICE Transactions on Information and Systems, special section on Information and Communication System Security E92-D(5), 818–825 (2009)
3. James, J.E.: Intelligent sensing of encrypted packets in wireless networks. Master's thesis, University of Colorado (2005)
4. Johnson, D.B., Maltz, D.A.: Dynamic source routing in ad hoc wireless networks. In: Mobile Computing, pp. 153–181. Kluwer Academic Publishers, Dordrecht (1996)
5. Lomb, N.R.: Least-squares frequency analysis of unequally spaced data. Astrophysics and Space Science, 447–462 (1976)
6. The Network Simulator ns-2 (October 2001), `http://www.isi.edu/nsnam/ns/`
7. Partridge, C., Cousins, D., Jackson, A.W., Krishnan, R., Saxena, T., Timothy Strayer, W.: Using signal processing to analyze wireless data traffic. In: WiSE 2002: Proceedings of the 1st ACM workshop on Wireless security, pp. 67–76. ACM, New York (2002)
8. Wright, C., Monrose, F., Masson, G.M.: Hmm profiles for network traffic classification. In: VizSEC/DMSEC 2004: Proceedings of the 2004 ACM workshop on Visualization and data mining for computer security, pp. 9–15. ACM, New York (2004)