Enabling Accurate Anomaly Detection Using Progressive Security-Aware Packet Sampling

By
Sardar Ali
2007-NUST-MS-PhD IT-19

Supervisor
Dr. Syed Ali Khayam
NUST-SEECS

A thesis submitted in partial fulfillment of the requirements for the degree of Masters of Science in Information Technology (MS IT)

In
School of Electrical Engineering and Computer Science,
National University of Sciences and Technology (NUST),
Islamabad, Pakistan.

(November 2009)

Abstract

Real-time Anomaly Detection Systems (ADSs) use packet sampling to realize traffic analysis at wire speeds. While recent studies have shown that a considerable loss of anomaly detection accuracy is incurred due to sampling, solutions to mitigate this loss are largely unexplored. In this thesis, we propose a Progressive Security-Aware Packet Sampling (PSAS) algorithm which enables a real-time inline anomaly detector to achieve higher accuracy by sampling larger volumes of malicious traffic than random sampling, while adhering to a given sampling budget. High malicious sampling rates are achieved by deploying inline ADSs progressively on a packet's path. Each ADS encodes a binary score (malicious or benign) of a sampled packet into the packet before forwarding it to the next hop node. The next hop node then samples packets marked as malicious with a higher probability. We analytically prove that under certain realistic conditions, irrespective of the intrusion detection algorithm used to formulate the packet score, PSAS always provides higher malicious packet sampling rates. To empirically evaluate the proposed PSAS algorithm, we simultaneously collect an Internet traffic dataset containing DoS and portscan attacks. Experimental results using four existing anomaly detectors show that PSAS, while having no extra communication overhead and extremely low complexity, allows these detectors to achieve significantly higher accuracies than those operating on random packet samples.

Certificate of Originality

I hereby declare that this submission is my own work and to the best of my knowledge it contains no materials previously published or written by another person, nor material which to a substantial extent has been accepted for the award of any degree or diploma at NUST SEECS or at any other educational institute, except where due acknowledgement has been made in the thesis. Any contribution made to the research by others, with whom I have worked at NUST SEECS or elsewhere, is explicitly acknowledged in the thesis.

I also declare that the intellectual content of this thesis is the product of my own work, except for the assistance from others in the project's design and conception or in style, presentation and linguistics which has been acknowledged.

Author Name: **Sardar Ali**

Signature: ________________

Acknowledgments

First and foremost, I am immensely thankful to Almighty Allah for letting me pursue and fulfill my dreams. Nothing could have been possible without His blessings.

I would like to thank my parents for their support throughout my educational career, specially in the last year of my Master degree. They have always supported and encouraged me to do my best in all matters of life. I dedicate this work to them. I would also like to thank my one and only one sister Seema, for her love and prayers for the successful completion of this work; and to Najma for her unending love and support.

My heartfelt thanks to my committee members, project members Irfan, Sajjad, Unum and Naurin; and to everyone at WISNET Lab, and all others who contributed in any way towards the successful completion of this thesis.

Finally, this thesis would not have been possible without the expert guidance of my advisor, Dr. Syed Ali Khayam, who has been a great source of inspiration for me during these years of research. Despite all the assistance provided by Dr. Ali Khayam and others, I alone remain responsible for any errors or omissions which may unwittingly remain.

Sardar Ali

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction and Motivation

1.1 Introduction

The last few years have witnessed an exponential increase in the volume and sophistication of network attacks. To combat these rapidly evolving attacks, design of accurate Anomaly Detection Systems (ADSs), which can detect *zero-day* (previously unseen) attacks, has received significant attention with commercial ADSs now experiencing widespread deployments. In view of the unprecedented traffic volumes observed on contemporary enterprise networks and due in part to the stringent memory and complexity constraints of network devices, it is not possible for a real-time ADS to examine every packet in detail. Packet and flow sampling, originally proposed for network monitoring applications, are now being used to reduce the amount of data to be analyzed by a real-time ADS [1, 2]. Commercial ADS products are integrating sampling and anomaly detection algorithms in the routing fabric in order to achieve high-speed and truly-inline anomaly detection in real-time [3]–[6].

Packet sampling is an inherently lossy process which provides an incomplete and biased approximation of the underlying traffic. While minimization of estimation error on flow statistics is well-investigated [7]–[10], there have only been a handful of studies on the impact of packet sampling on anomaly detection [11]–[14]. While these studies unanimously agree that packet sampling can introduce significant accuracy degradations in an ADS, solutions to mitigate this accuracy loss are largely unexplored in research literature. The seminal paper in this domain concluded that [12]: *"anomaly detection algorithms can be improved under sampling if the information loss and distortions is compensated or better avoided. Another relevant open question is whether correlating sampled traces from multiple vantage points could improve the anomaly detection process at relatively low sampling rates, hence avoiding the need for detailed packet trace collection."*

In this thesis, we propose a solution to simultaneously address these open problems by enabling an inline ADS to achieve higher accuracy under sampling by correlating traffic from different points of deployment in a network. Specifically, as opposed to prior studies which spatially distribute ADSs in a network [15, 16], we propose that ADSs are deployed progressively on nodes on a packet's path. We then allow these ADSs to communicate with each other by encoding their binary score (malicious or benign) of the packet inside the packet's header before forwarding it to the next hop node. The ADS operating at the next hop uses this score as side information for packet sampling and anomaly detection. This binary side information can be effortlessly encoded inside IP packets, thus allowing different nodes to collaborate without any additional communication overhead.

We show that this simple collaboration model, referred to as Progressive Security-Aware Sampling (PSAS), enables inline anomaly detectors to achieve significantly higher accuracies by mitigating the information loss under sampling. First, we empirically show that, for a fixed sampling budget, an increase in the amount of malicious traffic in the sampled subset induces at least a linear, and mostly a much faster than linear, improvement in an ADS' accuracy. To achieve these accuracy dividends, we propose the PSAS algorithm which samples packets marked as malicious with higher probabilities, while adhering to a given packet sampling budget. We analytically compare security-aware and random sampling for a fixed sampling budget. This comparison reveals that, regardless of the ADS algorithms employed by each node, PSAS samples considerably more malicious packets at each node than blind random sampling.

To evaluate accuracy dividends and complexity of the proposed PSAS algorithm, we collect a labeled dataset of Internet attack traffic at three different points of deployment; these attacks include DoS and portscan attacks launched at varying rates. Using this dataset, we input randomly sampled and security-aware sampled traffic to four existing anomaly detectors [17]–[20]. ROC-based performance evaluation substantiates that security-aware traffic samples enable the anomaly detectors to consistently achieve significantly higher accuracies than random packet samples. These accuracy improvements are sustained for both low and high rate attacks. Moreover, we show that, in addition to having no additional communication overhead and memory requirements, PSAS' run-time complexity is comparable to random sampling.

1.2 Background and Motivation

Most network anomalies tend to persist over time and are detected by performing sophisticated statistical analysis on a time-series of network parameters. In this context, packet and flow sampling techniques can have a serious adverse affect on the accuracy of the ADSs that are operating on the sampled traffic. Mai et al. [11] evaluated the impact of packet sampling on three portscan detection algorithms and concluded that packet sampling is an inherently lossy process which provides an incomplete and biased approximation of the underlying traffic. This work was extended in [12] and the affect of sampling was analyzed using four popular sampling techniques; random packet sampling, random flow sampling, sample-and-hold [21], and smart sampling [22]. Three anomaly detection techniques were used to cover broad categories of volumetric and portscan anomaly detection, namely the wavelet analysis approach [23], Threshold Random Walk (TRW) [24] and Time Access Pattern Scheme (TAPS) [25]. Results showed that random packet sampling, sample and hold, and smart sampling adversely affect both volumetric and portscan-based anomaly detectors. Similarly, it was shown in [13] that the accuracy of an ADS is dependent on the rate of sampling when flow based metrics are used. Brauckhoff et al. [14] analyzed the volume and feature entropy metrics and showed that packet sampling does not have much impact on volumetric packet counts but can introduce significant bias in flow counts. Feature entropies are also disturbed but the traffic pattern is generally visible. The biased and incomplete traffic captured by a packet sampler when input to an anomaly detector induces an undesirable loss of accuracy, thereby compromising the purpose for which the traffic was being sampled. Intuitively, an ADS operating on sampled traffic would want to operate on as much malicious data as possible. Therefore, instead of the security-unaware or blind packet/flow samplers, we need to design security-aware packet sampling algorithms.

1.3 Contribution

To the best of the authors' knowledge, this thesis proposes the first known solution to mitigate sampling-induced accuracy loss in an anomaly detection system. PSAS sampling is efficient, having no communication overhead and low complexity. We also showed that the sampling-induced accuracy degradation in an ADS can be significantly reduced by PSAS, with promising avenues for further research in this area. Problem statement and its breakdown is as under.

1.3.1 Problem Statement

The problem statement of our research thesis is:

> *"To devise a security-aware packet sampling algorithm which addresses these open problems by enabling an inline ADS to achieve higher accuracy under sampling by correlating traffic from different points of deployment in a network"*

1.3.2 Problem Breakdown

The specific objectives of this project are:

- *Dataset Collection:* While there exist a few public and labeled traffic attack datasets [29]–[32], these datasets do not satisfy our requirements (see chapter 3). Therefore, we collect our own traffic dataset. For repeatable performance evaluation, our labeled dataset is publicly available at `http://wisnet.seecs.edu.pk/datasets/`.

- *Implementation of Exising Sampling Algorithms:* In order to evaluate our proposed packet sampling algorithm, we need to implement the existing sampling algorithms.

- *Design and implementation of PSAS algorithm:* The objective of this step is to design a Progressive Security-Aware Packet Sampling (PSAS) algorithm which enables a real-time inline anomaly detector to achieve higher accuracy by sampling larger volumes of malicious traffic than random sampling, while adhering to a given sampling budget.

- *Performance Evaluation:* To analytically compare the performance of the proposed packet sampling algorithm with existing sampling algorithms; and to experimentally compare and evaluate the impact of the proposed packet sampling algorithm on existing anomaly detection algorithms.

1.4 Thesis Organization

The remainder of this thesis is structured as follows:

Chapter 2 provides discussion on some of the existing packet sampling techniques, anomaly detection systems used in the experiments, ADS accuracy criteria, and related work in the subjected domain. The impact of existing sampling techniques on anomaly detection systems is also discussed.

Chapter 3 is dedicated to the data collection activity. Unique characteristics of the newly collected dataset, experimental setup of the data collection process, detail of types of rates used to generate the attack traffic, and dataset's statistics are detailed in this chapter.

In Chapter 4, outlines the impact of *increasing ratio of malicious traffic* on ADS accuracy, design constraints of a practical security-aware packet sampler, the proposed progressive security-aware packet sampling algorithm, and analytical comparison of random and our proposed PSAS algorithm.

Experimental performance evaluation of the proposed security-aware sampling algorithm under averaged and varying attack rates, and PSAS' complexity measures are provided in Chapter 5.

Limitations of the proposed security-aware packet sampler and its countermeasures are discussed in Chapter 6. Chapter 7 summarizes key conclusions of this thesis.

Chapter 2

Literature Review

This chapter provides the background literature review of prominent anomaly detection systems (ADSs) and existing sampling techniques. We review four prominent anomaly detection systems (ADSs) and existing sampling techniques. These ADSs and sampling techniques are detailed in subsequent sections.

2.1 Overview of Anomaly Detection Systems

Since it is not possible to evaluate all existing ADSs, we selected the following four ADSs for this study: Maximum Entropy Anomaly Detector [17]; Credit-Based Threshold Random Walk (TRW) Anomaly Detector [18]; Packet Header Anomaly Detector (PHAD) [19]; and Network Traffic Anomaly Detector (NETAD) [20]. The two main rationales for choosing these ADSs were:

1. *Diversity in Accuracy*: These detectors have been shown to provide varying accuracies at different points of deployment [32];

2. *Diversity in Detection Principles and Features:* These ADSs use different traffic features and detection principles and operate at different traffic granularities.

The rest of this section briefly summarizes the basic detection principles of these anomaly detectors. Interested readers are referred to the original papers [17, 18, 19, 20] for details description of each detector.

2.1.1 Maximum Entropy Anomaly Detector

[17]: This detector computes real-time ADS scores of various classes of network traffic based on a baseline benign traffic distribution. An alarm is raised if a packet class's ADS score repeatedly exceeds a fixed threshold a certain number of times. We varied this threshold of obtain accuracy points on the ROC plane. To identify maliciousness at the packet level, we identified the packet classes which exceeded the detection threshold in a time-window and then marked all packets belonging to that class as malicious.

2.1.2 Credit-Based Threshold Random Walk (TRW) Algorithm

[24, 18]: The original TRW algorithm [24] computes an ADS score by applying the sequential hypothesis on a remote host's connection attempts. This ADS score is thresholded to determine whether or not a remote host is a scanner. TRW-CB [18] is a hybrid solution, leveraging the complementary strengths of Rate Limiting and TRW. A credit increase/decrease algorithm is used to slow down hosts that are experiencing unsuccessful connections. We generate ROCs for TRW-CB by varying its upper and lower hypothesis testing thresholds.

2.1.3 Packet Header Anomaly Detector (PHAD)

[19]: PHAD learns the normal range of values for all 33 fields in the Ethernet, IP, TCP, UDP and ICMP headers. An anomaly

score is assigned to each packet header field in the testing phase and the fields' scores are summed to obtain a packet's aggregate anomaly score. We evaluate PHAD-C32 [19] using the following packet header fields: source IP, destination IP, source port, destination port, protocol type and TCP flags. The top n values are thresholded as anomalous. The value of n is varied to generate ROCs.

2.1.4 Network Traffic Anomaly Detector (NETAD)

[20]: NETAD detects incoming IP traffic anomalies and operates on the first 48 bytes of a packet including header in a modeled subset. It computes a packet score depending on the time and frequency of each byte of packet in the modeled subset. All packets exceeding a certain threshold are marked as anomalous. For our performance evaluation, we operated NETAD in the reverse (outgoing) direction. As with PHAD, the top n values are thresholded as anomalous.

2.2 Sampling Techniques

Many packet sampling techniques have been proposed over the last few years. Due to computation and memory constraints, these techniques use different attributes to estimate the traffic. At a high-level, these techniques either sample traffic on the basis of complete packet contents or maintain flow-level information.

In this section, we review four prominent sampling techniques; detail of each of these is as under:

2.2.1 Random Packet Sampling

In random packet sampling, the selection of packets is triggered in accordance to a random process. It can either be count-based or probabilistic. In count based n samples are selected out of N packets, hence it is sometimes called n-out-of-N sampling. For this sampling schema each packet has an equal chance of being drawn. One way of achieving a simple random sample is to randomly generate n different numbers in the range of 1 to N and then choose all packets with these positions. This procedure is repeated for every N packets. For this kind of sampling the sample size is fixed. In probabilistic sampling samples are chosen in accordance to a pre-defined selection probability. The sample size can be different for consecutive intervals. Random packet sampling simply samples a packet with a small probability $r < 1$.

2.2.2 Random Flow Sampling

Random flow sampling first classifies packets into flows based on the five-tuple: (Source IP address, destination IP address, source port, destination port, protocol). The sampler then samples each flow with some probability $p < 1$.

2.2.3 Sample and Hold

Sample and Hold [21] is similar to ordinary sampling such that each packet is sampled with a probability $h * s < 1$, where $h * s$is chosen as if each byte is sampled with a probability h. The probability that a byte would not be sampled is $1 - h$. The packet is dropped if all of its bytes are not selected. Thus the sampling probability for a packet of size s is given by:

$$h * s = 1 - (1 - h)^s$$

If a packet is sampled and the flow it belongs to has no entry in the flow memory, a new entry is created. However, after an entry is created for a flow, unlike in simple sampling, the entry for every subsequent packet belonging to the flow is updated.

2.2.4 Smart Sampling

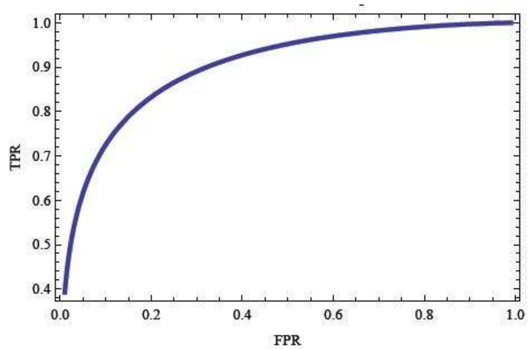
This is a size-dependent flow record selection algorithm [22] and applies to complete flow records. Given a set of flows of sizes $S = \{x_i : i = 1, 2, ..., n\}$, smart sampling selects a flow of size x with a probability $p(x)$ to form a set of selected flows of S'. The goal is to achieve an unbiased estimator of the total byte count. The following solution was shown to be optimal in terms of balancing the opposing constraints of keeping the variance of X' small, while reducing the sample size $N' = |S'|$:

$$f(x_i) = f_z(x_i) = \begin{cases} \dfrac{x_i}{z} \text{ if } x_i < z \\ 1 \text{ if } x_i \geq z \end{cases}$$

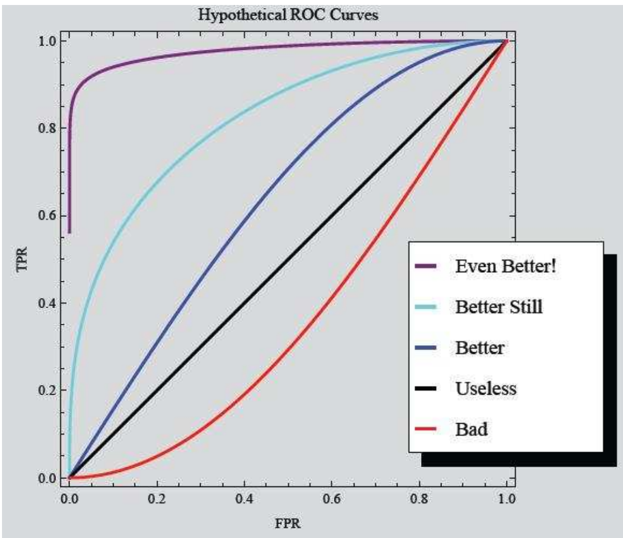
where z is a threshold that trades off accuracy for reduction in bandwidth requirement.

2.2.5 Discussion on Sampling Techniques

We emphasize that the sampling techniques described above are designed to provide a sampled dataset that is representative of the overall traffic behavior. The biased and incomplete traffic captured by a packet sampler when input to an anomaly detector causes loss of accuracy, thereby compromising the purpose for which the traffic was being sampled. Intuitively, an ADS operating on sampled traffic would want to operate on as much malicious data as possible. Therefore, instead of the security-unaware or blind packet/flow samplers, we need to design security-aware packet sampling algorithms.



(a) ROC Curve



(b) ROC Curve Comparison

Figure 2.1: A typical ROC curve; comparison of different ROCs.

2.3 ADS Accuracy Criteria

The accuracy of an intrusion detection system is generally evaluated on two competing criteria:

1. *Detection rate:* What fraction of anomalies are correctly detected by the IDS.

2. *False Alarm rate:* What fraction of the total anomalies detected by the IDS are in fact benign data.

To understand the tradeoff between these accuracy criteria, consider an IDS that classifies all the test data as anomalous. Such an IDS will achieve 100% detection rate, but at the cost of an unacceptable 100% false alarm rate. At the other end of this spectrum, consider an IDS that classifies all of the test data as normal. This IDS will have an attractive 0% false alarm rate, but is useless because it does not detect any anomalies. To evaluate the accuracy of an IDS, detection thresholds of the IDS are tuned and for each threshold value the detection rate is plotted against the false alarm rate. Each point on such a plot, referred to as an ROC curve [33], represents performance results for one configuration (or threshold value) whereas the curve represents the behavior for the complete set of configurations.

A receiver operating characteristics (ROC) curve is a technique for visualizing, organizing and selecting classifiers based on their performance [34]–[35]. A typical ROC curve is shown in Fig. 2.1 (a). The ROC curve is the plot of TPR (true positive rate) vs. FPR (false positive rate) for different threshold values.

The diagonal line y = x represents the strategy of random guessing. For example, if a classifier randomly guesses the positive class half the time, it can be expected to get half the positives and half the negatives correct; this yields the point (0.5, 0.5) in ROC space.

Any classifier that appears in the lower right triangle performs worse than random guessing. This triangle is therefore usually empty in ROC graphs.

What is of interest are the curves in the upper left triangle. Higher the curve, better the performance of the classifier. This is shown in Figure 2.1 (b).

2.4 Related Work

Most network anomalies tend to persist over time and are detected by performing sophisticated statistical analysis on a time-series of network parameters. In this context, packet and flow sampling techniques can have a serious adverse affect on the accuracy of the ADSs that are operating on the sampled traffic. Mai et al. [11] evaluated the impact of packet sampling on three portscan detection algorithms and concluded that packet sampling is an inherently lossy process which provides an incomplete and biased approximation of the underlying traffic. This work was extended in [12] and the affect of sampling was analyzed using four popular sampling techniques; random packet sampling, random flow sampling, sample-and-hold [21], and smart sampling [22]. Three anomaly detection techniques were used to cover broad categories of volumetric and portscan anomaly detection, namely the wavelet analysis approach [23], Threshold Random Walk (TRW) [24] and Time Access Pattern Scheme (TAPS) [25]. Results showed that random packet sampling, sample and hold, and smart sampling adversely affect both volumetric and portscan-based anomaly detectors. Similarly, it was shown in [13] that the accuracy of an ADS is dependent on the rate of sampling when flow based metrics are used. Brauckhoff et al. [14] analyzed the volume and feature entropy metrics and showed that packet sampling does not have much impact on volumetric packet counts but can introduce significant bias in flow counts. Feature entropies are also disturbed but the traffic pattern is generally visible.

A common approach followed by existing work is to use multiple ADSs on a single hop [26]–[28]. However, if the sampled subset of traffic is not representative of the overall traffic trends (e.g., does not contain enough malicious packets), then adding more ADSs on the same node will not increase an improvement in accuracy. Our focus on this work is to improve the sampled subset to facilitate the ADS deployed on a node. Hence, the multiple ADSs based detectors can also benefit from using a security-aware sampler.

In view of the above discussion, we concern ourselves with security-aware packet sampling for an inline and real-time ADS. Due to our focus on real-time anomaly detection (which is typically integrated with the routing fabric), we do not consider flow sampling algorithms in this work.[1]

[1] While some recently-proposed real-time flow sampling algorithms [36], [37] can also benefit from the proposed PSAS algorithm, we do not consider them in this thesis because they will introduce undesirable communication overhead between communicating nodes.

Chapter 3

Attack Traffic Dataset

For the present research problem, we needed a traffic dataset that meets the following requirements:

1. Attack traffic is captured as it passes through different points in a network;

2. At each deployment point, benign (background) and attack data had to be labeled accurately to allow judicious evaluation of the impact of sampling on ADS accuracy;

3. For comprehensive performance evaluation, we needed attacks of different types (DoS, portscan, etc.) and rates;

4. For repeatable performance benchmarking by future studies, the dataset had to be publicly available; and

5. To cater for different types of ADSs and attacks (present and future), the dataset should contain different types (ICMP, TCP, UDP, etc.) of packets with full (header+payload) packet information.

While there exist a few public and labeled traffic attack datasets [29, 30, 31, 32], these datasets do not satisfy the requirements set above. Therefore, we collected our own traffic dataset and the rest of this section explains our data collection experiment and some preliminary data statistics. For repeatable performance evaluation, our labeled dataset is publicly available at `http://wisnet.seecs.edu.pk/datasets/`.

Perhaps the most unique requirement of our study is simultaneous data collection at different deployment points. Note that as we move from endpoints towards an enterprise's network perimeter, the scope of responsibility of a network entity, in terms of traffic volume and number of network nodes generating that traffic, increases accordingly. We conducted our experiment at three progressive points of deployment in our school's network: Endpoints, Research Lab Router and Research Wing Router. As can be intuitively deduced, each one of these deployment points had a very different traffic scope in terms of traffic volume and number of nodes. We now explain data collection at each of these deployment points.

3.1 Endpoint Traffic

3.1.1 Endpoint Background Traffic

Before the attacks were launched, some background (benign) data had to be collected at each network entity in order to train our algorithms under normal circumstances. At the endpoint, this background dataset was collected at three lab computers with human users (research students) in our research lab. Background data were logged during six separate periods, each one of over three hours duration, for an aggregate of approximately nineteen hours. More specifically, traffic was collected on six separate days during peak hours of Internet activity.

Different types of activities were taking place on these systems, including: peer-to-peer file sharing, software downloading from remote servers, web browsing, real-time video streaming, etc. Therefore, a considerable amount of background traffic was generated by the applications running on these endpoints. Such high, yet realistic, background traffic was introduced so that the attack traffic mixes up with benign data and does not stand out. During the onset of an attack, this background traffic remained uninterrupted.

3.1.2 Endpoint Attack Traffic

The three endpoints described earlier were scheduled to simultaneously launch each attack. The motivation behind this attack scenario was to emulate a botnet or localized scanning scenario in which a pool of comprised hosts exist in a network.

We launch TCP, UDP, and ICMP based attacks since a majority of contemporary attacks are launched using these protocols. All the three attacking machines started their transmission simultaneously and each exploit was launched for a period of five minutes. A total of six attacks comprising three portscan attacks and an equal number of DoS attacks were launched on servers setup outside our network. The exploits involved in the former are ICMP Path MTU Discovery, ICMP Protocol Unreachable (Blind Connection Reset) and TCP-SYN portscans. DoS attacks included in our study are TCP flood, UDP flood (fraggle) and ICMP echo ping flood (smurf) attacks; readers are referred to [38]–[40] for details of each attack.

The source IPs were spoofed for all (TCP, UDP and ICMP) DoS attacks, while servers setup at two different public IPs were attacked. Ports 1433, 22, 138, 137, 21 were attacked in the TCP-SYN Flood, while ports 22, 80, 135, 1433 were targeted with UDP floods. For each TCP portscan experiment, two distinct attacks were launched, first on port 80 and then on port 135. TCP portscan packets had fixed source IPs while the probed destination IPs were generated randomly. ICMP scans were also sent to randomly-generated IP addresses.

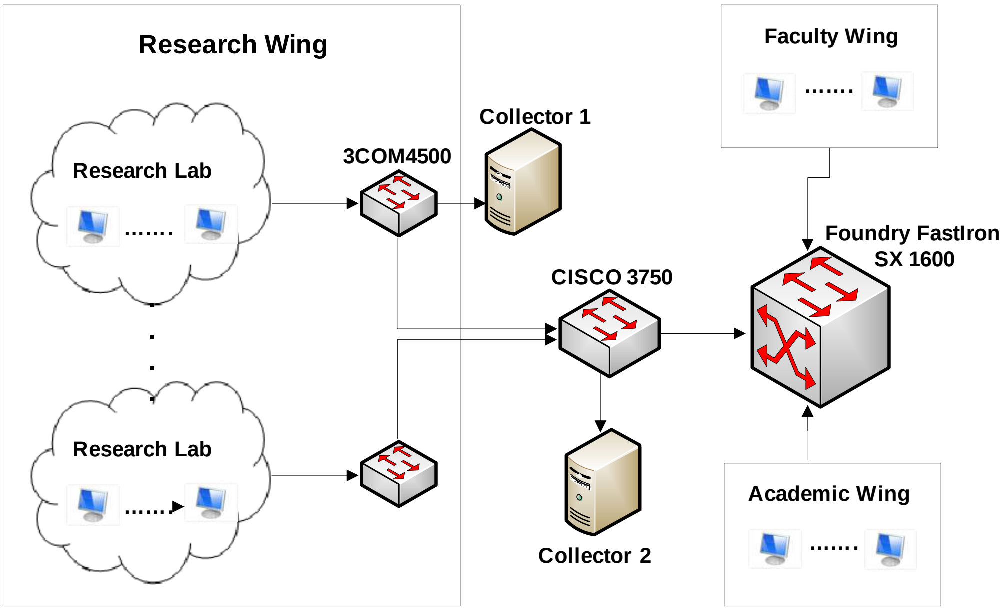


Figure 3.1: Network diagram of the data collection setup.

To label the attack traffic, we needed to embed an attack marker inside each malicious packet. In order to maintain the labeling across different packet hops, the packet marker had to be embedded in a protocol field that does not change at each hop. To this end, we set the reserved flag in the IP header of each attack packet using raw socket; this bit is unused and generally a default value of zero is used for it.

The attacks were launched at five different rates. The rates were progressively increased to launch the attacks over a range of values (0.1, 1.0, 10, 100, and 1000 pkts/sec). For the portscan attacks, the slow rates are ideal as hackers can avoid detection using very low rates while the damage caused by DoS attacks is more prominent at high rates. The range of rates ensured the comprehensive results from both these types of attacks. It also enabled us to check the robustness of our algorithms in identifying the low-rate attacks besides the higher rate (more obvious) ones.

Table 3.1: Background Traffic Information

Date	Background Traffic Statistics					
	Endpoints		Lab Router		Wing Router	
	μ	σ	μ	σ	μ	σ
	pkts/sec	pkts/sec	pkts/sec	pkts/sec	pkts/sec	pkts/sec
03/07/2009	191.4	136.6	1388.4	1256.2	3440.5	1614.3
03/09/2009	312.5	173.4	875.7	664.7	2751.2	1740.9
03/12/2009	562.8	210.2	1324.9	328.9	3445.5	1683.5
03/18/2009	643.5	209.7	1249.0	338.7	2988.0	1513.5
03/19/2009	495.9	173.3	1146.1	757.8	2939.5	1550.6
04/02/2009	416.7	191.0	1029.2	682.8	3168.8	1785.2

3.2 Lab and Research Wing Routers' Background and Attack Traffic

All the research labs in the School of Electrical Engineering & Computer Science (SEECS), NUST [`www.seecs.edu.pk`] are located in three distinct research wings. Traffic from each lab is routed by a 3Com 4500G switch. Traffic from all the lab routers is relayed to a research wing router (Cisco 3750) using fibre connections. These wing routers are in turn connected to a distribution router that handles traffic from the entire school. This network topology is shown in Fig.3.1

Due to privacy constraints, we were not allowed to log traffic at the distribution router. Therefore, we setup our traffic collection at the first and second hops. At the first hop, a port was mirrored on our research lab's router to receive the entire lab's traffic (inbound, outbound and internally routed). The lab contains 28 computers running different operating systems (Windows XP/Vista and Linux), applications and services. As mentioned earlier, three of these computers were used to generate attack data while the remaining computers served as background traffic sources. At the second hop, we collected traffic by mirroring a port on the router that manages traffic for our

Table 3.2: Background Traffic Information During Attacks

Attack Name	Attack Rate (pkts/sec)	Background Traffic Statistics at Attack Time (pkts/sec)					
		Endpoints		Lab Router		Wing Router	
		μ	σ	μ	σ	μ	σ
ICMP Protocol Unreachable Portscans	0.1	151.4	115.3	897.8	440.6	2646.6	578.2
	1	201.8	114.3	963.6	140.2	2948.4	371.1
	10	202.2	59.1	1211.2	309.2	3305.2	340.7
	100	264.8	117.5	2310.9	907.5	6495.8	3075.6
	1000	212.0	50.9	937.7	144.0	3082.1	279.3
ICMP Path MTU Discovery Portscans	0.1	360.9	96.0	809.3	144.8	2255.7	341.8
	1	377.1	166.8	1037.5	417.0	5381.6	2409.0
	10	394.9	84.3	962.7	151.9	3242.6	550.5
	100	377.0	93.4	515.1	224.0	1893.9	255.6
	1000	430.8	83.1	899.6	105.1	3242.5	222.1
TCP-SYN portscans	0.1	576.7	94.5	1184.6	138.2	2462.9	474.4
	1	549.4	146.8	1487.2	265.0	3002.6	398.0
	10	534.0	81.9	1645.5	180.9	3325.2	397.7
	100	555.5	67.3	1244.6	188.1	6100.0	2492.4
	1000	698.8	96.3	1253.9	138.4	3084.7	247.4
ICMP echo ping flood (DoS)	0.1	478.2	59.4	943.2	96.6	2021.9	184.3
	1	452.7	76.7	1024.7	103.3	2466.8	272.6
	10	786.2	75.5	1616.3	150.8	4318.5	1790.1
	100	819.4	82.9	1438.1	141.2	5565.0	2493.8
	1000	639.2	119.7	1191.4	124.4	3128.4	245.2
TCP-SYN flood (DoS)	0.1	354.3	52.9	781.2	109.8	2240.1	216.7
	1	504.6	62.6	1175.5	142.7	2699.1	328.8
	10	724.6	118.2	2734.3	1777.2	4409.8	1666.2
	100	471.9	90.5	1031.7	123.1	3964.1	1670.4
	1000	426.0	59.2	980.4	106.8	3000.9	238.0
UDP flood fraggle	0.1	323.5	48.7	693.7	108.2	2025.8	506.4
	1	300.1	61.7	907.4	113.7	2479.1	291.0
	10	421.3	54.7	2261.8	1847.1	4028.4	1893.1
	100	494.2	66.7	1151.9	157.6	6565.7	3006.9
	1000	578.7	62.3	1069.7	111.5	2883.7	260.8

research wing. This router handled traffic from approximately 50 hosts. Again, 3 hosts were generating attack traffic, while the remaining hosts served as background traffic sources.

Since our attacks consisted of only TCP, UDP and ICMP packets, we filtered packets other protocols from the traffic capture. Also, as the attack victims were setup outside of our network, only outbound traffic was retained for analysis.

3.3 Preliminary Traffic Statistics

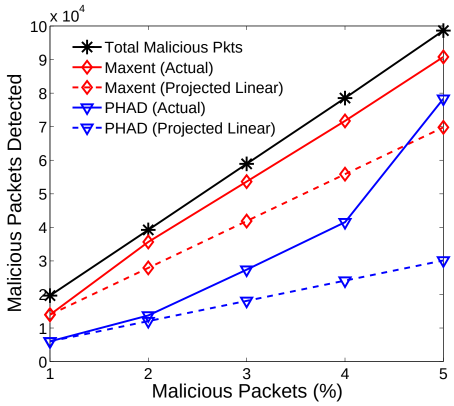
Tables 3.1 and 3.2 show the background and attack traffic statistics. From the endpoint data statistics (Table 3.1), it can be observed that the mean traffic rate is fairly high on all three endpoints. This is mainly because of the peer-to-peer file sharing activity on these hosts. Also, note that there is a large variance around the mean which was observed because of the bursty video streaming applications.

Table 3.2 shows the diversity of the collected attack dataset. At the endpoints, the background traffic rate during low rate attacks (0.1 and 1 attack packets/sec) is two or three orders of magnitude greater than the attack rate. On the other hand, background traffic rate is comparable to or less than the high rate attacks (1000 packets/sec.) At the Lab router, the low-rate attacks are further diminished by large volumes of background traffic. However, the high rate attacks still comprise a considerable fraction of the total traffic even at the Lab router. At the Wing router, the high rate attacks have two to three times less rate than the background traffic and therefore do not dominate the total traffic. Based on this attack and background traffic rate diversity, we expect that detection will become more and more difficult as we move from the endpoints to the Wing router mainly because the attack traffic will mix with considerable volumes of background traffic.

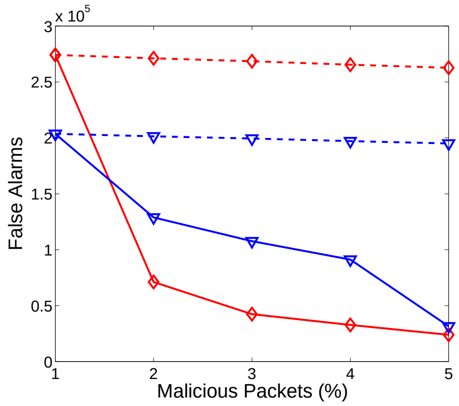
Chapter 4

Security-Aware Packet Sampling

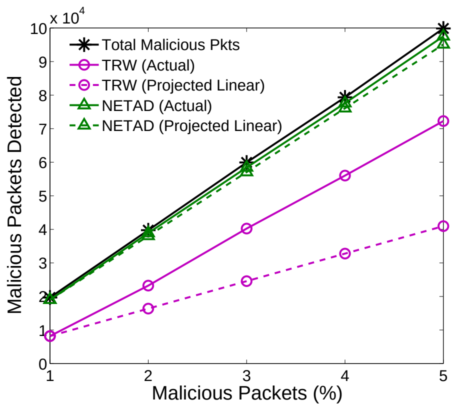
Prior studies have shown that detection rate of an anomaly detector degrades with a decrease in the rate of sampling [12]–[14]. While these studies comprehensively evaluated sampling-induced accuracy loss in ADSs, solutions to mitigate this loss have not been investigated so far. An important question that is still unanswered in this regard is: *For a given and fixed sampling budget, would an ADS' accuracy improve if we can somehow sample a larger fraction of malicious traffic?* If this question is answered in affirmative, another resultant question is: *How much improvement in accuracy should we expect with such* security-aware *sampling?* Finally, and most importantly, *how can we design an* efficient *(low-complexity, low-overhead) security-aware packet sampler to sample higher fractions of malicious packets?* In this section, we empirically answer the first two questions by evaluating ADS' under increasing number of malicious samples. After establishing consistent accuracy benefits provided by higher volumes of malicious samples, the remainder of this section is dedicated to designing and analytically evaluating an efficient security-aware sampler.



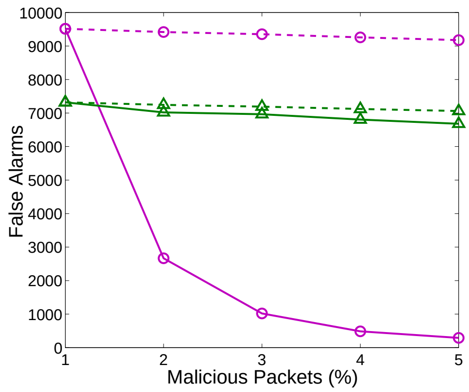
(a) Relative Detection



(b) Relative False Alarms



(c) Relative Detection



(d) Relative False Alarms

Figure 4.1: Linear, maximum and actual number of relative detections and relative false alarms under different malicious-to-benign sampling ratios; total sampling budget is fixed at $p_s = 0.05$.

4.1 Impact of Increasing Malicious Packet Samples on ADS Accuracy

To empirically answer the first two questions posed above, we use TCP portscan and TCP-SYN flood attacks at different rates.

From each attack dataset, we created five sampled subsets each with a sampling budget of $p_s = 0.05$; i.e., 5 out of every 100 packets were sampled. To emulate higher malicious packet fractions within this sampling budget, we respectively introduced 1%, 2%, 3%, 4% and 5% of malicious traffic samples in the five datasets. To analyze the impact of sampling with an increasing ratio of malicious-to-benign packets, the portscan datasets were input to TRW and NETAD, while the TCP-SYN flood datasets were input to Maximum Entropy and PHAD detectors.

It can be intuitively argued that a linear increase in the number of malicious packets at an ADS' input should introduce a linear increase in accuracy; this projected linear trend is shown as a dotted line in Fig. 4.1. The lines marked using asterisks in the detection plots represent the total number of malicious packets that are sampled in each dataset; i.e., the maximum number of detections that can be achieved by an ADS. Note that for Maximum Entropy, TRW and PHAD, a much faster than linear improvement in detection rate is observed. At the same time, the false positive rates of these detectors decrease exponentially with an increase in malicious traffic samples. Improvements for the NETAD detector are largely linear because the detector inherently has a very high detection rate with very few false positives even for the 1% dataset. It should be highlighted that accuracy improvements get more and more pronounced with an increase in the number of malicious sampled traffic. The detection rates of Maximum Entropy, TRW and PHAD quickly approach the maximum with an increase in the fraction of malicious packets. For instance, the PHAD detector achieves approximately 80% detection for the 5% dataset as opposed to only 25% detection rate for the 1% dataset. These detection rate improvements are complemented by drastically reduced false positives rates.

Based on the proof-of-concept results of this section, we conclude that a loss of sampling-induced accuracy in an ADS is proportional to the malicious-to-benign traffic samples at its input. Thus, in addition to being detected, these additional malicious packets also facilitate detection of other packets. The better-than-linear accuracy improvements achieved by increasing the number of malicious packets in a traffic ensemble motivates the need for a security-aware packet sampling algorithm which can sample higher volumes of malicious traffic. We propose such a technique in subsequent section.

4.2 Design Constraints

While having high malicious sampling rates, a practical security-aware sampler should satisfy these design constraints:

1. It should sample high volumes of malicious traffic;
2. It should be *generic* or algorithm-independent so that it can be seamlessly integrated with any anomaly detector;
3. It should have *low* (if any) *communication overhead* to allow inline realization; and
4. It should have *low complexity*[1] to facilitate its real-time implementation. Consequently, while we will allow our approach to incorporate some changes to the nodes' operation and for some information to be communicated between nodes, these changes must have very low complexity and communication overhead.

[1]We define complexity in terms of run-time complexity and memory usage.

4.3 Progressive Security-Aware Packet Sampling

At this point, we have established the considerable accuracy benefits of sampling higher volumes of malicious traffic. Therefore, we turn our attention to the last question of *how* a security-aware sampler will sample higher volumes of malicious traffic. While having high malicious sampling rates, a practical security-aware sampler must also satisfy the other design constraints set forth in Section 2

We propose a Progressive Security Aware Sampling (PSAS) algorithm which operates on the following principle: ADSs are deployed progressively on nodes on a packet's path. These ADSs communicate with each other by encoding their binary score (malicious or benign) of a packet inside the packet's header before forwarding it to the next hop node. The first node uses random sampling since it has no prior information to perform informed (security-aware) sampling. The security-aware sampler (PSAS) operating at the next hop uses this score as side information to sample packets marked as malicious (by the last hop node) with higher probabilities, while adhering to a given sampling budget. It should be clear that when a packet is classified as malicious at a node $k-1$ it may or may not be classified as malicious at the next node k. Specifically, node $k-1$ marks sampled and potentially-malicious packets to facilitate sampling at node k. Under a given packet sampling budget $p_s^{(k)}$ at node k, traffic which has been classified and then marked as malicious by the ADS at node $k-1$ is sampled with a high probability $p_{s_o}^{(k)}$ at next node k. After this security-aware sampling, the remaining packet sampling budget is exhausted by random samples from unmarked traffic.

The ADS deployed at each node k marks the classified as malicious packets independent of the fact that a packet was pre-

Algorithm 1: PSAS Algorithm

Input: Input Traffic D, Sampling Budget p_s, Security-Aware Sampling rate p_{s_o}, and a random number generator rnd.
Output: Sampled Traffic d

```
begin
    d̂ ← 0; */ d̂ is the number of sampled marked packets. */
    foreach (Packet p in D) do
        f_s ← p_s − p_{s_o} × d̂/|D|;
        /* score is a bit that contains the packet's security mark. */
        if p.score == malicious then
            generate rnd;
            if rnd ≤ p_{s_o} then
                d.add(p);
                d̂ = d̂ + 1;
                /* packets sampled by PSAS sampler become input to the ADS at each
                PSAS node. The ADS process and calculate its malicious score. */
                p.score = processPacket(p);
            end
        end
        else
            generate rnd;
            if rnd ≤ f_s then
                d.add(p);
                /* packets sampled by PSAS sampler become input to the ADS at each
                PSAS node. The ADS process and calculate its malicious score. */
                p.score = processPacket(p);
            end
        end
        /* forward packet p to the next hop node */
        p.forward(p.destIP);
    end
end
```

viously marked as malicious or it is previously unmarked. The mark on a packet is used by a PSAS sampler to preferentially sample the packets at the input of the ADS; however, the mark is not used as side information during ADS processing. Hence, the ADS at each node marks the classified as malicious packets which come: 1) from the previously marked packets; and 2) from randomly sampled packets. Consequently, as compared to random sampling, PSAS increases the number of correctly marked packets along a packet path as the number or nodes increases.

To meet the sampling budget, unmarked traffic is randomly sampled according to the following sampling function:

$$f_s^{(k)} = p_s^{(k)} - p_{s_o}^{(k)}(p_{\widehat{M}/M}^{(k)} + p_{\widehat{M}/B}^{(k)}).$$

The value of $f_s^{(k)}$ is greater or equal to zero. In the worst case, where all the sampled packet are marked as malicious by a node $k-1$, and sampled with $p_{s_o}^{(k)} = 1$ at the next hop k results in $f_s^{(k)} = 0$.

PSAS' apparently simple methodology satisfies our design constraints:

1. It can be observed intuitively—and will be mathematically proven shortly—PSAS will sample higher volumes of malicious packets if the progressive anomaly detectors are accurate. In fact, since anomaly detection accuracy generally degrades as we move from the endpoints to the network core [32], PSAS' sampling efficiency—which is driven by the previous hops—should improve at each progressive node.

2. PSAS can be used with *any* ADS. In fact, since PSAS allows different ADSs to be deployed at each hop, each of these ADSs can be customized for the traffic characteristics and attack vulnerabilities for a given point of network deployment.

3. PSAS has no additional communication overhead because progressive nodes communicate using only a single bit which can be encoded in unused IP packet headers, thereby precluding the need for an additional communication channel [15, 16] between nodes.

4. PSAS has very low complexity; empirical results substantiate this claim in the following section.

Stepwise execution of the proposed PSAS algorithm is shown in Algorithm 1. The following section mathematically proves that under certain realistic conditions PSAS always samples more malicious packets than blind random sampling.

4.4 Analytical Comparison of PSAS and Random Sampling

We first detail our assumptions and system model which is followed by analytical comparison of the two sampling approaches.

4.4.1 System Model and Assumptions

For analytical comparison, we make the following realistic assumptions:

- The total sampling budget is fixed to $p_s^{(k)}$;
- All the attacking nodes belong to the same subnet and each node i of them generates the malicious traffic at λ_{Mi} packets per unit time;
- Benign traffic increases at each node along the path;
- Probability of correct detection of malicious packets $p_d^{(k)}$ is greater or equal to probability of false positives $p_f^{(k)}$; and
- $p_{s_\circ}^{(k)} = 1$; this assumption is invoked to simplify mathematical exposition.

Based on the above assumptions, the rate of malicious traffic at each node is the same λ_M, while the rate of benign traffic at the k-th hop is $\sum_{i=1}^{k} \lambda_B^{(i)}$. The k-th node samples the incoming traffic with probability $p_s^{(k)}$ and passes it to the ADS which marks it based on its maliciousness level. Two types of traffic

Table 4.1: Symbols Definitions

Symbol	Definition
$\lambda_B^{(k)}$	Rate of benign traffic at k-th hop.
$\lambda_M^{(k)}$	Rate of malicious traffic at k-th hop.
$p_d^{(k)}$	Probability that a malicious packet will be detected at k-th hop.
$p_f^{(k)}$	Probability that a benign packet will be misclassified as malicious at k-th hop.
$p_{B\|B}^{(k)}$	Probability that an unmarked benign packet is received at k-th hop.
$p_{M\|M}^{(k)}$	Probability that an unmarked malicious packet is received at k-th hop.
$p_{\widehat{M}\|B}^{(k)}$	Probability that a benign packet mistakenly marked as malicious is received at k-th hop.
$p_{\widehat{M}\|M}^{(k)}$	Probability that a malicious packet correctly marked as malicious is received at k-th hop.

are received at each node: marked traffic (i.e., traffic marked as malicious) and unmarked traffic (i.e., traffic marked as benign or previously unsampled traffic).

To analytically model the packet sampling operation, we adopt a unique perspective: We treat sampling and the malicious traffic detection algorithm at each node as a channel. The input of this channel comprises four types of traffic called *symbols* in communication theory literature. The four symbols are: 1) unmarked benign packets, 2) unmarked malicious packets, 3) packets marked as malicious which are in fact malicious (correct detections), and 4) packets marked as malicious which are in fact benign (false positives). We follow the notation described in Table 4.1.

The probabilities that one symbol will get mapped to another is dependent on the accuracy ($p_f^{(k)}$ and $p_d^{(k)}$) of the k-th hop ADS as shown in Fig. 4.2. For instance, the probability that once sampled a marked malicious packet will again be marked as

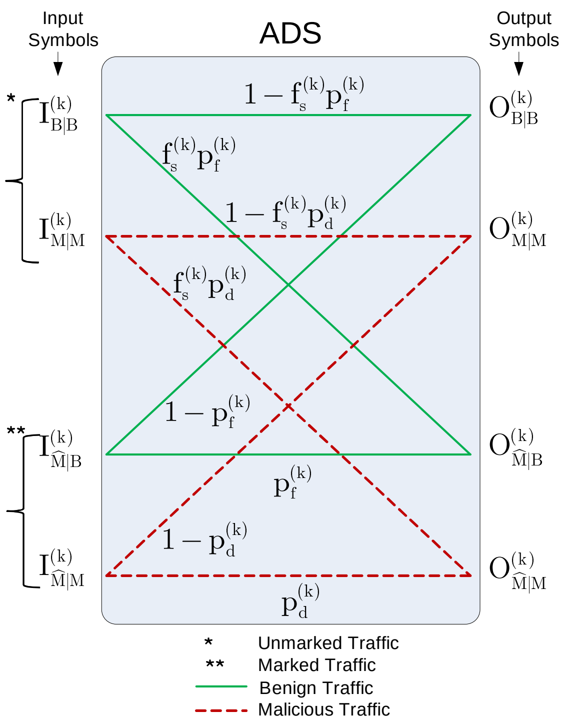


Figure 4.2: Probabilistic model of an ADS operating on security-aware traffic samples.

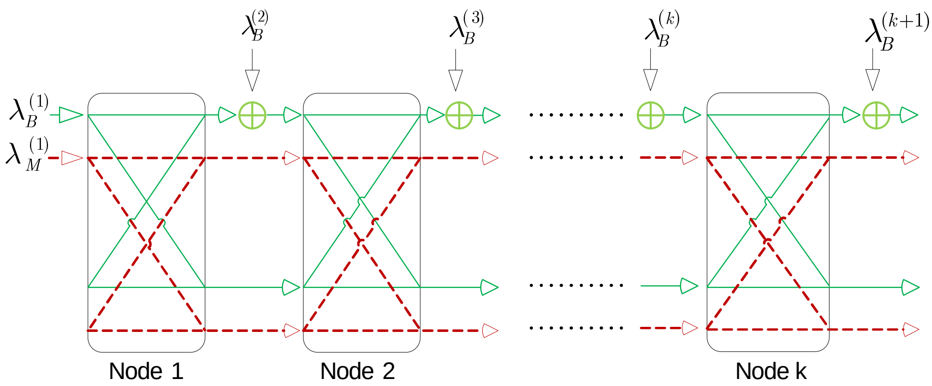


Figure 4.3: Security-aware sampling algorithm as a cascaded channel.

malicious is $p_d^{(k)}$. These probabilities are tuned in accordance with the sampling function $f_s^{(k)}$ defined earlier.

At the first node, we only have two types of traffic (unmarked benign and unmarked malicious); i.e., the probability of all other symbols is zero. Progressive packet markings by subsequent nodes will result in the cascaded channel shown in Fig. 4.3.

4.4.2 Malicious Traffic Sampling Rates

We now analytically compare the malicious traffic sampling rate of the proposed security-aware sampling with random sampling. In security-aware sampling, we sample from two different types of traffic: marked (as malicious) and unmarked (unsampled or marked as benign). We state the first result based on the above sampling function as follows.

Lemma 1. *The ratio of malicious packets in marked traffic is higher than the ratio of malicious packets in unmarked traffic at any node k for all $p_d^{(k)} \geq p_f^{(k)}$, where $k = 1, 2, \ldots$.*

Proof. The ratio of malicious packets in marked traffic is $\varepsilon_{\widehat{M}} = \frac{p^{(k)}_{\widehat{M}|M}}{p^{(k)}_{\widehat{M}|M}+p^{(k)}_{\widehat{M}|B}}$, while the ratio of malicious packets in unmarked traffic is $\varepsilon_M = \frac{p^{(k)}_{M|M}}{p^{(k)}_{B|B}+p^{(k)}_{M|M}}$. To prove that security-aware sampling algorithm samples more malicious packets than random sampling, we need to show that $\varepsilon_{\widehat{M}}$ is greater than ε_M. That is, we have to show that:

$$p^{(k)}_{B|B} p^{(k)}_{\widehat{M}|M} > p^{(k)}_{M|M} p^{(k)}_{\widehat{M}|B}.$$

Putting values from our system model shown in Fig. 4.2, we get

$$\left[p^{(k-1)}_{B|B}\left(1 - f_s^{(k)} p_f^{(k)}\right) + p^{(k-1)}_{\widehat{M}|B}\left(1 - p_f^{(k)}\right)\right] \times \left[p^{(k-1)}_{\widehat{M}|M} p_d^{(k)} + p^{(k-1)}_{M|M} f_s^{(k)} p_d^{(k)}\right]$$

$$\lesseqgtr$$

$$\left[p_{M|M}^{(k-1)}\left(1-f_s^{(k)}p_f^{(k)}\right)+p_{\widehat{M}|M}^{(k-1)}\left(1-p_d^{(k)}\right)\right]\times\left[p_{\widehat{M}|B}^{(k-1)}p_f^{(k)}+p_{B|B}^{(k-1)}f_s^{(k)}p_f^{(k)}\right]$$

$$\begin{gathered}\Rightarrow p_{B|B}^{(k-1)}p_{M|M}^{(k-1)}\left(1-f_s^{(k)}p_f^{(k)}\right)f_s^{(k)}\left(p_d^{(k)}-p_f^{(k)}\right)+p_{B|B}^{(k-1)}p_{\widehat{M}|M}^{(k-1)}\left(p_d^{(k)}-f_s^{(k)}p_f^{(k)}\right)\\+p_{\widehat{M}|B}^{(k-1)}p_{\widehat{M}|M}^{(k-1)}\left(p_d^{(k)}-p_f^{(k)}\right)\\ \lesseqgtr\\ p_{M|M}^{(k-1)}p_{\widehat{M}|B}^{(k-1)}\left[\left(1-f_s^{(k)}p_f^{(k)}\right)p_f^{(k)}-\left(1-p_f^{(k)}\right)f_s^{(k)}p_d^{(k)}\right]\end{gathered}\tag{4.1}$$

All the terms at the left hand side (LHS) of (4.1) are positive (>0), while the term on the right hand side (RHS) may or may not be positive. In general, all the terms on the LHS will sum to a much larger probability than the RHS terms. Even in the worst case, comparing the uncertain term on the RHS of equation (4.1) with a term on the LHS yields:

$$p_{B|B}^{(k-1)}p_{\widehat{M}|M}^{(k-1)}\geq p_{M|M}^{(k-1)}p_{\widehat{M}|B}^{(k-1)}$$

where the inequality holds as long as $p_d^{(k)}>p_f^{(k)}$ and the benign traffic rate is higher than the malicious traffic rate. □

The $p_d^{(k)}\geq p_f^{(k)}$ condition in the above lemma is quite relaxed. Recall that, as opposed to random sampling which samples from the entire traffic randomly, the proposed PSAS sampler samples packets marked as malicious with higher probability. Hence, in essence the above lemma states that for a given sampling budget $p_s^{(k)}$, the fraction of malicious traffic will be higher in security-aware sampled traffic.

By further constraining the relation between $p_d^{(k)}$ and $p_f^{(k)}$ within realistic limits, we reach the following corollary.

Corollary 1. *The ratio of malicious packets in marked traffic is much higher than the ratio of malicious packets in unmarked traffic at any node k for all $p_d^{(k)}\geq 2p_f^{(k)}$, where $k=1,2,\ldots$.*

Proof. Equation (4.1) can be written as:

$$\begin{gathered} p_{B|B}^{(k-1)} p_{M|M}^{(k-1)} \left(1 - f_s^{(k)} p_f^{(k)}\right) f_s^{(k)} \left(p_d^{(k)} - p_f^{(k)}\right) + p_{B|B}^{(k-1)} p_{\widehat{M}|M}^{(k-1)} \left(p_d^{(k)} - f_s^{(k)} p_f^{(k)}\right) + \\ p_{\widehat{M}|B}^{(k-1)} p_{\widehat{M}|M}^{(k-1)} \left(p_d^{(k)} - p_f^{(k)}\right) + p_{M|M}^{(k-1)} p_{\widehat{M}|B}^{(k-1)} \left(1 - p_f^{(k)}\right) f_s^{(k)} p_d^{(k)} + p_{M|M}^{(k-1)} p_{\widehat{M}|B}^{(k-1)} f_s^{(k)} p_f^{(k)} p_f^{(k)} \\ >> \\ p_{M|M}^{(k-1)} p_{\widehat{M}|B}^{(k-1)} p_f^{(k)}. \end{gathered} \tag{4.2}$$

All the terms in the above equation are positive as long as $p_d^{(k)} \geq p_f^{(k)}$. The term at the right side of the above equation is smaller than the second term at the left side which implies that the left side is much greater than the right side. By taking only the second term from the left side, we get

$$p_{B|B}^{(k-1)} p_{\widehat{M}|M}^{(k-1)} \left(p_d^{(k)} - f_s^{(k)} p_f^{(k)}\right) > p_{M|M}^{(k-1)} p_{\widehat{M}|B}^{(k-1)} p_f^{(k)},$$

which is true if $p_d^{(k)} - f_s^{(k)} p_f^{(k)} \geq p_f^{(k)}$, a condition that is satisfied when $p_d^{(k)} \geq 2p_f^{(k)}$. □

This corollary states that, under the very reasonable condition of $p_f^{(k)} = p_d^{(k)}/2$, PSAS will always sample considerably more malicious packets than random sampling. Note that these constraints on detection and false positive rates should be satisfied by *any* practical ADS. Hence, irrespective of the ADS used at each hop, PSAS should *always* sample higher fractions of malicious traffic than random sampling.

4.5 Summary and Discussion

Fig. 4.4 shows the system-level operation of the proposed security-aware sampling algorithm. As with existing commercial products [3, 4, 5, 6], the sampling algorithm and ADS are anticipated to be incorporated inside the router as shown in Fig. 4.4. Since

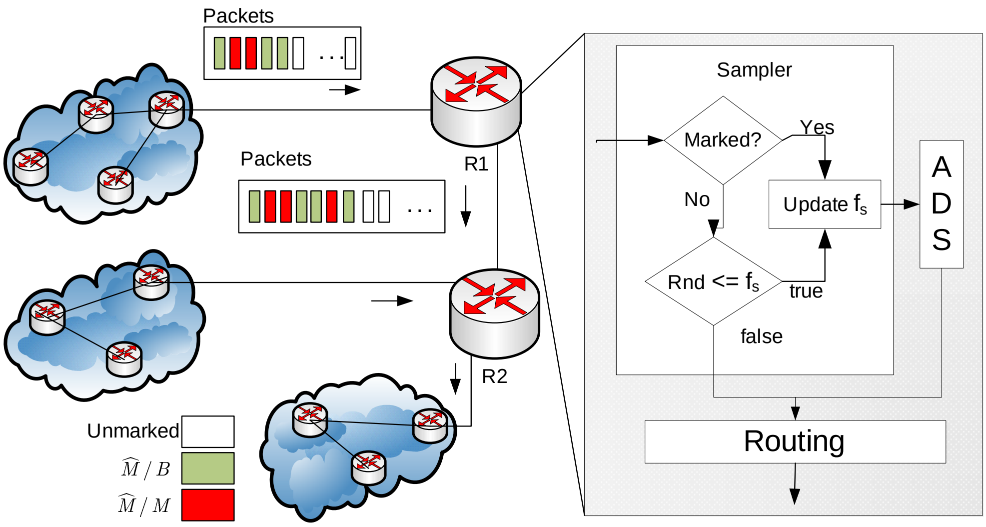


Figure 4.4: Pictorial representation of security-aware sampling.

the complexity of the proposed sampling algorithm is negligible as compared to the existing sampling and anomaly detection logics, we do not expect it to be an overhead. The only overhead is that multiple integrated Sampler-ADS-Router need to be deployed in the network path. This overhead can be minimized by deploying these devices only at one or two hops near the gateway router. Such a strategy–as shown in the following section–will yield the best results because at these hops a random sampler will not be able to sample enough malicious packets due to overwhelmingly high volumes of benign packets.

Chapter 5

Performance Evaluation

5.1 Accuracy Evaluation

Random sampling mainly causes an increase in missed detections. To cater for these missed detections, an ADS' classification threshold is generally decreased so that the few malicious packets which have been randomly sampled can be classified correctly. Interestingly, such a strategy results in more false positive because many benign packets are classified as malicious due to the low threshold. Therefore, random sampling affects both the detection rate and false alarm rate of an ADS. PSAS mitigates this problem by sampling malicious packets preferentially.

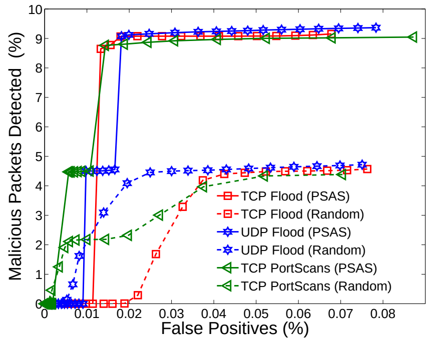
We use Receiver Operating Characteristic (ROC) curves to evaluate the accuracy improvements provided by PSAS. We deploy the same ADS at each hop and repeat the experiment for each of the four ADSs. ADSs are evaluated on TCP-SYN flood, UDP flood, and TCP portscans. We separately input security-aware and randomly sampled packets into these ADSs. Other sampling parameters are as follows: $p_s^{(k)} = 0.05; p_{s_o}^{(k)} = 1; k = 0, 1, 2$.

We designed an experimental setup for three cascaded nodes (Endpoint, lab router, and research wing router). The first node (endpoint) on a packet's path has no prior information about the maliciousness of the packet. We randomly sample packets at this first node. The second node along the packet path (first hop lab router) has some knowledge about the maliciousness of the traffic. The second node samples the packet marked by the first node as malicious with higher probability and the remaining sampling budget, if any, is exhausted by sampling the remaining budget randomly. The ADS at this second node marks any packets detected as malicious and similarly the third node along the packet path (e.g., the second hop research wing router in our experiments) follows the same procedure of sampling the marked packets with higher probability, marking any of the sampled traffic that is considered malicious by the ADS at that node and forwarding the (marked or unmarked) packet to the next hop along the path.

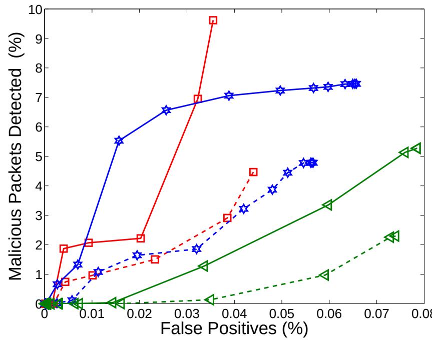
We obtain the results on the ADSs in offline mode. First, we input the randomly sampled endpoint traffic of the collected dataset to the ADSs. The ADSs mark the classified as malicious packets and then we use these marked packets as side information for the PSAS sampler to sample the marked packets preferentially in the lab router's traffic in the collected dataset. Similarly, the PSAS sampler uses the marked packets from the lab router traffic to sample the second hop research wing traffic of the dataset.

5.1.1 Averaged Accuracy Results

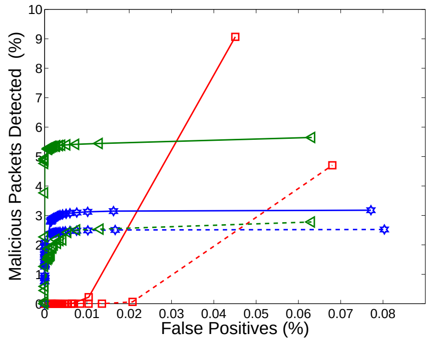
This section shows averaged accuracy results for all three attacks. Fig. 5.1 show that PSAS introduces a significant and consistent improvement in anomaly detection accuracy at the first hop. The most significant improvements are observed for the flood attacks; for instance, at a false positive rate of 0.03 for the TCP flood, approximately 3 and 2 times higher detection rate than random sampling are observed for Maximum Entropy



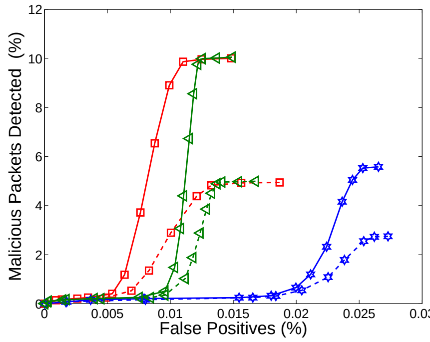
(a) Maximum Entropy



(b) PHAD



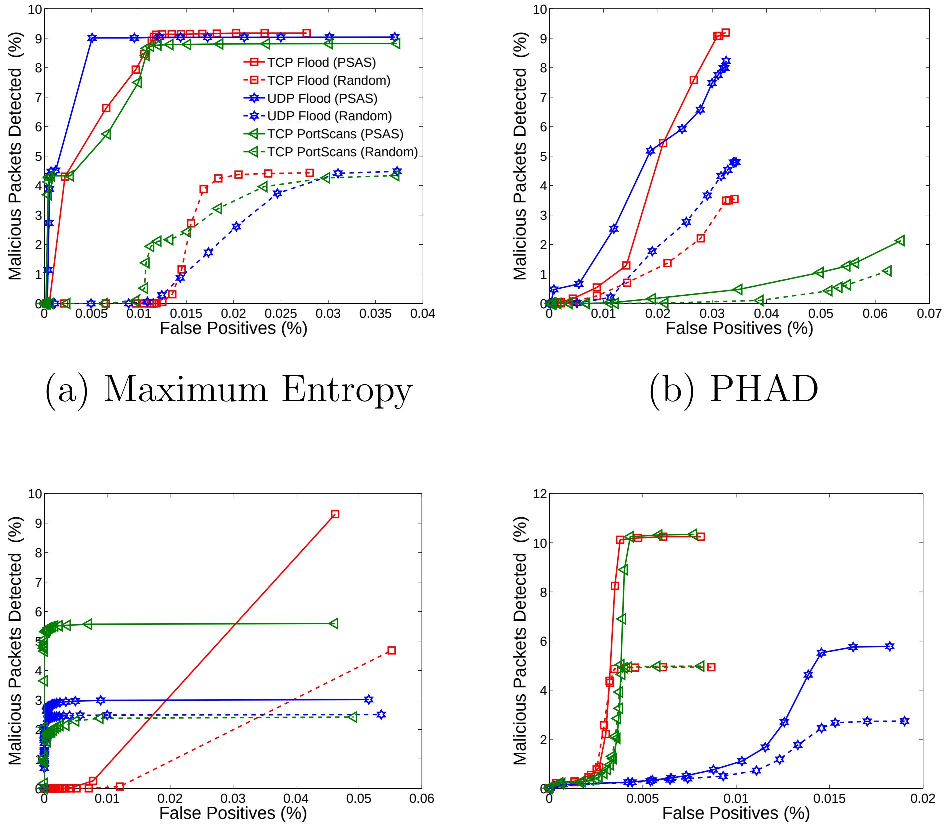
(c) TRW



(d) NETAD

Figure 5.1: ROC-based accuracy evaluation at the first hop; results are computed by averaging over all the attack packets of a particular attack.

and PHAD, respectively. Even for the portscan attacks, 3 and 10 times improvements in detection rate (at 0.03 false positive rate) over random sampling are respectively achieved for Maximum Entropy and PHAD. Similar improvements are observed for TRW-CB and NETAD at the first hop. For instance, at a false positive rate of 0.03, TRW-CB achieved 5 times improvement on TCP flood and 3 times on TCP Portscans datasets are achieved. Improvements of TRW-CB for UDP flood are not pro-



(a) Maximum Entropy (b) PHAD

(c) TRW (d) NETAD

Figure 5.2: ROC-based accuracy evaluation at the second hop; results are computed by averaging over all the attack packets of a particular attack.

nounced mainly because TRW is specifically designed to detect TCP portscans. NETAD at first hop achieved 4 times improvement for TCP portscans and TCP flood on a relatively lower false alarm rate of 0.012.

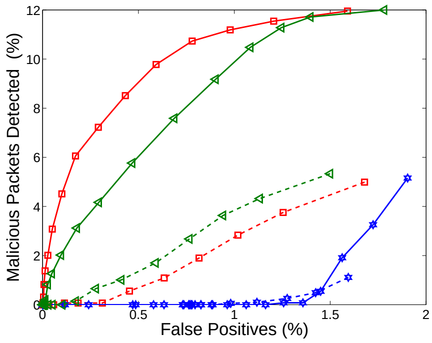
The improvements in accuracies are also quite pronounced at the second hop [Fig. 5.2]. For TCP portscans, TRW can achieve twice as many detections as random sampling for a false positive rate of 0.01. NETAD at the second hop has very low false positive rates but its detection rate saturates under random sampling. For the same false positive rate, PSAS allows NETAD to double its number of detections. Maximum Entropy on the second hop achieved 8 times improvement for the flood attacks and 4 times for the portscans attacks on a false alarm rate of 0.015. Similarly PHAD, at a false alarm rate of 0.02, achieved 5 times improvement for TCP flood and 3 times for UDP flood attacks' datasets.

We observed that PSAS' accuracy improvements get progressively more pronounced as the packets traverse through security-aware nodes. For instance, for the TCP portscan attack the Maximum Entropy detector could achieve approximately 100% increase in detection rate at hop 1, while the accuracy improvement at hop 2 was approximately 300%. Similar trends can be seen for other ADSs and attacks.

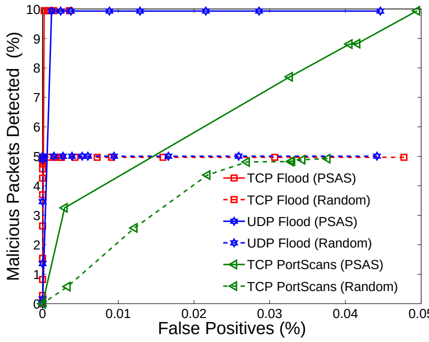
5.1.2 Accuracy Results Under Varying Attack Rates

This section shows the accuracy results of PSAS and random packet sampling for different attacks under varying rates. Results on lowest and highest attack rates are shown in this section. Results on medium attack rates are available at `http://wisnet.seecs.edu.pk/publications/2010/PSAS/`. Accuracy results for each ADS, under varying attack rates, are detailed separately as under:

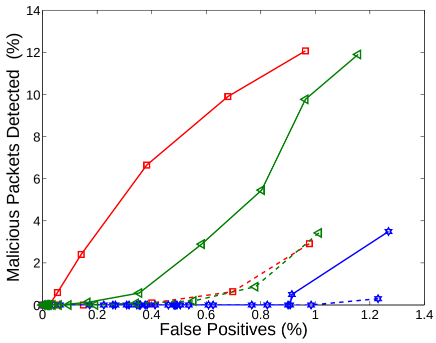
- *Maximum Entropy*
  Fig. 5.3 illustrates the accuracy results of Maximum Entropy under varying attack rates. It can be seen that the detection accuracy is improved more than double of the accuracy for random sampling at both first and second hops. The improvement is 3 times for low-rate (10 pkts/sec) attacks and 2 times for high-rate (1000 pkts/sec) attacks.



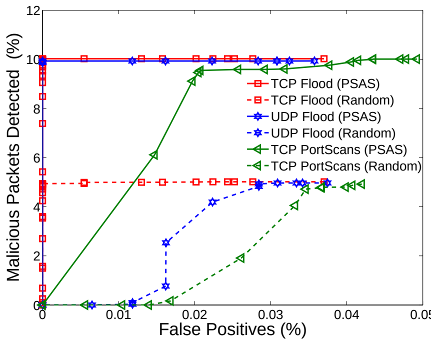
(a) First hop: attack rate 10 pkts/sec



(b) First hop: attack rate 1000 pkts/sec



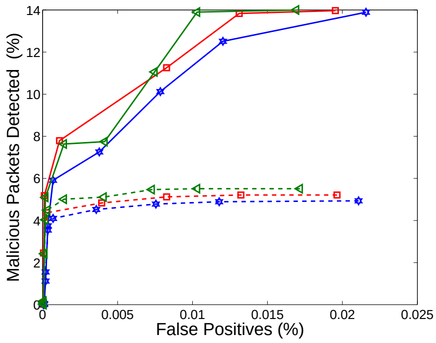
(c) Second hop: attack rate 10 pkts/sec



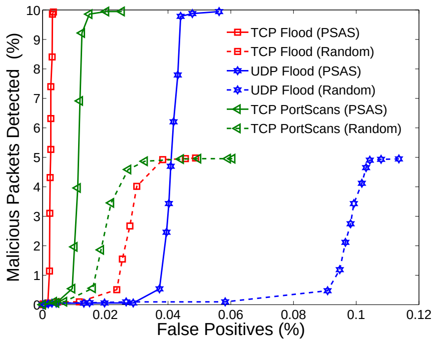
(d) Second hop: attack rate 1000 pkts/sec

Figure 5.3: Accuracy of Maximum Entropy ADS under varying attack intensities; results are computed separately for low and high rate attacks.

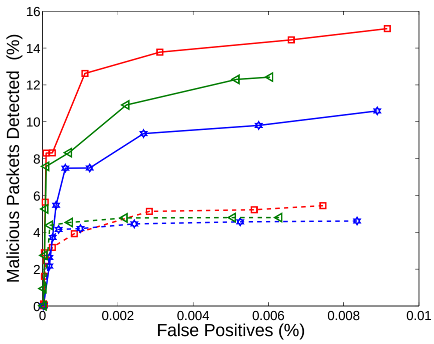
- *NETAD*
  Accuracy results of NETAD follow a trend similar to Maximum Entripy; while the false alarm rate is relatively very low. Fig. 5.4 illustrates the accuracy results of NETAD under varying (high and low-rate) attack rates. Improvement of PSAS on low rate attacks is 3 times while improvement



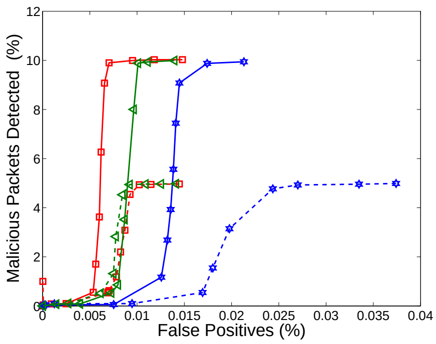
(a) First hop: attack rate 10 pkts/sec



(b) First hop: attack rate 1000 pkts/sec



(c) Second hop: attack rate 10 pkts/sec

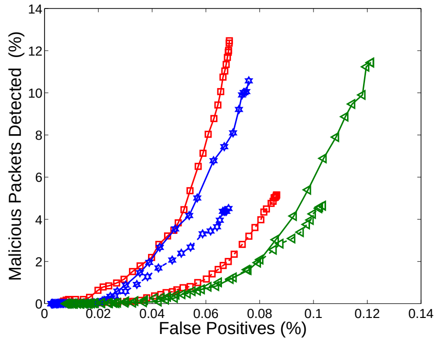


(d) Second hop: attack rate 1000 pkts/sec

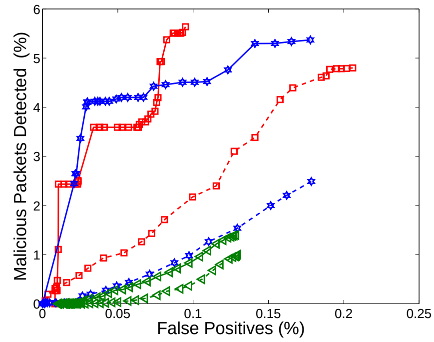
Figure 5.4: Accuracy of NETAD ADS under varying attack intensities; results are computed separately for low and high rate attacks.

on high rate attacks is more than doubled.

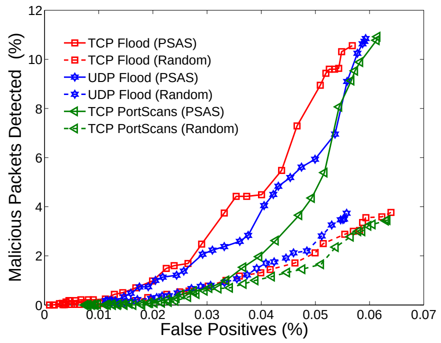
- *PHAD*
  PHAD produced good results on TCP and UDP flood attacks; while the detection rate is relatively lower for the TCP portscans attacks but still the improvement is more than doubled. Fig. 5.5 illustrates the accuracy results of



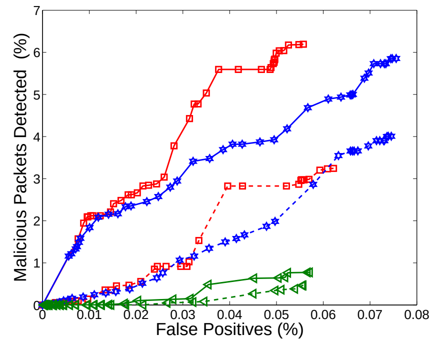
(a) First hop: attack rate 10 pkts/sec



(b) First hop: attack rate 1000 pkts/sec



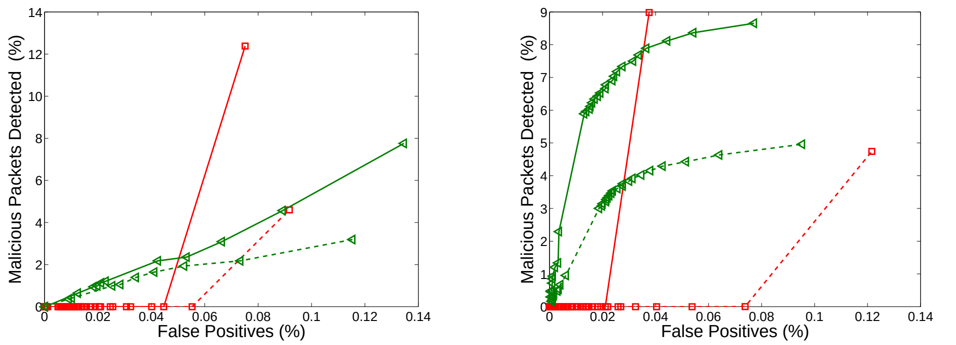
(c) Second hop: attack rate 10 pkts/sec



(d) Second hop: attack rate 1000 pkts/sec

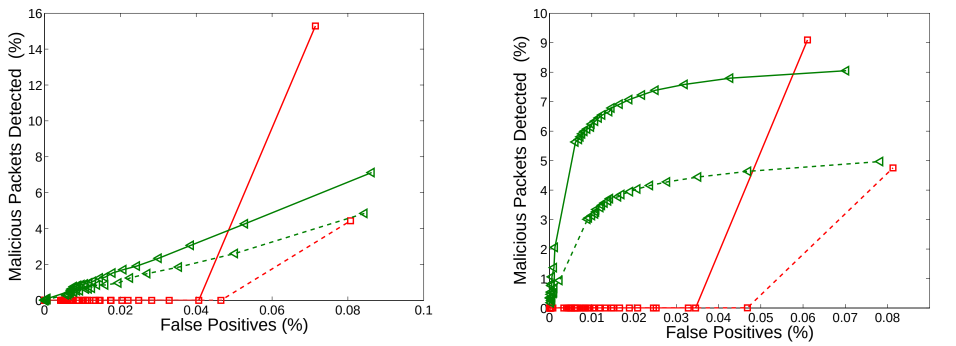
Figure 5.5: Accuracy of PHAD ADS under varying attack intensities; results are computed separately for low and high rate attacks.

PHAD under varying (high and low-rate) attack rates. Improvement for the low-rate attacks (10 pkts/sec) is slightly prominent than the results for high-rate attacks.



(a) First hop: attack rate 10 pkts/sec

(b) First hop: attack rate 1000 pkts/sec



(c) Second hop: attack rate 10 pkts/sec

(d) Second hop: attack rate 1000 pkts/sec

Figure 5.6: Accuracy of TRW-CB ADS under varying attack intensities; results are computed separately for low and high rate attacks.

- *TRW-CB*
  TRW-CB did not perform very well on TCP flood attacks i.e., the detection rate is relatively lower than TCP portscans. The reason of not performing well on flood attacks is that TRW is designed specifically for the detection of portscans attacks. Fig. 5.6 shows the accuracy results of TRW-CB

under varying attack rates. It can be seen that the acceptable detection rates are more than doubled for PSAS than using random packet sampling irrespective of the attack type and rate.

5.1.3 Discussion on Accuracy Evaluation

It is concluded from the accuracy results (generated by averaging different attack rates) [Figures 5.1,5.2] and from the results (generated separately for each attack rate) [Figures 5.3-5.6] that the detection accuracy is improved by a significant factor irrespective of attack rate and type. For example, in the case of a low-rate attack (considered more difficult to accurately detect as compared to high-rate attacks), the detection accuracies are more than double of the accuracies achieved using random packet sampling. Thus, we conclude that PSAS can improve anomaly detection accuracies regardless of the underlying attack rates and types.

We do not show the results of the ADSs on 0.1 and 1 pkts/sec attack intensities.The reason is that the affect of sampling on very low rate attacks is most severe and these attacks remain undetectable. In our scenario, under sampling budget of 5%, the intensities of 0.1 and 1 pkts/sec attacks decrease to 0.005 and 0.05 pkts/sec respectively. Detection of such low rate attacks at today's high speed links is very difficult to realize and therefore we observed a 0% detection rate for all ADSs.

5.2 Complexity and Communication Overhead

As emphasized earlier, our proposed PSAS algorithm allows different nodes in the network to communicate using only a binary score which can be easily encoded inside an IP packet. This procedure does not utilize any additional bandwidth or commu-

Table 5.1: Complexity of PSAS and Random Sampling to Sample One Second of Traffic

	PSAS	Random	PSAS	Random
Attack rate (pkts/sec)	10	10	1000	1000
Time (sec)	0.074	0.082	0.0578	0.0765

nication overhead. Moreover, PSAS does not require any extra memory because the packet marks are stored inside the packet. Therefore, additional data structures are not required by PSAS and its data memory requirements are identical to random sampling.

Table 5.1 shows that the run-time complexity[1] of PSAS is comparable to random sampling at low attack rates. Interestingly, the run-time complexity of PSAS is lower than random sampling at high attack rates. This was observed because the random sampler generates a random number rnd, between 0 and 1, for each incoming packet and samples the packet when the rnd is less than or equal to the sampling budget $p_s^{(k)}$; in our experiments we use $p_{s_o}^{(k)} = 1$ and consequently the PSAS' sampling simply involved picking up a large number of marked malicious packets and the overhead of random number generation was reduced.

We argue that our sampling algorithm is substantially less expensive as compared to the normal operation of a typical Gigabit network router which has to extract and change destination MAC addresses from each packet, as well as update the CRC value. In comparison, checking and modifying a single bit value in each packet has negligible complexity.

The computational complexity of PSAS may be further improved by modifying the packet marking technique as under: when a packet is marked as malicious at a node k, all the sub-

[1] Complexity is measured using the `hprof` tool on a dual core 2.2 GHz Intel machine. File I/O is not included in complexity.

sequent packets of the same flow can be marked as malicious without the need of inspecting each individual sampled packet by the ADS. However, this technique does not necessarily provide higher detection accuracy because the detection by the ADS might be incorrect.

Chapter 6

Limitations and Countermeasures

We now highlight some limitations of the proposed PSAS technique and offer solutions to circumvent these limitations.

- Since PSAS samples every marked packet with probability $p_{s_\circ}^{(k)}$, if a malicious packet is skipped at node $k-1$, it will likely continue to be skipped further along its path. To counter this issue, the sampling parameters $p_s^{(k)}$ and $p_{s_\circ}^{(k)}$ can be tuned to support detection of new malicious packets while sustaining previously detected threats.

- A solution which depends on coordination among routers or network devices causes the system to be more complex as compared to typical ADS deployments. PSAS, however, has a simple communication scheme which does not have any additional bandwidth overhead. Thus, a marginal change in ADS implementation and deployment may be considered a worthwhile trade-off for the substantial security improvements brought about.

- Malicious packets can evade sampling by increasing the rate of attack so that the sampling budget is exhausted; e.g., DoS attacks can be used to hide portscan attacks. This type of evasion can be mitigated by maintaining a list of malicious hosts observed in window n and then sampling these hosts preferentially in window $n + 1$.

- Inline intrusion detection can have an adverse affect on delay sensitive applications; for example, undesirable jitter may be introduced in a multimedia application. Such a scenario will only arise if the delay-sensitive packets are marked as malicious. This problem can only be mitigated by improving anomaly detection accuracy.

Chapter 7

Conclusions and Future Work

7.1 Conclusions

At high-speed links, it is not feasible for network devices to analyze each and every packet. Real-time high-speed anomaly detection systems use packet sampling to realize traffic analysis at wire speed. Sampling is a lossy process which results in an incomplete and biased approximation of the underlying traffic. Packet sampling can introduce significant accuracy degradations in an ADS [11]–[14], solutions to mitigate this accuracy loss are largely unexplored in research literature.

In this thesis, we propose a solution to simultaneously address these open problems by enabling an inline ADS to achieve higher accuracy under sampling by correlating traffic from different points of deployment in a network.

We propose a Progressive Security Aware Sampling (PSAS) algorithm which operates on the following principle: ADSs are deployed progressively on nodes on a packets path. These ADSs communicate with each other by encoding their binary score (malicious or benign) of a packet inside the packets header before forwarding it to the next hop node. The security-aware sampler (PSAS) operating at the next hop uses this score as side information to sample packets marked as malicious (by the last hop node) with higher probabilities, while adhering to a given sampling budget.

We show that the proposed simple collaboration model, referred to as Progressive Security-Aware Sampling (PSAS), enables inline anomaly detectors to achieve significantly higher accuracies by mitigating the information loss under sampling.

While there exist a few public and labeled traffic attack datasets; these datasets do not satisfy our requirements (see chapter 3). Therefore, we collect our own traffic dataset. For repeatable performance evaluation, our labeled dataset is publicly available at http://wisnet.seecs.edu.pk/datasets/.

We analytically proved that, under some realistic constraints on detection and false alarm rates, PSAS always sample considerably more malicious packets than random sampling. Note that these constraints on detection and false positive rates should be satisfied by any practical ADS. Hence, irrespective of the ADS used at each hop, PSAS should always sample higher fractions of malicious traffic than random sampling.

PSAS sampling is efficient, having no communication overhead and low complexity. From the accuracy results, we conclude that the sampling-induced accuracy degradation in an ADS can be significantly mitigated irrespective of attack rate and type. We also observed that PSAS accuracy improvements get progressively more pronounced as the packets traverse through securityaware nodes.

To the best of the author's knowledge, this thesis proposes the first known solution to mitigate sampling-induced accuracy loss in an anomaly detection system, with promising avenues for further research in this area.

7.2 Future Work

This research work is based on progressive marking of packets along its path and hence needs the ADS(s) to be deployed on multiple hops. As a future work, a distributed packet sampling algorithm can be devised which can somehow intelligently sample the malicious packets without the need to get the binary decision from the previous hop. Such a *Distributed Security-Aware Sampling (DSAS)* may be based on a technique which logs some information about the packets/flows marked as malicious in window k and that information can be used to sample packets in the $(K+1)^{th}$ window.

Bibliography

[1] M. S. Kim, H. J. Kang, S. C. Hung, S. H. Chung, and J. W. Hong, "A Flow-based Method for Abnormal Network Traffic Detection," *IEEE/IFIP NOMS*, 2004.

[2] A. Lakhina, M. Crovella, and C. Diot, "Mining Anomalies Using Traffic Feature Distributions," *ACM SIGCOMM*, 2005.

[3] Cisco Anomaly Guard Module Homepage, `www.cisco.com/en/US/products/ps6235/`.

[4] Arbor Networks Peakflow-X Homepage, `http://www.arbornetworks.com/en/peakflow-x.html`.

[5] Endace NinjaBox Homepage, `http://www.endace.com/ninjabox.html`.

[6] FireEye Homepage, `http://www.fireeye.com/`.

[7] B.-Y. Choi, J. Park, and Z.-L. Zhang, "Adaptive random sampling for total load estimation," *IEEE ICC*, 2003.

[8] N. Duffield, C. Lund, and M. Thorup, "Properties and prediction of flow statistics from sampled packet streams," *ACM IMC*, 2002.

[9] N. Duffield, C. Lund, and M. Thorup, "Estimating Flow Distributions from Sampled Flow Statistics," *ACM SIGCOMM*, 2003.

[10] N. Hohn and D. Veitch, "Inverting Sampled Traffic," *ACM IMC*, 2003.

[11] J. Mai, A. Sridharan, C. N. Chuah, H. Zang, and T. Ye, "Impact of packet sampling on portscan detection," *IEEE J. SAC*, 24(12):2285-2298, 2006.

[12] J. Mai, C. N. Chuah, A. Sridharan, T. YE, and H. Zang, "Is sampled data sufficient for anomaly detection?" *ACM IMC*, 2006.

[13] G. Androulidakis, V. Chatzigiannakis, S. Papavassiliou, M. Grammatikou, V. Maglaris, "Understanding and Evaluating the Impact of Sampling on Anomaly Detection Techniques," *IEEE MILCOM*, 2006.

[14] D. Brauckhoff, B. Tellenbach, A. Wagner, M. May, A. Lakhina, "Impact of Packet Sampling on Anomaly Detection Metrics," *ACM IMC*, 2006.

[15] Y. Kim, W. C. Lau, M. C. Chuah, and H. J. Chao, "PacketScore: Statistics-based Overload Control against Distributed Denial-of-Service Attacks," *IEEE INFOCOM*, 2004.

[16] P. E. Ayres, H. Sun, and H. J. Chao, "ALPi: A DDoS Defense System for High-Speed Networks," *IEEE J. SAC*, 24(10):1864-1876, 2006.

[17] Y. Gu, A. McCullum, and D. Towsley, "Detecting anomalies in network traffic using maximum entropy estimation," *ACM IMC*, 2005.

[18] S. E. Schechter, J. Jung, and A. W. Berger, "Fast detection of scanning worm infections," *RAID*, 2004.

[19] M. V. Mahoney and P. K. Chan, “PHAD: Packet Header Anomaly Detection for Indentifying Hostile Network Traffic,” Technical Report, *Florida Tech.*, CS-2001-4.

[20] M. V. Mahoney, ”Network traffic anomaly detection based on packet bytes,” *ACM Symposium on Applied Computing*, 2003.

[21] C. Estan and G. Varghese, “New Directions in Traffic Measurement and Accounting,” *ACM SIGCOMM*, 2002.

[22] N. Duffield, C. Lund, and M. Thorup, “Properties and Prediction of Flow Statistics from Sampled Packet Streams,” *ACM IMW*, 2002.

[23] P. Barford, J. Kline, D. Plonka, and A. Ron, “A Signal Analysis of Network Traffic Anomalies,” *ACM IMW*, 2002.

[24] J. Jung, V. Paxson, A. W. Berger, H. Balakrishnan, “Fast portscan detection using sequential hypothesis testing,” *IEEE Symp S&P*, 2004.

[25] A. Sridharan, T. Ye, and S. Bhattacharyya, “Connection Port Scan Detection on the Backbone,” *IPCC Malware Workshop*, 2006.

[26] R. Pokrywka, “Reducing False Alarm Rate in Anomaly Detection with Layered Filtering,” *ICCS*, 2008.

[27] R. Perdisci, D. Ariu, P. Fogla, G. Giacinto, W. Lee, “McPAD: A multiple classifier system for accurate payload-based anomaly detection”, *Computer Networks*, 2009.

[28] L. Huang, X. Nguyen, M. Garofalakis, J. M. Hellerstein, M. I. Jordan, A. D. Joseph, N. Taft, “Communication-Efficient Online Detection of Network-Wide Anomalies,” *IEEE Infocom*, 2007.

[29] DARPA Intrusion Detection Data Sets, www.ll.mit.edu/mission/communications/ist/corpora/ideval/data/index.html.

[30] LBNL/ICSI Dataset, www.icir.org/enterprise-tracing/download.html.

[31] Endpoint Dataset, wisnet.seecs.edu.pk/projects/ENS/DataSets.html.

[32] A. B. Ashfaq, M. J. Robert, A. Mumtaz, M. Q. Ali, A. Sajjad, and S. A. Khayam, "A Comparative Evaluation of Anomaly Detectors under Portscan Attacks," *RAID*, 2008.

[33] Lippmann, R.P., Haines, J.W., Fried, D. J., Korba, J., Das, K.: The 1999 DARPA OffLine Intrusion Detection Evaluation.In: *Comp. Networks*, 34(2), pp. 579-595 (2000)

[34] Fawcett T.: ROC Graphs: Notes and Practical Considerations for Researchers. Technical report, Palo Alto, USA : HP Laboratories (2004)

[35] Fawcett T.: An introduction to ROC analysis. In: *Elsevier* (2005)

[36] A. Kumar and J. Xu, "Sketch Guided Sampling-Using On-Line Estimates of Flow Size for Adaptive Data Collection," *IEEE INFOCOM*, 2006.

[37] A. Ramachandran, S. Seetharaman, and N. Feamster, "Fast monitoring of traffic subpopulations," *ACM IMC*, 2008.

[38] Internet Security Systems, http://www.iss.net/security_center/advice/Exploits/TCP/SYN_flood/default.htm.

[39] S. Kumar, "Smurf-based Distributed Denial of Service (DDoS) Attack Amplification in Internet," *IEEE Second*

International Conference on Internet Monitoring and Protection (ICIMP), 2007.

[40] K. Wooding, "Magnification Attacks - Smurf, Fraggle, and Others," `http://www.codetalker.com/whitepapers/dossmurf.html`.