J Comput Virol (2011) 7:63–81
DOI 10.1007/s11416-009-0133-5

ORIGINAL PAPER

Accuracy improving guidelines for network anomaly detection systems

Ayesha Binte Ashfaq · Muhammad Qasim Ali · Syed Ali Khayam

Received: 4 May 2009 / Accepted: 20 August 2009 / Published online: 4 September 2009

Abstract An unprecedented growth in computer and communication systems in the last two decades has resulted in a proportional increase in the number and sophistication of network attacks. In particular, the number of previously-unseen attacks has increased exponentially in the last few years. Due to the rapidly evolving nature of network attacks, a considerable paradigm shift has taken place in the intrusion detection community. The main focus is now on Network Anomaly Detection Systems (NADSs) which model and flag deviations from normal/benign behavior of a network and can hence detect previously-unseen attacks. Contemporary NADS borrow concepts from a variety of theoretical fields (e.g., Information theory, stochastic and machine learning, signal processing, etc.) to model benign behavior. These NADSs, however, fall short of achieving acceptable performance levels as therefore widespread commercial deployments. Thus, in this paper, we firstly evaluate the performance of eight prominent network-based anomaly detectors under malicious portscan attacks to identify which NADSs perform better than others and why. These NADSs are evaluated on three criteria: accuracy (ROC curves), scalability (with respect to varying normal and attack traffic rates, and deployment points) and detection delay. These criteria are evaluated using two independently collected datasets with complementary strengths. We then propose novel methods and promising guidelines to improve the accuracy and scalability of existing and future anomaly detectors. Experimental analysis of the proposed guidelines is also presented for the proof of concept.

A. B. Ashfaq (✉) · M. Q. Ali · S. A. Khayam
School of Electrical Engineering and Computer Science,
National University of Science and Technology (NUST),
Islamabad 44000, Pakistan
e-mail: ayesha.ashfaq@seecs.edu.pk

M. Q. Ali
e-mail: mqasim.ali@seecs.edu.pk

S. A. Khayam
e-mail: ali.khayam@seecs.edu.pk

1 Introduction

The CodeRed worm of 2001 catalyzed a notable shift in the network attack paradigm. This first largescale malicious attack revealed the destructive capability of an automated and distributed attack that can be launched using compromised hosts. Since then, network attacks have evolved considerably and malware, botnets, spam, phishing, and denial of service attacks have become continuous and imminent threats for today's networks and hosts. Financial losses due to these attacks have been overwhelming. For instance, the economic losses to recover from the CodeRed worm alone are estimated at \$2.6 billion [2]. In addition to the short-term revenue losses for businesses and enterprises, network attacks also compromise information confidentiality/integrity and cause disruption of service, thus resulting in a long-term loss of credibility.

In order to combat the rapidly-evolving attacks, network intrusion detection methods have also become increasingly sophisticated. In broad terms, the field of intrusion detection comprises two types of detection methods: misuse detection and anomaly detection. Misuse detection, the predominant detection method employed in today's anti-virus software, requires a signature of an attack to be known before the attack can be detected. While such signature-based detectors can provide 100% detection rates for known attacks, they have an inherent limitation of not being able to detect new or previously-unseen attacks; a 468% increase in previously-unseen attacks was reported over just a six month period in 2007 [1]. Moreover, development and dissemination of attack signatures require human intervention and therefore misuse detectors are finding it difficult to cope with rapidly-evolving network intrusions. On the other end of the intrusion detection spectrum are Network Anomaly Detection Systems (NADSs) which model the benign or normal traffic behavior of a network or host and detect significant deviations from this model to identify anomalies in network traffic. Since NADSs rely on normal traffic behavior for attack detection, they can detect previously-unknown attacks. Consequently, significant research effort has been focussed on development of NADSs in the past few years [3].[1]

The main challenge of NADSs is to define a robust model of normal traffic behavior. In antivirus detection, however, this is trivial since the normal behavior pertains to the defined signatures learnt over a period of time. These learnt signatures are then matched against the signatures in traffic, under analysis, to detect known malware. However, since NADSs strive to detect unknown malware as well (zero-day attacks), the detection is inherently more complicated. In NADSs, an accurate model of normal traffic behavior needs to cater for changes in normal behavior over time. Such changes in normal traffic behavior lead to potentially low detection rates and high false alarm rates of NADSs. In view of the vast research literature on network anomaly detection, it is important to evaluate the performance of these NADSs and identify the shortcomings. Based on our findings, we propose novel guidelines that can be used to improve the accuracy of existing and future NADSs along with experimental results for the proof of concept.

Following are the objectives of this research work:

- to quantify and compare the accuracies of some of the prominent detectors under varying rates of attack and normal traffic and at different points of deployment;
- to identify promising traffic features and theoretical frameworks for portscan anomaly detection;
- to investigate the detection delay of anomaly detectors;
- to identify a set of promising portscan detection guidelines that build on the strengths and avoid the weaknesses of the evaluated anomaly detectors; and finally
- to provide experimental results for the accuracy improvements achieved by the proposed guidelines.

We evaluate the NADSs on three criteria: accuracy, scalability, and detection delay. Accuracy is evaluated by comparing ROC (detection rate versus false alarm rate) characteristics of the NADSs. Scalability is evaluated with respect to different background and attack traffic rates. Since the two datasets used in this study are collected at different network entities and contain attacks with different characteristics, evaluation over these datasets allows us to compare the scalability of the proposed NADSs under varying traffic volumes. Detection delay is evaluated separately for high- and low-rate attacks.

Based on our findings, we propose a few promising guidelines to improve the accuracy and scalability of existing and future NADSs. Our results show that the proposed guidelines result in an average detection rate increase of 5–10%, while reducing the false alarm rates up to 50%.

The remainder of this document is structured as follows: Sect. 2 outlines the existing research in the areas of NADSs and IDS evaluation. Section 3 gives a brief description of the evaluated anomaly detection systems and the datasets used. Section 4 provides the comparative performance evaluation results in the form of ROC curves on the two datasets [53]. Section 5 provides the lessons learnt from the comparative performance evaluation of prominent NADSs. Moreover, promising portscan detection guidelines are proposed to improve the performance of existing and future NADSs. Experimental results are also provided in the form of ROC curves for the performance improvements realized by jointly using the proposed guidelines. Section 6 summarizes the key conclusions of this work.

2 Related work

In this section, we focus on prior IDS/ADS evaluation studies. Details of anomaly detectors evaluated in this work are deferred to subsequent sections.

Performance evaluation of IDSs received significant attention from the industry and academia in the late 1990's [30–44]. However, in the past few years, only four studies have performed comparative comparison of anomaly detectors [27–29]. Similarly, very few prior studies have performed ROC analysis of the evaluated IDSs. Still fewer studies have made their evaluation datasets available online.

DARPA-funded IDS evaluation studies by the MIT Lincoln Lab in 1998 and 1999 represent a shift in the IDS evaluation methodology [4,37]. Datasets used in these studies were made publicly available [38] and the ROC method used in these studies has since become the de facto standard for IDS accuracy evaluation. While some shortcomings of the DARPA evaluation have been highlighted [45,46], in the absence of other benchmarks, the results and datasets of this study have been used extensively in subsequent works. In the present study's context, the DARPA dataset is somewhat dated.

The four recent ADS evaluation studies focus on specific types of detectors and attacks [27–29]. The study by Wong et al. [27] is most relevant in the present context. Wong et al. [27] evaluated four variants of the rate limiting detector under

[1] Interestingly, the promise and advantages of anomaly detectors over signature detectors were identified by the seminal DARPA-funded IDS evaluation studies much before the CodeRed worm [4,37].

portscan attacks at two different network points [5–12]. Two findings of this study are pertinent to the present work: (1) classical rate limiting is not an effective technique for portscan detection, and (2) rate limiting can operate on aggregate-level DNS traffic and hence can potentially scale to core-level deployments. Attack and background traffic data used in this study are not publicly available.

Ingham and Inoue [28] compared seven HTTP anomaly detection techniques under real-world attacks reported at public databases. These authors report the same evaluation difficulties that were faced by us: (1) Some anomaly detectors are not described completely; (2) Implementation source code is not available; and (3) labeled data used for algorithm evaluation are not publicly available. Consequently, the authors in [28] make their implementation and attack data publicly available "to encourage further experimentation". We subscribe to the same viewpoint and therefore all data and implementation used in this project are available online [26]. Lazarevic et al. [29] performed a comparative analysis of four data mining based anomaly detection techniques. The live network traffic data used in this study is not publicly available.

3 ADS evaluation framework

In this section, we would give details regarding the evaluated anomaly detection systems and the datasets used. Moreover, characteristic features of the two datasets will also be provided.

3.1 Anomaly detection algorithms

We will focus on network-based anomaly detectors and compare the anomaly detectors proposed in [5,8,9,13,16,19,21,22]. Most of these detectors are quite popular and used frequently for performance comparison and benchmarking in the Intrusion Detection (ID) research community. Improvements to these algorithms have also been proposed in [7,10–12,14,15,17,18,20,27].[2]

Before briefly describing these detectors, we highlight that some of these detectors are designed specifically for portscan detection, while others are general-purpose network anomaly detectors. We provide brief descriptions of the evaluated algorithms. Majorly we will focus on the algorithm adaptation and parameter tuning for the datasets under consideration. Readers are referred to [5,8,9,13,16,19,21,22] for details of the algorithms. For techniques operating on fixed-sized time windows, we use a window of 20 s. All other parameters not mentioned in this section are the same as those described in the algorithms' respective papers.

3.1.1 Rate limiting [5,6]

Detects anomalous connection behavior by relying on the premise that an infected host will try to connect to many different machines in a short period of time. Rate limiting detects portscans by putting new connections exceeding a certain threshold in a queue. An alarm is raised when the queue length, η_q, exceeds a threshold. ROCs for endpoints are generated by varying $\eta_q = \mu + k\sigma$, where μ and σ represent the sample mean and sample standard deviation of the connection rates in the training set, and $k = 0, 1, 2, \ldots$ is a positive integer. Large values of k will provide low false alarm and detection rates, while small values will render high false alarm and detection rates. In the Lawrence Berkeley National Laboratory (LBNL) dataset [25], connection rate variance in the background traffic is more than the variance in the attack traffic. Therefore, to obtain a range of detection and false alarm rates for the LBNL dataset, we use a threshold of $\eta_q = w\mu$, with a varying parameter $0 \leq w \leq 1$, and the queue is varied between 5 and 100 sessions.

3.1.2 Threshold Random Walk (TRW) algorithm [8]

Detects incoming portscans by noting that the probability of a connection attempt being a success should be much higher for a benign host than for a scanner. To leverage this observation, Threshold Random Walk (TRW) uses sequential hypothesis testing (i.e., a likelihood ratio test) to classify whether or not a remote host is a scanner. We plot ROCs for this algorithm by setting different values of false alarm and detection rates and computing the likelihood ratio thresholds, η_0 and η_1, using the method described in [8].

3.1.3 TRW with Credit-based Rate Limiting (TRW-CB)[9]

A hybrid solution to leverage the complementary strengths of Rate Limiting and TRW was proposed by Schechter et al. [9]. Reverse TRW is an anomaly detector that limits the rate at which new connections are initiated by applying the sequential hypothesis testing in a reverse chronological order. A credit increase/decrease algorithm is used to slow down hosts that are experiencing unsuccessful connections. We plot ROCs for this technique for varying η_0 and η_1 as in the TRW case.

3.1.4 Maximum Entropy [21]

Detector estimates the benign traffic distribution using maximum entropy estimation. Traffic is divided into 2,348 packet classes and maximum entropy estimation is then used to

[2] Some promising commercial ADSs are also available in the market now [23,24]. We did not have access to these ADSs, and therefore these commercial products are not evaluated in this study.

develop a baseline benign distribution for each packet class. Packet class distributions observed in real-time windows are then compared with the baseline distribution using the Kullback-Leibler (K-L) divergence measure. An alarm is raised if a packet class' K-L divergence exceeds a threshold, η_k, more than h times in the last W windows of t seconds each. Thus the Maximum Entropy method incurs a detection delay of at least $h \times t$ seconds. ROCs are generated by varying η_k.

3.1.5 Packet Header Anomaly Detection (PHAD) [13]

Learns the normal range of values for all 33 fields in the Ethernet, IP, TCP, UDP and ICMP headers. A 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 [13] using the following packet header fields: source IP, destination IP, source port, destination port, protocol type and TCP flags. Normal intervals for the six fields are learned from 5 days of training data. In the test data, fields' values not falling in the learned intervals are flagged as suspect. Then the top n packet score values are termed as anomalous. The value of n is varied over a range to obtain ROC curves.

3.1.6 PCA-based subspace method [16]

Uses Principal Component Analysis (PCA) to separate a link's traffic measurement space into useful subspaces for analysis, with each subspace representing either benign or anomalous traffic behavior. The authors proposed to apply PCA for domain reduction of the Origin-Destination (OD) flows in three dimensions: number of bytes, packets, IP-level OD flows. The top k eigenvectors represent normal subspaces. It has been shown that most of the variance in a link's traffic is generally captured by 5 principal components [16]. A recent study showed that the detection rate of PCA varies with the level and method of aggregation [52]. It was also concluded in [52] that it may be impractical to run a PCA-based anomaly detector over data aggregated at the level of OD flows. We evaluate the subspace method using the number of TCP flows aggregated in 10 min intervals. To generate ROC results, we changed the number of normal subspace as $k = 1, 2, \ldots, 15$. Since the principal components capture maximum variance of the data, as we increase k, the dimension of the residual subspace reduces and fewer observations are available for detection. In other words, as more and more principal components are selected as normal subspaces, the detection and false alarm rates decrease proportionally. Since there is no clear detection threshold, we could not obtain the whole range of ROC values for the subspace method. Nevertheless, we evaluate and report the subspace method's accuracy results for varying number of principal components.

3.1.7 Kalman filter based detector [19]

First filters out the normal traffic from the aggregate traffic, and then examines the residue for anomalies. In [19], the Kalman Filter operated on SNMP data to detect anomalies traversing multiple links. Since SNMP data was not available to us in either dataset, we model the traffic as a 2-D vector X_t. The first element of X_t is the total number of sessions (in the endpoint dataset) or packets (in the LBNL dataset), while the second element is the total number of distinct remote ports observed in the traffic. We defined a threshold, η_f on the residue value r to obtain ROC curves. Thresholding of r is identical to the rate limiting case. An alarm is raised, if $r < -\eta_f$ or $r > \eta_f$.

3.1.8 Next-generation intrusion detection expert system (NIDES) [22]

Is a statistical anomaly detector that detects anomalies by comparing a long-term traffic rate profile against a short-term, real-time profile. An anomaly is reported if the Q distribution of the real-time profile deviates considerably from the long-term values. After specific intervals, new value of Q are generated by monitoring the new rates and compared against a predefined threshold, η_s. If $\Pr(Q > q) < \eta_s$, an alarm is raised. We vary η_s over a range of values for ROC evaluation.

3.1.9 Discussion

The evaluated ADSs are quite diverse in their traffic features as well as their detection frameworks. These ADSs range from very simple rule modeling systems like Packet Header Anomaly Detection (PHAD) [13] to very complex and theoretically-inclined Self-Learning systems like the PCA-based subspace method [16] and the Sequential Hypothesis Testing technique [8]. However, all the anomaly detection techniques evaluated in this study are statistical anomaly detectors. Thus, from an attacker's perspective, whenever the detection technique is known, an attack can be built that can defeat the ADS detection [54,55]. For example, NIDES and rate limiting detectors rely on traffic rates for the detection of an anomaly. When this information is known to an attacker, it can build a malware that spreads at a rate lower than the threshold rate being analyzed by the ADS. Consequently, such a malware will go undetected.

3.2 Evaluation datasets

We wanted to use real, labeled and public background and attack datasets to measure the accuracy of the evaluated

Table 1 Background traffic information for the LBNL dataset

Date	Duration (mins)	LBNL hosts	Remote hosts	Backgnd rate (pkt/sec)	Attack rate (pkt/sec)
10/04/04	10	4,767	4,342	8.47	0.41
12/15/04	60	5,761	10,478	3.5	0.061
12/16/04	60	5,210	7,138	243.83	72

anomaly detectors. Real and labeled data allow realistic and repeatable quantification of an anomaly detector's accuracy, which is a main objective of this work. Moreover, as defined in Sect. 1, another objective is to evaluate the accuracy or scalability of the anomaly detectors under different normal and attack traffic rates and at different deployment points in the network. This evaluation objective is somewhat unique to this effort, with [27] being the only other study that provides some insight into host versus edge deployments.

Different network deployment points are responsible for handling traffic from varying number of nodes. For instance, an endpoint requires to cater for only its own traffic, while an edge router needs to monitor and analyze traffic from a variety of hosts in its subnet. In general, as one moves away from the endpoints towards the network core, the number of nodes, and consequently the traffic volume, that a network entity is responsible for increase considerably. We argue that if an algorithm that is designed to detect high- or low-rate attacks at a particular point of deployment, say an edge router, scales to and provides high accuracy at other traffic rates and deployment points, say at endpoints, then such an algorithm is quite valuable because it provides an off-the-shelf deployment option for different network entities. (We show later in this study that some existing algorithms are able to achieve this objective.)

To test the anomaly detectors for scalability, we use two real traffic datasets that have been independently-collected at different deployment points. The first dataset is collected at the edge router of the Lawrence Berkeley National Laboratory (LBNL), while the second dataset is collected at network endpoints by our research lab.[3] In this section, we describe the data collection setups and the attack and background traffic characteristics of the LBNL and the endpoint datasets.

3.2.1 The LBNL dataset

This dataset was obtained from two international network locations at the Lawrence Berkeley National Laboratory (LBNL) in USA. Traffic in this dataset comprises packet-level incoming, outgoing and internally-routed traffic streams at the LBNL edge routers. Traffic was anonymized using the `tcpmkpub` tool; refer to [47] for details of anonymization.

LBNL Background Traffic: LBNL data used in this study is collected during three distinct time periods. Some pertinent statistics of the background traffic are given in Table 1. The average remote session rate (i.e., sessions from distinct non-LBNL hosts) is approximately 4 sessions per second. The total TCP and UDP background traffic rate in packets per second is shown in column 5 of the table. A large variance can be observed in the background traffic rate at different dates. This variance will have an impact on the performance of volumetric anomaly detectors that rely on detecting bursts of normal and malicious traffic.

The main applications observed in internal and external traffic are Web (HTTP), Email and Name Services. Some other applications like Windows Services, Network File Services and Backup were being used by internal hosts; details of each service, information of each service's packets and other relevant description are provided in [48].

LBNL Attack Traffic: Attack traffic was isolated by identifying scans in the aggregate traffic traces. Scans were identified by flagging those hosts which unsuccessfully probed more than 20 hosts, out of which 16 hosts were probed in ascending or descending order [47]. Malicious traffic mostly comprises failed incoming TCP SYN requests; i.e., TCP portscans targeted towards LBNL hosts. However, there are also some outgoing TCP scans in the dataset. Most of the UDP traffic observed in the data (incoming and outgoing) comprises successful connections; i.e., host replies are received for the UDP flows. Table 1 [column 6] shows the attack rate observed in the LBNL dataset. Clearly, the attack rate is significantly lower than the background traffic rate. Thus these attacks can be considered low rate relative to the background traffic rate. (We show later that background and attack traffic at endpoints exhibit the opposite characteristics.)

Since most of the anomaly detectors used in this study operate on TCP, UDP and/or IP packet features, to maintain fairness we filtered the background data to retain only TCP and UDP traffic. Moreover, since most of the scanners were located outside the LBNL network, to remove any bias we filter out internally-routed traffic. After filtering the datasets, we merged all the background traffic data at different days

[3] We also wanted to use a traffic dataset collected at a backbone ISP network; such datasets have been used in some prior studies [16–18]. However, we could not find a publicly available ISP traffic dataset.

Table 2 Background traffic information for four endpoints with high and low rates

Endpoint ID	Endpoint type	Duration (months)	Total sessions	Mean session rate (/sec)
3	Home	3	373,009	1.92
4	Home	2	444,345	5.28
6	Univ	9	60,979	0.19
10	Univ	13	152,048	0.21

and ports. Synchronized malicious data chunks were then inserted in the merged background traffic.

3.2.2 Endpoint dataset

Since no publicly-available endpoint traffic set was available, we spent up to 14 months in collecting our own dataset on a diverse set of 13 endpoints. Complexity and privacy were two main reservations of the participants of the endpoint data collection study. To address these reservations, we developed a custom tool for endpoint data collection. This tool was a multi-threaded MS Windows application developed using the `Winpcap` API [49]. (Implementation of the tool is available at [26].) To reduce the packet logging complexity at the endpoints, we only logged some very elementary session-level information of TCP and UDP packets. Here a *session* corresponds to a bidirectional communication between two IP addresses; communication between the same IP address on different ports is considered part of the same network session. To ensure user privacy, the source IP address (which was fixed/static for a given host) is not logged, and each session entry is indexed by a one-way hash of the destination IP with the hostname. Most of the detectors evaluated in this work can operate with this level of data granularity.

Statistics of the two highest rate and the two lowest rate endpoints are listed in Table 2.[4] As it can be intuitively argued, the traffic rates observed at the endpoints are much lower than those at the LBNL router. In the endpoint context, we observed that home computers generate significantly higher traffic volumes than office and university computers because: (1) they are generally shared between multiple users, and (2) they run peer-to-peer and multimedia applications. The large traffic volumes of home computers are also evident from their high mean number of sessions per second. For this study, we use 6 weeks of endpoint traffic data for training and testing. Results for longer time periods were qualitatively similar.

To generate attack traffic, we infected Virtual Machines (VMs) on the endpoints by the following malware: `Zotob.G`, `Forbot-FU`, `Sdbot-AFR`, `Dloader-NY`, `SoBig.E@mm`, `MyDoom.A@mm`, `Blaster`, `Rbot-AQJ`, and `RBOT.CCC`; details of the malware can be found at [50]. These malware have diverse scanning rates and attack ports/applications. Table 3 shows statistics of the highest and lowest scan rate worms; `Dloader-NY` has the highest scan rate of 46.84 scans per second (sps), while `MyDoom-A` has the lowest scan rate of 0.14 sps, respectively. For completeness, we also simulated three additional worms that are somewhat different from the ones described above, namely `Witty`, `CodeRedv2` and a fictitious TCP worm with a fixed and unusual source port. `Witty` and `CodeRedv2` were simulated using the scan rates, pseudocode and parameters given in research and commercial literature [50,51].

Table 3 Endpoint attack traffic for two high- and two low-rate worms

Malware	Release Date	Avg. Scan Rate(/sec)	Port(s) Used
`Dloader-NY`	Jul 2005	46.84 sps	TCP 135,139
`Forbot-FU`	Sept 2005	32.53 sps	TCP 445
`MyDoom-A`	Jan 2006	0.14 sps	TCP 3127 − 3198
`Rbot-AQJ`	Oct 2005	0.68 sps	TCP 139,769

Endpoint Background Traffic: The users of these endpoints included home users, research students, and technical/administrative staff. Some endpoints, in particular home computers, were shared among multiple users. The endpoints used in this study were running different types of applications, including peer-to-peer file sharing software, online multimedia applications, network games, SQL/SAS clients etc.

Endpoint Attack Traffic: The attack traffic logged at the endpoints mostly comprises outgoing portscans. Note that this is the opposite of the LBNL dataset, in which most of the attack traffic is inbound. Moreover, the attack traffic rates (Table 3) in the endpoint case are generally much higher than the background traffic rates (Table 2). This characteristic is also the opposite of what was observed in the LBNL dataset. This diversity in attack direction and rates provides us a sound basis for performance comparison of the anomaly detectors evaluated in this study [8,9].

For each malware, attack traffic of 15 min duration was inserted in the background traffic of each endpoint at a random time instance. This operation was repeated to insert 100 non-overlapping attacks of each worm inside each endpoint's background traffic.

[4] The mean session rates in Table 2 are computed using time-windows containing one or more new sessions. Therefore, dividing total sessions by the duration does not yield the session rate of column 5.

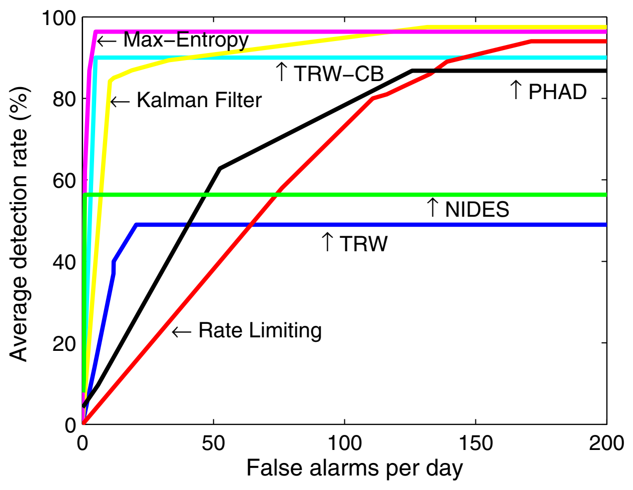


Fig. 1 ROC analysis on the endpoint dataset; each ROC is averaged over 13 endpoints with 12 attacks per endpoint and 100 instances per attack

4 Performance evalaution

In this section, we evaluate the accuracy, scalability and delay of the anomaly detectors described in the last section on the endpoint and router datasets.

4.1 Accuracy and scalability comparison

In this section, we present ROC analysis on the endpoint dataset. The following section explains the scalability experiments in which ROC analysis is performed on the LBNL dataset and the results are compared with the endpoint experiments.

4.1.1 Averaged ROCs for the endpoint dataset

Figure 1 provides the averaged ROC analysis of the anomaly detection schemes under consideration. Clearly, the Maximum Entropy detector provides the highest accuracy by achieving near 100% detection rate at a very low false alarm rate of approximately 5 alarms/day. The Maximum Entropy detector is followed closely by the credit-based TRW approach. TRW-CB achieves nearly 90% detection rate at a reasonable false alarm rate of approximately 5 alarms/day. The original TRW algorithm, however, provides very low detection rates for the endpoint dataset. Results of these three schemes are shown more clearly in Fig. 2a. Based on these results, the Maximum Entropy algorithm provides the best accuracy on endpoints, while TRW provides the best detection on LBNL dataset.

The Kalman Filter approach is also quite accurate as it provides up to 85% detection rates at a reasonably low false alarm cost. Rate Limiting, although designed to detect outgoing scanning attacks, provides very poor performance. This result substantiates the results of [27] where very high false positive rates for high detection rates were reported for classical rate limiting. Hence, we also deduce that rate limiting is ineffective for portscan detection at endpoints.

Packet Header Anomaly Detection does not perform well on the endpoint data set. The detection is accompanied with very high false alarm rates. NIDES achieve reasonable detection rates at very low false alarm rates, but is unable to substantially improve its detection rates afterwards. PHAD relies on previously seen values in the training dataset for anomaly detection. Therefore, if a scanner attacks a commonly-used port/IP then PHAD is unable to detect it. On similar grounds, if the malicious traffic is not bursty enough as compared to background traffic then NIDES will not detect it, irrespective of how much the detection threshold is tuned.

Due to the thresholding difficulties for the subspace method explained in Sect. 3, in Fig. 3, we report results for this technique for varying values of selected principal components. The highest detection rate of 22% is observed at $k = 2$ principal components. Its detection rate decreases further at $k = 5$ and drops to 0% at $k = 15$. False alarm rates show the opposite trend. Thus the subspace method fails to give acceptable accuracy on the endpoint dataset.

The ROC results for the endpoint dataset are somewhat surprising because two of the top three detectors are general-purpose anomaly detectors (Maximum Entropy and Kalman Filter), but still outperform other detectors designed specifically for portscan detection, such as the TRW and the Rate Limiting detectors. We, however, note that this analysis is not entirely fair to the TRW algorithm because TRW was designed to detect incoming portscans, whereas our endpoint attack traffic contains mostly outgoing scan packets. The credit-based variant of TRW achieves high accuracy because it leverages outgoing scans for portscan detection. Thus TRW-CB combines the complementary strengths of rate limiting and TRW to provide a practical and accurate portscan detector for endpoints. This result agrees with earlier results in [27].

4.1.2 ROCs for Low- and High-Rate endpoint attacks

To evaluate the scalability of the ADSs under high- and low-rate attack scenarios, Fig. 4 plots the ROCs for the highest rate (`Dloader-NY`) and lowest rate (`MyDoom-A`) attacks in the endpoint dataset. It can be observed that for the high-rate attack (Fig. 4a) Maximum Entropy, TRW, TRW-CB and Kalman Filter techniques provide excellent accuracy by achieving 100% or near-100% detection rates with few false alarms. NIDES' performance also improves as it achieves approximately 90% detection rate at very low false alarm rates. This is because the high-rate attack packets form bursts of malicious traffic that NIDES is tuned to detect.

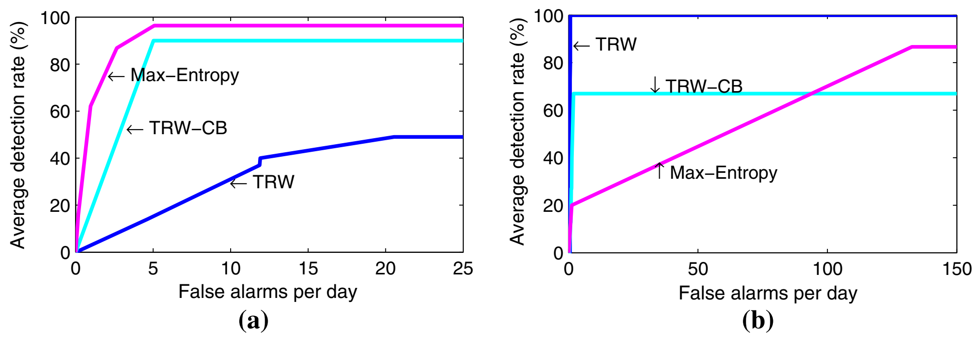


Fig. 2 Comparison of the Maximum Entropy, TRW and TRW-CB algorithms. **a** Endpoint dataset. **b** LBNL dataset

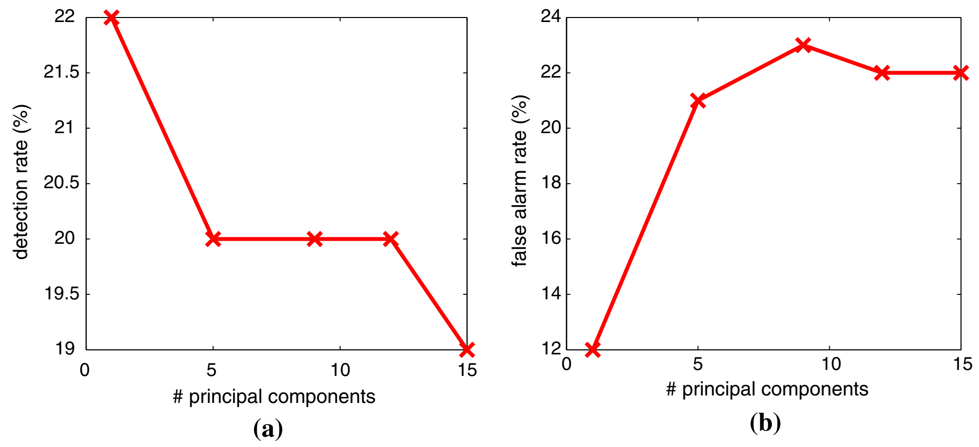


Fig. 3 Detection and false alarm rates for the subspace method [16]. **a** Detection rate. **b** False alarm rate

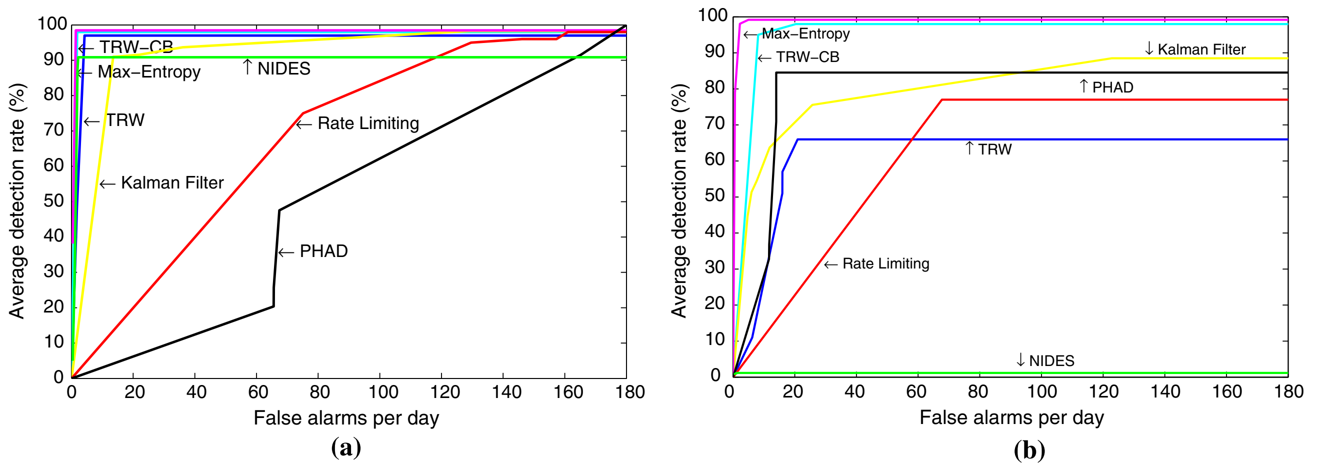


Fig. 4 ROC curves for the lowest and highest rate attack in the endpoint dataset; results averaged over 12 endpoints with 100 instances of each attack. **a** `Dloader-NY`, high scan rate. **b** `MyDoom-A`, low scan rate

Rate Limiting and PHAD do not perform well even under high attack rate scenarios.

Figure 4b shows that the accuracies of all detectors except PHAD and Maximum Entropy degrade under a low-rate attack scenario. Maximum Entropy achieves 100% detection rate with false alarm rate of 4–5 alarms/day. TRW-CB recovers quickly and achieves a near-100% detection rate for a daily false alarm rate around 10 alarms/day. NIDES, however, shows the biggest degradation in accuracy as its detection rate drops by approximately 90%. This is because low-rate attack traffic when mixed with normal traffic does not result in long attack bursts. TRW's accuracy is also affected significantly as its detection rate drops by about 35% as compared to the high-rate attack. PHAD does not rely on traffic rate for detection, and hence its accuracy is only dependent on the header values observed during training.

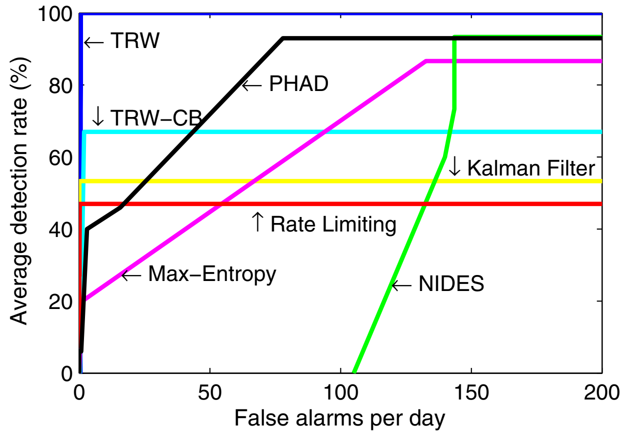


Fig. 5 ROC analysis on the LBNL dataset

4.1.3 Averaged ROCs for the LBNL dataset

Figure 5 shows the ROCs for the LBNL dataset. Comparison with Fig. 2a and b reveals that the Maximum Entropy detector is unable to maintain its high accuracy on the LBNL dataset; i.e., the Maximum Entropy algorithm cannot scale to different points of network deployment. TRW's performance improves significantly as it provides a 100% detection rate at a negligible false alarm cost. TRW-CB, on the other hand, achieves a detection rate of approximately 70%. Thus contrary to the endpoint dataset, the original TRW algorithm easily outperforms the TRW-CB algorithm on LBNL traces. As explained in Sect. 3.2.1, the LBNL attack traffic mostly comprises failed incoming TCP connection requests. TRW's forward sequential hypothesis based portscan detection algorithm is designed to detect such failed incoming connections, and therefore it provides high detection rates. Thus on an edge router, TRW represents a viable deployment option.

Kalman Filter detector's accuracy drops as it is unable to achieve a detection rate above 60%. PHAD provides very high detection rates, albeit at an unacceptable false alarm rate. Other detectors' results are similar to the endpoint case. (Results for the subspace method were similar to those reported earlier and are skipped for brevity.) It can be observed from Fig. 5 that all algorithms except TRW fail to achieve 100% detection rates on the LBNL dataset. This is because these algorithms inherently rely on the high burstiness and volumes of attack traffic. In the LBNL dataset, the attack traffic rate is much lower than the background traffic rate. Consequently, the attack traffic is distributed across multiple time windows, with each window containing very few attack packets. Such low density of attack traffic in the evaluated time-windows remains undetected regardless of how much the detection thresholds are decreased.

For clarity, we also provide the false alarm rates of the NADSs. Tables 4 and 5 give the detection and false alarms/day as well as the false alarm rate on the Endpoint and the LBNL datasets respectively. The tables provide the NADS accuracy at the best operating point on the ROC curve.

Table 4 presents a summary of the accuracy of the ADSs on the Endpoint dataset. The highest false alarm rate, on the Endpoint dataset, is 15% for the PHAD detector. PHAD follows the same trend when evaluated for false alarms/day. Rate limiting and Kalman filter follow closely with false alarm rates of approximately 12% and 11% respectively. Table 5 gives the accuracy results on the LBNL dataset. Maximum Entropy presents a false alarm rate of 77% and thus completely fails on the LBNL dataset as mentioned before. Another important insight, when evaluated on false alarm rate, is regarding TRW-CB which offers a false alarm rate of 62%. The LBNL dataset has mostly incoming scan packets towards the LBNL hosts, and only a few outgoing scans [47]. Since the TRW-CB detector detects outgoing scans, which are few, it offers low detection rate on the LBNL dataset. Also, since the amount of outgoing traffic is very small, a few false alarms/day translate into a very large percentage value. This renders the detector completely useless. Thus, evaluating detectors on false alarms/day as well as the false alarm rate provides a holistic view of the accuracy of the ADS. However, the best performing detectors on the Endpoint dataset (Maximum Entropy and TRW-CB detectors) completely fail to offer acceptable detection and false alarm rates on the LBNL dataset.

4.2 Delay comparison

Table 6 provides the detection delay for each anomaly detector. On the endpoint dataset, delay is reported for the highest and the lowest rate attacks, while on the LBNL dataset this delay is computed for the first attack that is detected by an anomaly detector. A delay value of ∞ is listed if an attack is not detected altogether. It can be observed that detection delay is reasonable (less than 1 s) for all the anomaly detectors except the Maximum Entropy detector which incurs very high detection delays. High delays are observed for the Maximum Entropy detector because it waits for perturbations in multiple time windows before raising an alarm. Among other viable alternatives, TRW-CB provides the lowest detection delays for all three experiments. Detection delay for the TRW is also reasonably low.

5 Lessons learnt

This section provides an outline of the objectives of the study and the deductions pertaining to these objectives. Moreover, promising portscan detection guidelines are proposed based on the NADS evaluation along with some experimental results, in the form of ROC curves, for the accuracy improvements realized due to the proposed guidelines.

Table 4 Accuracy of NADSs at the best operating point on Endpoint dataset

	Rate limiting	TRW	TRW-CB	Max entropy	NIDES	PHAD	Kalman filter
Detection rate (%)	80	49	90	96.4	56.37	86.77	85.03
False alarms/day	110.74	20.54	5.02	5.07	0.91	125.8	11.97
False alarm rate (%)	12.53	1.44	0.12	4.52	5.45	15.66	11.53

Table 5 Accuracy of NADSs at the best operating point on LBNL dataset

	Rate limiting	TRW	TRW-CB	Max entropy	NIDES	PHAD	Kalman filter
Detection rate (%)	47	100	67	86	93.33	93	53.33
False alarms/day	0.22	0.77	1.88	132	143.52	77.97	0.11
False alarm rate (%)	0.534	7.17	62.21	77.05	6.78	6.42	0.14

Table 6 Detection Delay of the Anomaly Detectors

	Rate limiting	TRW	TRW-CB	Max entropy	NIDES	PHAD	Subspace method	Kalman filter
MyDoom (msec)	310	510	40	215000	∞	900	79	377
Dloader-NY (msec)	140	320	20	56000	0.086	990	23	417
LBNL (msec)	660	660	290	86000	330	330	∞	800

5.1 Objectives of the study

In this study, we evaluated eight prominent network-based anomaly detectors using two portscan traffic datasets having complementary characteristics. These detectors were evaluated on accuracy, scalability and delay criteria. Based on the results of this research study, we now rephrase and summarize our deductions pertaining to the main objectives of this study:

- Which NADSs perform better than others? On the endpoint dataset, our evaluation shows that true anomaly detectors (Max Entropy, PHAD, Rate Limiting etc.) provides the best accuracy. However, on the router based LBNL dataset, we observe that the protocol level, preprogrammed NADSs (TRW, TRW-CB) outperform the true anomaly detectors. Thus the protocol level Maximum Entropy detector is unable to maintain its high accuracy on the LBNL dataset.
- Which algorithms provide the best accuracy under varying rates of attack and normal traffic and at different points of deployment? Under the varying attack and background traffic rates observed in the two datasets, a general-purpose Maximum Entropy Detector [21] and variants of the Threshold Random Walk (TRW) algorithm [8,9] provided the best overall performance under most evaluation criteria. In this context, TRW is suitable for deployment at routers, while TRW-CB and Maximum Entropy are suitable for deployment at endpoints.
- What are the promising traffic features and theoretical frameworks for portscan anomaly detection? The Maximum Entropy and TRW detectors use statistical distributions of failed connections, ports and IP addresses. Furthermore, based on the results of the Maximum Entropy detector on endpoints, a histogram-based detection approach, in which baseline frequency profiles of a set of features is compared with real-time feature frequencies, appears very promising.
- What detection delays are incurred by the anomaly detectors? If an attack is detected, detection delay is less than 1 s for all anomaly detectors, except the Maximum Entropy Estimation method which incurs very large delays.
- What are promising portscan detection guidelines that build on the strengths and avoid the weaknesses of the evaluated anomaly detectors? From the high detection rates of the Maximum Entropy and PHAD detectors, it appears that using a higher dimensional feature space facilitates detection, without compromising complexity. On the other hand, relying on specific traffic features (e.g., rate, connection failures, etc.) can degrade accuracy as the attack and background traffic characteristics change. In summary, a number of statistical features used in an intelligent histogram-based classification framework appear promising for portscan anomaly detection.

5.2 Why do some NADSs perform better than others?

Based on the accuracy results, protocol level true anomaly detectors provide the best accuracy on endpoints, while the protocol level preprogrammed NADSs provides the best detection on the LBNL dataset. Both the classes are based on packet level information. Since high-rate malicious packets affect an increase in the frequency of certain packet headers (e.g., the ports being attacked in case of a worm, or the IP being attacked in case of a DoS attack), attacks can be detected most accurately if the protocol level information is analyzed at different aggregation levels.

At the endpoints, the volume of traffic is much lesser than that at the router. Moreover, the endpoint behavior changes with time as different applications are used by the endhost. Thus NADSs that are used for deployment at endpoints need to train on benign data to cater for the changing user behaviors. That is why the true self learning NADSs, that trained on benign profiles provide the best accuracy on the endpoint dataset.

A router-based NADS analyzes large quantities of data and consequently the attack traffic gets averaged out in the normal traffic. Thus training on benign profiles would not render higher detection rates, as can be seen by the results of NADSs on LBNL dataset. However, intelligent statistical measures (e.g., the likelihood ratio test used by TRW) improve the NADS' accuracy. Thus to provide better accuracy on enterprise/network level traffic, selection of the right detection method/principle is more important to achieve higher detection rates than training on benign data.

In light of the accuracy evaluation results, Maximum Entropy provides best detection and false alarm rates on individual basis because of the following inbuilt characteristics:

- It segregates traffic into multiple packet classes;
- Analyzes a high dimensional feature space;
- Generates an alarm when anomalies span across multiple time windows.

PHAD detector operates on similar principles and thus also provides high detection rates. In all datasets we observe that traffic rates keep changing. While all NADSs apply fixed thresholds to classify anomalies in real-time traffic, an accurate NADS should vary its classification thresholds with respect to the changing patterns in benign traffic.

5.3 Promising guidelines to improve the accuracy of existing and future NADSs

Based on above discussion, we propose the following guidelines to improve the accuracy of NADSs:

Guideline 1 To provide high detection rates, endpoint based NADSs should be self learning systems which train on benign traffic profiles.

Guideline 2 To provide high detection rates, router based NADSs should employ intelligent detection principles for identifying anomalies in network traffic.

Guideline 3 NADSs should operate on protocol-level packet features.

Guideline 4 To reduce the false alarm rates, NADSs should raise an alarm only when they encounter anomalies spanning across multiple time windows.

Guideline 5 To improve the detection rates, NADSs should simultaneously consider multiple packet header fields, e.g. TCP SYN, Ports, Protocol etc.

Guideline 6 To improve detection rates, NADSs should segregate traffic into multiple packet classes before anomaly detection.

Guideline 7 Adaptive thresholding should be introduced to allow the NADSs to dynamically adjust their detection thresholds in accordance with the changing normal traffic characteristics.

It is important to clarify that these proposed guidelines, though complement each other, but each operate independently. Guidelines 4–7 aim at improving the accuracy of the anomaly detection system as well as reduce human intervention in their operation. Following is a detailed description of these guidelines and the accuracy improvements achieved:

5.3.1 Multi-window classification (Guideline 4)

We have seen in the comparative evaluation study of the NADSs, that most NADSs suffer from high false alarm rates. The problem mainly stems from the fact that most NADSs raise an alarm as soon as the first anomalous time window is identified. We observed that, due to an inherent burstiness present in attack traffic, anomalies tend to sustain across multiple time windows. An example of this behavior is shown in Fig. 6. In Fig. 6a, even if a false alarm is generated for a benign traffic window, the false alarm does not span multiple windows. On the other hand, anomalous activity tends to occur in bursts, and therefore multiple successive windows are flagged as anomalous. This difference between NADS classification on malicious and benign traffic can be leveraged to reduce an NADS' false alarm rate. Specifically, an NADS can reduce its false alarms if it raises an alarm only after sufficient number of anomalous time windows have been observed in a given time period. We call this simple existing technique *Multi-Window Classification*.

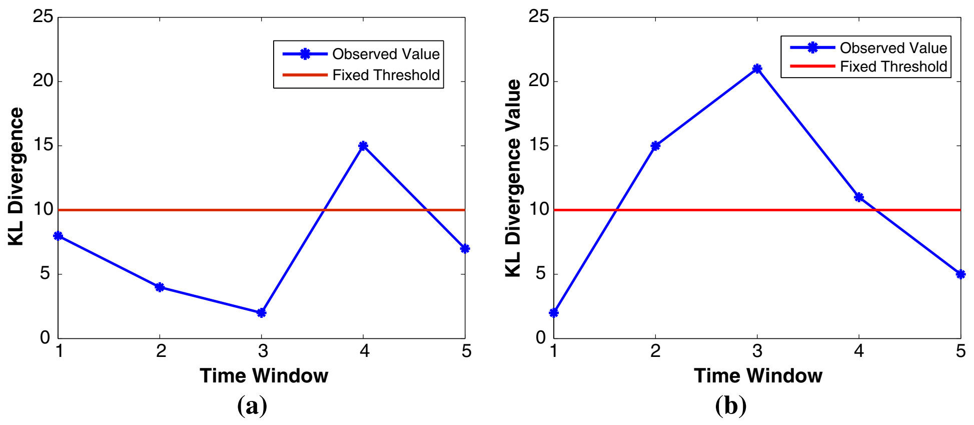


Fig. 6 An example of multi-window classification: Maximum-Entropy Detector's output on five LBNL time windows containing benign and malicious traffic. **a** Windows with Benign Traffic. **b** Windows with Malicious Traffic

For accurate multi-window classification, we consider a fixed number of w most recent classifications by an NADS. In the w classifications, a majority vote is taken to classify the current time window as benign or anomalous. It should be highlighted that multi-window classification will introduce detection delays in the NADSs. However, as already shown, detection delays of most existing NADSs are extremely low and hence these NADSs can tolerate slightly longer detection delays to achieve higher accuracies.

5.3.2 Feature space extension (Guideline 5)

We have seen in the comparative performance evaluation that Maximum Entropy and PHAD are the highest accuracy detectors. Both these detectors employ a rich feature space for detection. Thus greater the number of packet fields analyzed for anomaly detection, higher the probability of finding the anomaly. Thus, if within one time window, instead of analyzing a few packet header fields, the maximum available fields are analyzed, its highly probable that the NADS finds an anomaly that perturbs any of the observed packet feature.

Figure 7 shows the distribution of the packet score calculated for each packet based on the packet header fields analyzed. In Fig. 7a PHAD detector computes the packet score based on a single anomalous packet header field. Figure 7b shows the packet score distribution, for PHAD, when multiple packet header fields are simultaneously used for packet score calculation. In Fig. 7a, since packet score does not exceed the specified threshold value, PHAD detector fails to detect the anomalies which are otherwise detected if diverse packet features are analyzed as shown in Fig. 7b. Thus, using a rich feature space assists the detection of anomalies that perturb any network traffic feature resulting in high detection rates for the NADSs.

5.3.3 Traffic splitting (Guideline 6)

Traffic Splitting is also aimed at improving an NADS's detection rate.

Our preliminary investigation revealed that much of the malicious traffic is not detected because of an averaging-out effect introduced by relatively large volumes of benign background traffic. More specifically, if the attack traffic rate is comparable to or less than the background traffic rate then background traffic acts like noise during anomaly detection and allows malicious traffic to bypass an NADS.

As an example, note that aggregate traffic routed towards/from a network is a composite of multiple traffic types, such as TCP, UDP, ARP, ICMP traffic etc. Now consider the Witty worm which was a very high-rate UDP-based worm. Since TCP comprises of almost 80% of the traffic seen on the Internet, if an NADS analyzes aggregate network traffic then the attack traffic is overwhelmed by the majority TCP traffic. Such traffic averaging degrades the detection rate of an NADS. Note that in this example traffic other than UDP acts as noise and, depending upon whether the volume of background traffic is substantial, would either delay or prevent the detection of the anomaly.

To counter this problem, we propose to perform anomaly detection separately on different types of network traffic. Hence, we use traffic semantics to segregate a single traffic stream into multiple traffic substreams before anomaly detection is performed. Such *traffic splitting* will inherently allow the background traffic to be segregated from the attack traffic, thereby facilitating the anomaly detection phase. After traffic splitting, separate instances of an NADS operate on different substreams in parallel. The outputs of these NADS instances are combined to detect anomalies. Based on the example given above, traffic splitting should, in addition to improving detection rates, reduce detection delays.

As a proof-of-concept, in Fig. 8 shows a comparison of aggregate traffic with segregated TCP and UDP traffics. Both of these anomalous windows were observed and analyzed in the Threshold Random Walk Credit Based (TRW-CB) algorithm under RBOT.CCC's and Witty's malicious traffic. TRW-CB calculates the likelihood ratio for detecting anomalies. It is clear from the Fig. 8 that when aggregate traffic is analyzed without segregation, the output of the likelihood

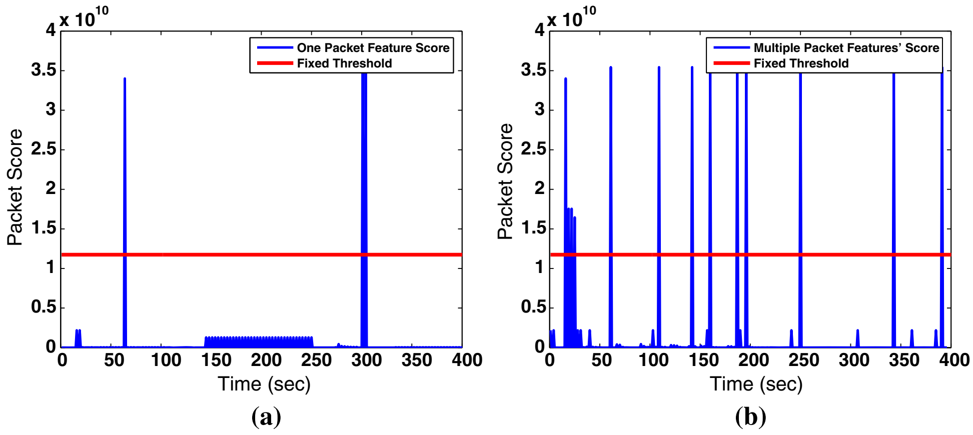


Fig. 7 PHAD Detector's output for a single and multiple features analyzed for anomaly detection on LBNL dataset. **a** One packet header field analyzed. **b** Multiple packet header field analyzed

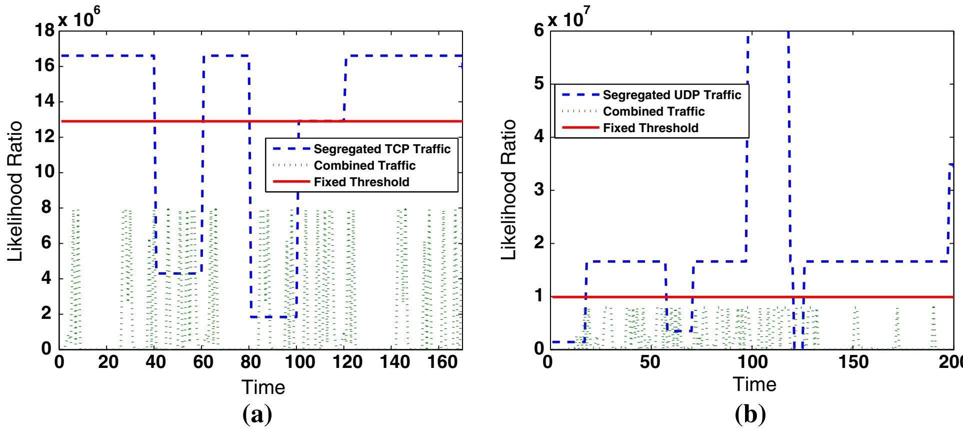


Fig. 8 An example of traffic splitting: segregated and aggregate traffic rates during a TCP and a UDP attack. **a** TCP attack. **b** UDP attack

ratio test does not cross the fixed TRW-CB threshold and the malicious traffic remains undetected for both examples. However, when traffic splitting is employed and TCP and UDP traffic is analyzed separately, the threshold is exceeded many times in the 200 s windows shown in Fig. 8a and b. Hence traffic splitting removes the noisy background traffic from malicious traffic and subsequently increases the detection rate of an NADS.

5.3.4 Adaptive thresholding (Guideline 7)

Traffic characteristics vary considerably across different organizations. For instance, traffic characteristics of academic & research organizations are quite different from commercial enterprises. Similarly, different network deployment points are responsible for handling traffic from varying number of nodes. For instance, an endpoint requires to cater for only its own traffic, while an edge router needs to monitor and analyze traffic from a variety of hosts in its subnet. Even for the same network entity, traffic characteristics keep changing due to diurnal and other network usage patterns. As an example, consider the LBNL background traffic rates shown in Fig. 9 (*solid line*). It can be observed that the traffic rates change from approximately 500–10,000 pkts/s within a few seconds. Under such varying traffic characteristics, existing NADSs require regular manual intervention for accurate operation. More specifically, a system or network administrator is responsible for adjusting the sensitivity of the anomaly detectors when the number of false alarms (i.e., traffic classified as malicious but which is in fact benign) increases. This sensitivity is adjusted using detection thresholds which are used to flag an anomaly. However, this repeated manual input renders an NADS less automated and more prone to configuration errors.

We argue that an effective ADS should automatically detect varying traffic patterns and adjust its detection threshold in accordance with varying traffic characteristics. If accurate, such an adaptive thresholding mechanism can eliminate the need for human threshold tuning, thereby making an NADS more automated. Moreover, as a by-product adaptive thresholding should also improve the accuracy of an NADS by tracking normal traffic patterns. (Here accuracy is defined in terms of detection versus false alarm rates.) In this section, we propose adaptive thresholding techniques that can accurately track the changing behavior of network traffic.

Threshold Prediction: We use adaptive thresholding to track the values of the detection feature(s) that an NADS is

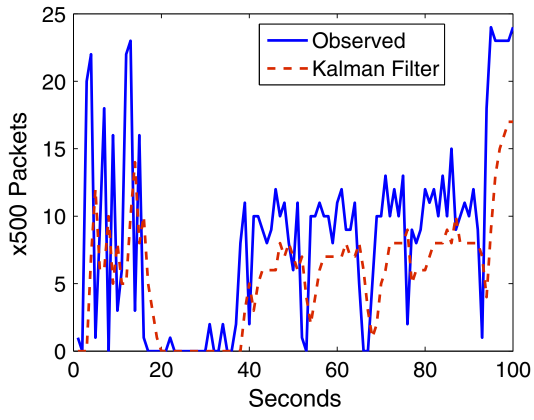


Fig. 9 Background traffic rate prediction for the LBNL dataset

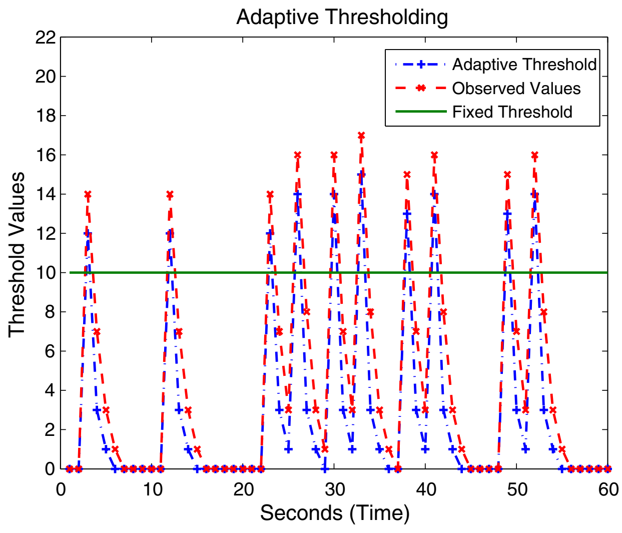


Fig. 10 Behavior of adaptive and fixed threshold in anomalous window

employing. For instance, in the Maximum-Entropy detector, the adaptive thresholding logic will use prior observed values to predict the next K-L divergence values of each traffic class, while in the TRW detector the output of the likelihood ratio test will be tracked by the adaptive thresholding module. Irrespective of the traffic metric being employed by an NADS, a good adaptive thresholding module should be able to predict the next value with high accuracy. To achieve accurate threshold prediction, we used a stochastic algorithm i.e., Kalman filter based detector [56]. Kalman filter based prediction is a well-known technique and was readily applied for adaptive thresholding. However, we divided the observed metrics/scores into equal sized bins (i.e. ranges). These bins were then predicted using kalman filter. This provides the range of values expected for the next time interval.

Since the main motivation for adaptive thresholding is to reduce human intervention by accurately tracking varying traffic characteristics. As an example, consider Fig. 9 which shows the traffic rates (pkts/sec) observed in a 100 s subset of the LBNL dataset and the rates predicted by kalman filter. For prediction, rates were divided into bins of $k = 500$ packets and predicted on per second basis. It can be seen in Fig. 9 that kalman predictor follows the observed rate trend.

Similarly, Fig. 10 shows the threshold tracking accuracy of the predictor in an anomalous LBNL time window of the Maximum-Entropy detector; the threshold in this case is the K-L divergence of a packet class that has been perturbed by the attack. It can be clearly seen that the kalman predictor estimates the highly-varying K-L divergence values with remarkable accuracy. Furthermore, note in Fig. 10 that selecting a fixed threshold may allow a few anomalies to go undetected, especially anomalies which do not cause significant perturbations in the actual network traffic. For instance, in the 60 s output shown in Fig. 10 only 10 of these values cross the fixed threshold. In this experiment, the Maximum-Entropy algorithm was detecting an anomaly if 12 or more values in a 60 s window exceed the fixed threshold. Hence, this anomaly will not be detected by a fixed threshold. Adaptive thresholding, on the other hand, accurately predicts the K-L divergence in the next window and the observed (perturbed) divergence exceeds this threshold more than 20 times in a 60 s window, thereby allowing the Maximum-Entropy detector to flag the anomaly. Furthermore, observe from Fig. 10 that for many seconds the observed K-L values drop to 0. These low values give a crafty attacker the leverage to introduce malicious traffic that does not exceed the fixed threshold of [0; 10]. However, an adaptive threshold immediately learns the change and set the threshold to 0, thus ensuring that no room is available for such a mimicry attack.

5.4 NADS accuracy improvements by traffic splitting and multi-window classification

We now jointly apply the two accuracy improvement techniques of traffic splitting and multi-window classification on the TRW and the TRW-CB detectors to observe the accuracy improvements that can be achieved. TRW and TRW-CB are evaluated to present the worst case scenario as both these detectors already incorporate some notion of traffic splitting. For traffic splitting, we segregate traffic into four classes: (1) UDP, (2) TCP, (3) broadcast, and (4) all other traffic types. For multi-window classification, we use a window size of $w = 5$; i.e., a majority vote of the last 5 NADS classifications is used to flag the current window as normal or anomalous.

It can be seen in Fig. 11a and b that the proposed techniques provide consistent accuracy improvements for TRW and TRW-CB on both datasets. On the LBNL dataset, while the detection rates are similar to the original algorithms, substantial reductions are observed in the false alarm rates. On the endpoint dataset, the proposed techniques effect significant improvements in both detection as well as false alarms rates as shown in Fig. 11a.

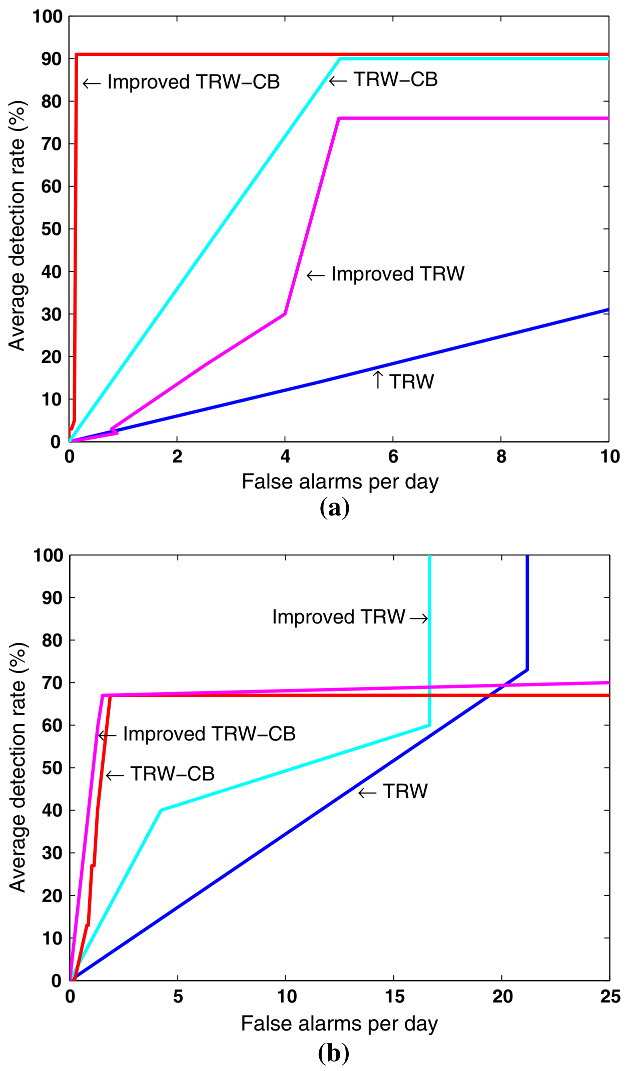


Fig. 11 ROC-based accuracy comparison of TRW and TRW-CB with and without traffic splitting and multi-window classification. **a** Endpoint dataset. **b** LBNL dataset

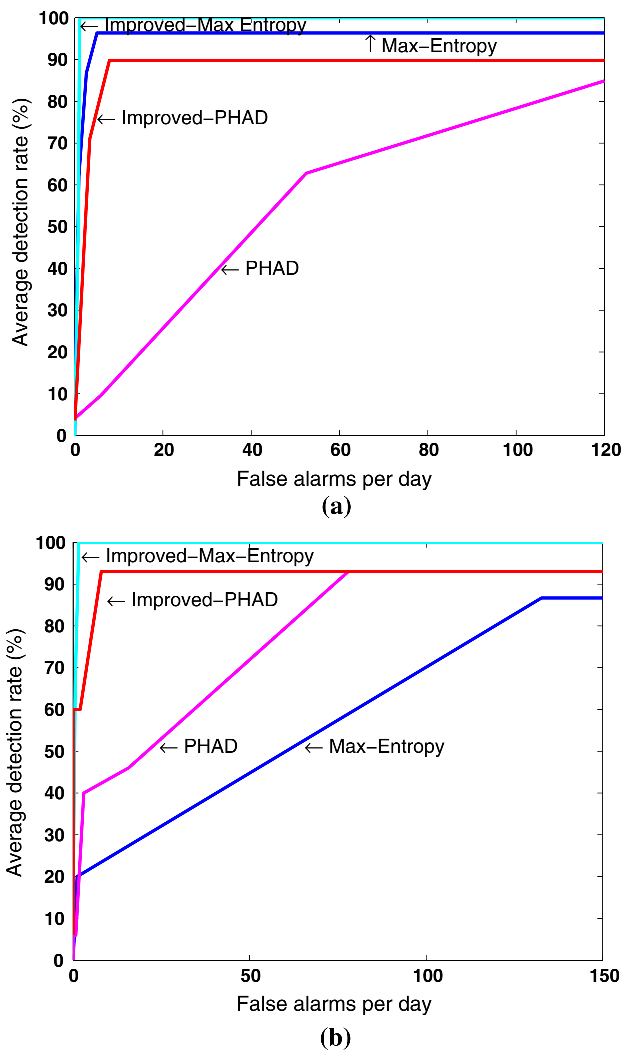


Fig. 12 ROC-based accuracy comparison of PHAD and Maximum Entropy with and without feature space extension and multi-window classification. **a** Endpoint dataset. **b** LBNL dataset

5.5 NADS accuracy improvements by feature space extension and multi-window classification

We evaluate two NADSs, namely the Maximum Entropy and PHAD detectors. These detectors already follow Guideline 5 because they operate on a high dimensional feature space. Furthermore, the Maximum Entropy detector already has the notion of time extension (Guideline 4) is built into it. More specifically, the Maximum Entropy detector segregates traffic into multiple protocol classes before the anomaly detection step and then raises an alarm only after an anomaly is observed in multiple time windows. Such a strategy results in very low false alarm rates, but compromises the detection rate. To make this algorithm compliant with Guideline 5, instead of waiting for multiple windows before making a decision, we modified the Maximum Entropy method to raise an alarm if the divergence of a given number of protocol classes in a time-window exceed a threshold.

Similarly, a time extension is introduced into PHAD to reduce its false alarm rate. Specifically, to make PHAD complaint with Guideline 4, an alarm is raised only when an anomaly is observed in multiple time windows/packets.

Figures 12a and b show a comparative analysis of the original and the Improved variants of PHAD and Maximum Entropy. It can be seen that evaluating PHAD across multiple time windows using a high dimensional feature space clearly improves the accuracy of the detector. Similarly, evaluating Maximum Entropy across its feature space instead of its original time domain design considerably improves the accuracy of the detector.

Evaluating NADSs in space and across multiple time windows might have an impact on the detection delay of the

Table 7 Detection Delay of the Improved variants of Maximum Entropy and PHAD

	ST-Max Entropy	ST-PHAD
`MyDoom` (msec)	157	900
`Dloader-NY` (msec)	100	990
LBNL (msec)	333	330

detectors. So we also perform delay comparison so as to observe the extent of the delay that guideline 4 can incur. Table 7 provides the detection delay for Maximum Entropy detector and PHAD. It can be observed that the detection delay for the Improved variant of Maximum Entropy detector is dramatically lower than the original algorithm. This is because the Improved variant of the Maximum Entropy detector does not wait for multiple anomalous windows before raising an alarm. For PHAD, the detection delay remains unaltered because the Improved variants simultaneously operates in space and time.

5.6 NADS accuracy improvements by adaptive thresholding

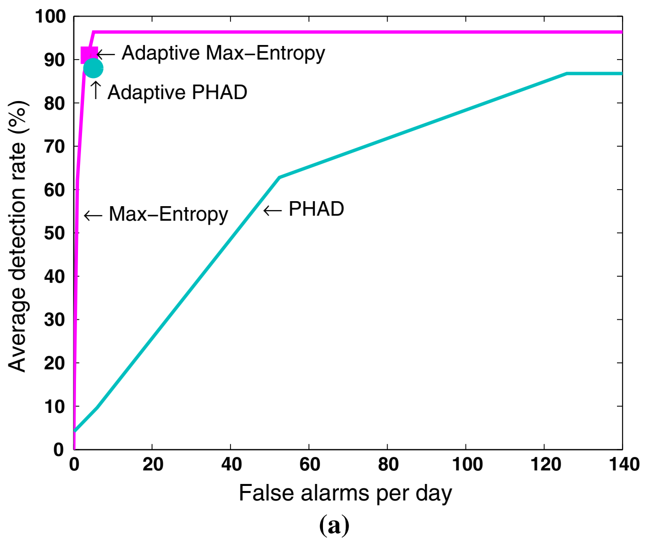
As mentioned earlier, one should also expect an accurate adaptive thresholding algorithm to provide accuracy improvements in an ADS. Accuracy of an ADS is defined by two competing criteria: (1) detection rate and (2) false alarm rate. To comprehensively evaluate the accuracy of an ADS, detection thresholds of the ADS are tuned and for each fixed threshold value the detection rate is plotted against the false alarm rate (per day) [4]. Each point on such a plot, referred to as an ROC curve [4], represents performance results for one configuration (or fixed threshold value) whereas the curve represents the behavior for the complete set of configurations. The steepest curve represents the best accuracy. Throughout this paper, we use ROC curves to evaluate the accuracy of the original algorithms. Due to a lack of threshold tuning capability, adaptive thresholding only results in a single point on the ROC plane. Figure 13 shows the ROC-based accuracy comparison of the Maximum Entropy and the PHAD detectors with and without adaptive thresholding. It can be clearly seen that for both datasets adaptive thresholding results in a dramatic improvement in PHAD's accuracy. In particular, PHAD originally had a very high false alarm rates, but the proposed technique adapts to the traffic trends and prevents the false alarms caused by legitimate change in traffic behavior. Significant improvement was not observed for the Maximum-Entropy detector on the endpoint dataset because even the original algorithm provided very high accuracy on the endpoints. However, Maximum Entropy failed to maintain its performance across the LBNL dataset because the significant traffic variations at an edge router introduce significant false alarms for the Maximum-Entropy detector. Even for the LBNL dataset, adaptive thresholding decreases this false alarm rate, but still cannot bring it to a reasonable point. We will show later that the multi-window classification technique that is designed to reduce false alarms can cater for these high false alarms.

In the following section, we highlight the accuracy improvements that can be realized by jointly applying the proposed guidelines.

5.7 NADS accuracy improvements by jointly using the proposed guidelines

We now argue that these described guidelines are complementary to each other and can be applied simultaneously to achieve higher accuracy while limiting the need for human intervention during NADS operation.

Figure 14 shows a block diagram of an NADS that jointly employs the proposed guidelines. The first thing we note is that none of the guidelines require modifications to the NADS. The Traffic Splitter is working as a pre-NADS phase which is segregating a single stream of traffic into multiple packet classes; these classes can be formed on any basis. The second phase, as shown in Fig. 14 is the feature space extension. Once traffic is segregated into multiple packet classes, each packet class is further segregated into multiple packet features to be examined for an anomaly. Each packet feature class is sent to a separate NADS instance which uses the threshold provided by the Adaptive Thresholding Module to classify traffic in the observed window as benign or anomalous. Outputs from multiple instances of the NADS, each analyzing a unique packet feature class, are combined into a single result and handed over to the Multi-Window Classifier. The Multi-Window Classifier acting as a post-NADS phase takes the majority vote of prior classification results to decide whether or not an alarm should be raised. Figure 15 shows the accuracy of five prominent NADSs along with the jointly-improved versions of these detectors after application of the proposed guidelines; all parameters are the same as described in previous sections. Due to a lack of threshold tuning capability, adaptive thresholding only results in a single point on the ROC plane. Figure 15a shows the accuracy comparison for the endpoint dataset. Maximum Entropy improves slightly on its already accurate performance. Jointly-improved Kalman Filter detector provides better accuracy than the original algorithm with a detection rates of approximately 96%. TRW-CB detector maintains similar accuracy as before, but with the additional advantage of ADS automation. PHAD and TRW detectors show dramatic improvements in accuracies as they achieve detection rate improvements of approximately 45 and 70%, respectively, without compromising their low false alarms rates. Note that although there was a slight degradation in



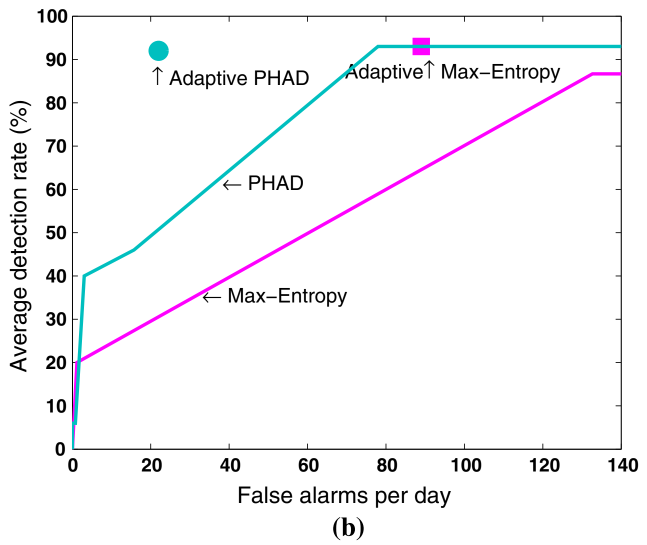


Fig. 13 ROC-based ADS accuracy comparison of adaptive thresholding and the original anomaly detection algorithms; for the endpoint dataset, each ROC point is averaged over 13 endpoints with 12 attacks per endpoint and 100 instances per attack. **a** Endpoint dataset. **b** LBNL dataset

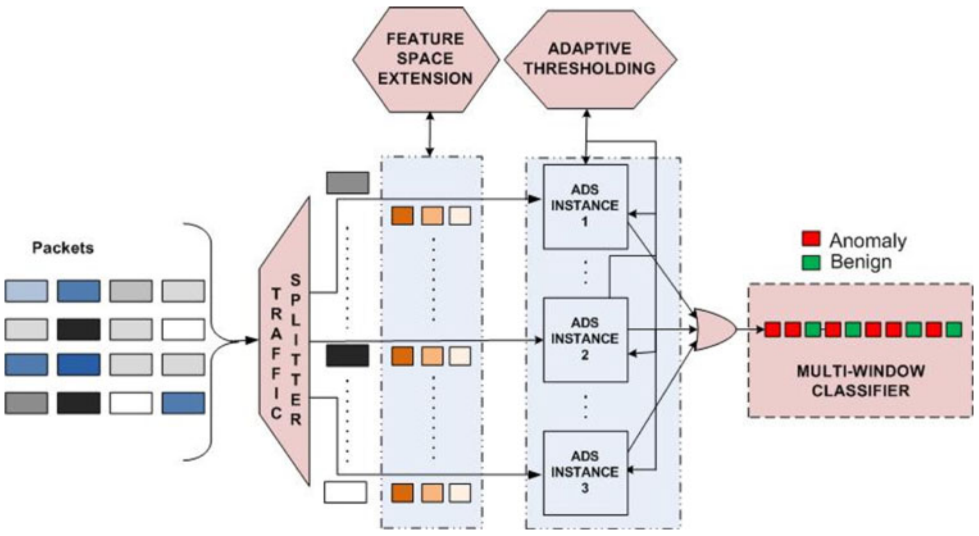


Fig. 14 Block diagram of Network-based Anomaly Detection System that jointly employs the proposed guidelines

the jointly-improved TRW's accuracy on the LBNL dataset, on the endpoint dataset the proposed techniques provide remarkable accuracy improvements for TRW. Thus the proposed guidelines, in addition to the benefits enumerated above, allow an NADS to scale to different points of deployment in the network.

From Fig. 15b, marked and mostly consistent improvements in all the NADSs' accuracies can be observed on the LBNL dataset. The Maximum-Entropy detector achieves a remarkable 100% detection rate at a reasonable false rate. Kalman Filter based detector's accuracy also improves drastically as its detection rate increases from 54 to 80% with few false alarms. A similar accuracy improvement trend is observed for the PHAD detector. No improvement in detection/false alarm rate is observed in the TRW-CB detector; however, it continues to provide the same detection rates without any human intervention and at an acceptable false alarm rate. The TRW detector is the only exception on the LBNL dataset as it incurs somewhat higher false alarms after using the proposed guidelines.

6 Conclusion

The main aim of this research work was to develop a better understanding of the basic building blocks of Network-based Anomaly Detection Systems and to identify the features that distinguish one NADS from the other. We initially propose two multidimensional taxonomies of Network-based Anomaly Detection Systems with an aim to obtain insights into why some NADSs perform better than others. Our first taxonomy classifies NADSs based on their learning behavior and detection principles. The second taxonomy categorizes NADSs using traffic scales and semantics.

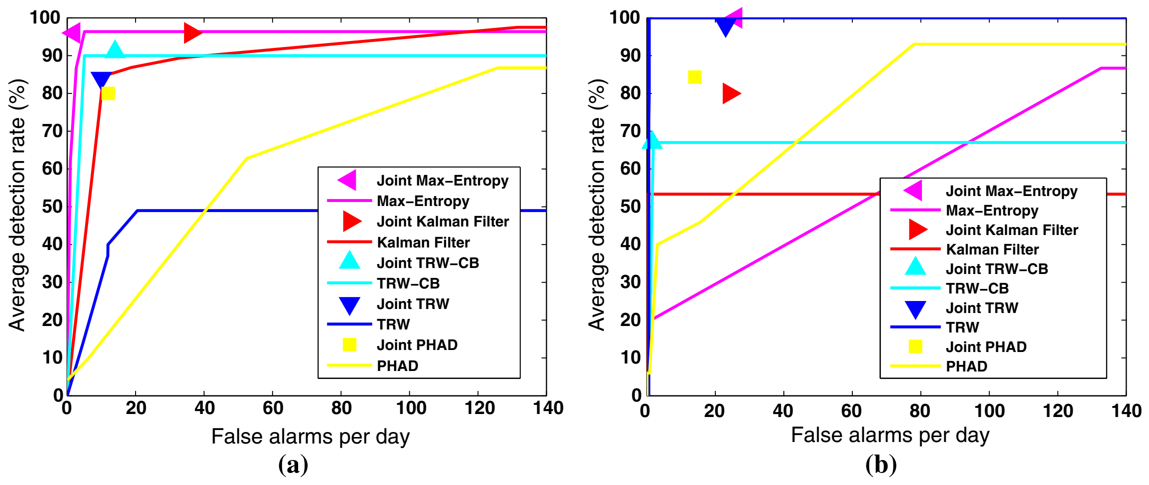


Fig. 15 ROC-based accuracy evaluation of original and improved NADS algorithms. **a** Endpoint dataset. **b** LBNL dataset

Moreover, we evaluated and compared eight prominent network based anomaly detectors on two independently collected public portscan datasets. These NADSs employed different traffic features and diverse theoretical frameworks for anomaly detection and have been used frequently for performance benchmarking in Intrusion Detection research literature.

NADSs were evaluated on three criteria: accuracy, scalability, and detection delay. Accuracy was evaluated by comparing ROC (false alarms per day versus detection rate) characteristics of the NADSs. Scalability was evaluated with respect to different background and attack traffic rates. Since the two datasets used in this study were collected at different network entities and contained attacks with different characteristics, evaluation over these datasets allowed us to compare the scalability of the proposed NADSs under varying traffic volumes. Detection delay was evaluated separately for high- and low-rate attacks.

Based on our findings, we proposed a few promising portscan detection guidelines to improve the accuracy and scalability of existing and future NADSs. Our experimental results showed that the proposed guidelines resulted in an average detection rate increase of 5–10%, while reducing the false alarm rates up to 50%. Thus, the proposed guidelines, while reducing human intervention, provided drastic and consistent improvements in NADSs' accuracies.

References

1. Symantec Internet Security Threat Reports I–XI (Jan 2002–Jan 2008)
2. Computer Economics: 2001 Economic impact of malicious code attacks, http://www.computereconomics.com/cei/press/pr92101.html
3. http://wisnet.niit.edu.pk/projects/adeval/Literature_survey/Bibliography.doc
4. Lippmann, R.P., Haines, J.W., Fried, D.J., Korba, J., Das, K.: The 1999 DARPA offline intrusion detection evaluation. Comp. Netw. **34**(2), 579–595 (2000)
5. Williamson, M.M.: Throttling viruses: restricting propagation to defeat malicious mobile code. In: ACSAC (2002)
6. Twycross, J., Williamson, M.M.: Implementing and testing a virus throttle. In: Usenix Security (2003)
7. Sellke, S., Shroff, N.B., Bagchi, S.: Modeling and automated containment of worms. In: DSN (2005)
8. Jung, J., Paxson, V., Berger, A.W., Balakrishnan, H.: Fast portscan detection using sequential hypothesis testing. In: IEEE Symp Sec and Priv (2004)
9. Schechter, S.E., Jung, J., Berger, A.W.: Fast detection of scanning worm infections. In: RAID (2004)
10. Weaver, N., Staniford, S., Paxson, V.: Very fast containment of scanning worms. In: Usenix Security (2004)
11. Chen, S., Tang, Y.: Slowing down internet worms. In: IEEE ICDCS (2004)
12. Ganger, G., Economou, G., Bielski, S.: Self-Securing network interfaces: what, why, and how. Carnegie Mellon University Technical Report, CMU-CS-02-144 (2002)
13. Mahoney, M.V., Chan, P.K.: PHAD: packet header anomaly detection for indentifying hostile network traffic. Florida Tech. technical report CS-2001-4 (2001)
14. Mahoney, M.V., Chan, P.K.: Learning models of network traffic for detecting novel attacks. Florida Tech. technical report CS-2002-08 (2002)
15. Mahoney, M.V., Chan, P.K.: Network traffic anomaly detection based on packet bytes. In: ACM SAC (2003)
16. Lakhina, A., Crovella, M., Diot, C.: Characterization of network-wide traffic anomalies in traffic flows. In: ACM Internet Measurement Conference (IMC) (2004)
17. Lakhina, A., Crovella, M., Diot, C.: Diagnosing network-wide traffic anomalies. In: ACM SIGCOMM (2004)
18. Lakhina, A., Crovella, M., Diot, C.: Mining anomalies using traffic feature distributions. In: ACM SIGCOMM (2005)
19. Soule, A., Salamatian, K., Taft, N.: Combining filtering and statistical methods for anomaly detection. In: ACM/Usenix IMC (2005)
20. Zou, C.C., Gao, L., Gong, W., Towsley, D.: Monitoring and early warning of internet worms. In: ACM CCS (2003)
21. Gu, Y., McCullum, A., Towsley, D.: Detecting anomalies in network traffic using maximum entropy estimation. In: ACM/Usenix IMC (2005)

22. Next-Generation Intrusion Detection Expert System (NIDES), http://www.csl.sri.com/projects/nides/
23. Peakflow-SP and Peakflow-X, http://www.arbornetworks.com/peakflowsp, http://www.arbornetworks.com/peakflowx
24. Cisco IOS Flexible Network Flow, http://www.cisco.com/go/netflow
25. LBNL/ICSI Enterprise Tracing Project, http://www.icir.org/enterprise-tracing/download.html
26. WisNet ADS Comparison Homepage, http://wisnet.seecs.edu.pk/projects/adeval
27. Wong, C., Bielski, S., Studer, A., Wang, C.: Empirical analysis of rate limiting mechanisms. In: RAID (2005)
28. Ingham, K.L., Inoue, H.: Comparing anomaly detection techniques for HTTP. In: RAID (2007)
29. Lazarevic, A., Ertoz, L., Kumar, V., Ozgur, A., Srivastava, J.: A comparative study of anomaly detection schemes in network intrusion detection. In: SIAM SDM (2003)
30. Mueller, P., Shipley, G.: Dragon claws its way to the top. In: Network Computing, http://www.networkcomputing.com/1217/1217f2.html (2001)
31. The NSS Group: intrusion detection systems group test (Edition 2) http://nsslabs.com/group-tests/intrusion-detection-systems-ids-group-test-edition-2.html (2001)
32. Yocom, B., Brown, K.: Intrusion battleground evolves, Network World Fusion, http://www.nwfusion.com/reviews/2001/1008bg.html (2001)
33. Durst, R., Champion, T., Witten, B., Miller, E., Spagnuolo, L.: Testing and evaluating computer intrusion detection systems. Comm. ACM **42**(7), 53–61 (1999)
34. Shipley, G.: ISS realsecure pushes past newer IDS players. In: Network Computing, http://www.networkcomputing.com/1010/1010r1.html (1999)
35. Shipley, G.: Intrusion Detection, Take Two. In: Network Computing, http://www.nwc.com/1023/1023f1.html (1999)
36. Roesch, M.: Snort—lightweight intrusion detection for networks. In: USENIX LISA (1999)
37. Lippmann, R.P., Fried, D.J., Graf, I., Haines, J.W., Kendall, K.R., McClung, D., Weber, D., Webster, S.E., Wyschogrod, D., Cunningham, R.K., Zissman, M.A.: Evaluating intrusion detection systems: the 1998 DARPA off-line intrusion detection evaluation. In: DISCEX, (2) pp. 12–26 (2000)
38. DARPA-sponsored IDS Evaluation (1998 and 1999) by MIT Lincoln Lab, www.ll.mit.edu/IST/ideval/data/data_index.html
39. Debar, H., Dacier, M., Wespi, A., Lampart, S.: A workbench for intrusion detection systems. IBM Zurich Research Laboratory (1998)
40. Denmac Systems, Inc.: Network based intrusion detection: a review of technologies (1999)
41. Ptacek, T.H., Newsham, T.N.: Insertion, evasion, and denial of service: eluding network intrusion detection. Secure Networks, Inc. (1998)
42. Aguirre, S.J., Hill, W.H.: Intrusion detection Fly-Off: implications for the United States Navy. MITRE Technical Report MTR 97W096 (1997)
43. Puketza, N., Chung, M., Olsson, R.A., Mukherjee, B.: A software platform for testing intrusion detection systems. IEEE Softw **14**(5), 43–51 (1997)
44. Puketza, N.F., Zhang, K., Chung, M., Mukherjee, B., Olsson, R.A.: A methodology for testing intrusion detection systems. IEEE Trans. Soft. Eng. **10**(22), 719–729 (1996)
45. McHugh, J.: The 1998 Lincoln laboratory IDS evaluation (A Critique). In: RAID (2000)
46. Mahoney, M.V., Chan, P.K.: An analysis of the 1999 DARPA/Lincoln laboratory evaluation data for network anomaly detection. In: RAID (2003)
47. Pang, R., Allman, M., Paxson, V., Lee, J.: The devil and packet trace anonymization. ACM CCR **36**(1) (2006)
48. Pang, R., Allman, M., Bennett, M., Lee, J., Paxson, V., Tierney, B.: A first look at modern enterprise traffic. In: ACM/USENIX IMC (2005)
49. Winpcap homepage, http://www.winpcap.org/
50. Symantec Security Response, http://securityresponse.symantec.com/avcenter
51. Shannon, C., Moore, D.: The spread of the Witty worm. IEEE Sec. Priv. **2**(4), 46–50 (2004)
52. Ringberg, H., Rexford, J., Soule, A., Diot, C.: Sensitivity of PCA for traffic anomaly detection. In: ACM SIGMETRICS (2007)
53. Ashfaq, A.B., Joseph, M.J., Mumtaz, A., Ali, M.Q., Sajjad, A. and Khayam, S.A.: A comparative evaluation of anomaly detectors under portscan attacks. In: Recent Advances in Intrusion Detection (RAID) (2008)
54. Filiol, E., Josse, S.: A statistical model for undecidable viral detection. J. Comput. Virol. **3**, 65–74 (2007)
55. Filiol, E., Josse, S.: Malicious cryptography. In: CanSecWest (2008)
56. van Trees, H.L.: Detection, Estimation and Modulation Theory: Part I, 1st edn. Wiley-Interscience, New Yok (2001)