Progressive Differential Thresholding for Network Anomaly Detection

Sardar Ali, Hassan Khan, Muhammad Ahmad, and Syed Ali Khayam
School of Electrical Engineering and Computer Science (SEECS), National University of Sciences and Technology (NUST), Islamabad, Pakistan
Email: {sardar.ali, hassan.khan, muhammad.ahmad, ali.khayam}@seeecs.nust.edu.pk

Abstract—In this paper, we propose a Progressive Differential Thresholding (PDT) framework for coordinated network anomaly detection. Under the proposed framework, nodes present on a packet’s path progressively encode their opinion (malicious or benign) inside a packet. Subsequent nodes on the path use the encoded opinion as side-information to adapt their anomaly detection thresholds and in turn improve their classification accuracies. Accuracy benefits of PDT are evaluated through experimental evaluations of multiple non-proprietary anomaly detectors on a publicly-available attack dataset. These evaluations indicate that, while being distributed and having negligible complexity and communication overheads, the proposed PDT framework provides considerable and consistent improvements in anomaly detection accuracy. We observe upto 54% improvements in ADS detection accuracy while upto 4 times reduction in the false alarm rates.

I. INTRODUCTION

To cater for accuracy and scalability shortcomings of standalone traffic anomaly detectors, coordinated anomaly detection frameworks are now being pursued. Thus far, research in this domain has been focusing on developing coordinated frameworks that can be implemented by spatially distributing ADSs in the network with a central server performing attack correlation [2],[3], [4]. The centralized nature of these approaches introduces detection delays, communication overheads, and computational complexity on the central server. Consequently, these approaches cannot scale to largescale deployments.

In view of the scalability issues of spatially distributed ADSs, we advocate a different perspective using a Progressive Node Collaboration (PNC) model. Under the PNC model, nodes present on a packet’s path progressively encode their opinion (malicious or benign) inside a packet. This opinion serves as an effective side-information that can be used by subsequent nodes to improve anomaly detection accuracy. PNC offers several advantages over the centralized model: 1) PNC is completely decentralized; 2) It can be implemented with negligible complexity and communication overhead; and 3) It does not require all nodes to support a fixed set of anomaly detection algorithms and communication protocols, thereby allowing interoperability among diverse hardware and software modules.

In this paper, we propose a method to improve ADS detection accuracy using a PNC-based Progressive Differential Thresholding (PDT) framework. The operation of contemporary ADSs can be conceptually subdivided into two sequential function: traffic processing and thresholding. In the traffic processing function, high dimensional traffic data is projected into a lower dimensional space where some mathematical functions or rules are applied to compute an anomaly score for the input traffic. In the second step, a threshold function is applied to the anomaly score and the traffic is classified as malicious if the score falls above or below a predefined threshold. Under the PDT framework, ADS at each node uses the packet scores from the previous PNC nodes to adapt its anomaly classification threshold. Thus, relaxed thresholds are used for packets which are marked benign by previous nodes, while strict thresholds are applied on packets marked as malicious. This technique promises higher accuracies and scalable detection of network anomalies, with negligible communication or processing overheads.

We provide experimental evidence on a labeled and publicly-available dataset of attack traffic at three different points of deployment. The attack traffic in our dataset includes attacks of different types and varying rates. Performance evaluations using NETAD [5] and Maximum Entropy [6] show that, while having negligible computational complexity and communication overhead, the proposed approach provides significant (upto 54%) increase in the detection rate and a substantial upto 4 times decrease in the false alarm rate.

II. RELATED WORK

Several distributed solutions have been proposed to deal with DDoS attacks which correlate traffic from different deployment points in the network by sending traffic to a centralized server [2],[3], [4]. However, accompanying detection delays, communication overheads, and computational complexity makes these approaches less attractive for large scale deployments. Therefore, in this section, we only discuss relevant differential thresholding and decentralized approaches for node collaboration.

Agosta et al. [7] proposed an anomaly detector which adjusts its threshold according to the variations observed in input. In a recent paper [8], we proposed a Markovian stochastic model of temporal dependence in IDS anomaly scores where conditional entropy analysis is used to determine the order
of the Markov chain that should be used for threshold estimation. Unlike these time-based threshold adaptation approaches, we propose a spatially-adaptive threshold adaptation method which relies on opinions of previous nodes on a packet’s path.

In order to mitigate the DDoS attacks, some prior studies have proposed to spatially distribute ADSs in a network [9], [10]. These approaches require an additional channel for communication between different nodes. In [1], we proposed a PNC model in which ADSs are deployed progressively on nodes on a packet’s path. The proposed Progressive Security-Aware Sampling (PSAS) approach allowed ADSs to encode packet scores inside packet header. The encoded score was used by ADSs at subsequent hops to sample the marked as malicious packets with high probabilities.

### III. Evaluation Dataset

In order to evaluate the proposed PDT framework, we needed attack traffic at different deployment points in a network. Furthermore, for comprehensive evaluation, we needed attacks of different types (DoS, portscan, etc.) and different rates for each attack type. While some old attack datasets are available [11]-[13], they neither contain traffic at different deployment points nor attacks of different types (DoS, portscan, etc.) and different rates for each attack type. Therefore, we collect our own network traffic dataset and make it publicly available for repeatable performance evaluations. The rest of this section describes our data collection experiment.

We collected network traces at three progressive points of deployment in our campus network. The research labs in our campus are located in the three research wings. Traffic from all the research lab router (first hop) is relayed to a research wing router (second hop) using fiber connections. The research wing router is connected to a distribution router that handles traffic of the entire university. Due to privacy constraints, we were not allowed to log traffic at the distribution router. The research lab contains 28 computers, while the research wing router routed traffic for more than 50 computers. The network topology is shown in Figure 1. The details of attack and background traffic are provided in following subsections.

#### Attack Traffic

For attack traffic, we launch two attack types (TCP SYN, ICMP Protocol Unreachable, and ICMP Path MTU Discovery) and DoS (TCP SYN, fraggle UDP flood, and ICMP echo ping flood) simultaneously from three end hosts in our research lab. The DoS attacks were launched on two servers under our administration with public IP addresses. Each attack was launched for a period of five minutes with spoofed IP address. For every attack type, two high-rate (100, 1000 packets/sec) and three low-rate (0.1, 1, 10 packets/sec) instances were launched.

#### Background Traffic

We captured the normal traffic in six periods, each one of over three hours. During traffic capturing, different applications were hosted on the machines including file transfer, web browsing, instant messaging, real-time video streaming, peer-to-peer, etc.

#### Discussion

Table I shows the diversity of the collected attack dataset. At the endpoints, the background traffic rate during low rate attacks orders of magnitude greater than the attack rate. On the other hand, background traffic rate is comparable to other rates.
deployments are as follows: key features of PDT that make it more attractive in largescale serious deployment constraints of centralized approaches. The A. PDT Advantages threshold is used, whereas a strict threshold is used for packets marked as malicious by previous hop. For packets marked benign by the previous hop, a relaxed threshold is used, whereas a strict threshold is used for packets marked as benign by previous hop. For the proof-of-concept experiments in this paper, we use two thresholds for classification of packets. The ADS operating at the next hop uses this score as side information for anomaly detection. This binary side information can be effortlessly encoded inside IP packets, thus allowing different nodes to collaborate without any additional communication overhead. For the experiments in this paper, we use the IP reserved flag to communicate the binary score. We now propose and evaluate a Progressive Differential Thresholding (PDT) algorithm under the PNC model. The first node on a packet’s path uses a fixed threshold value since it has no prior information to perform differential thresholding. The ADS operating at the next hop uses the score embedded by first hop as side information and applies different thresholds to different packets. For the proof-of-concept experiments in this paper, we use two thresholds for classification of packets. For packets marked benign by the previous hop, a relaxed threshold is used, whereas a strict threshold is used for packets marked as malicious by previous hop.

A. PDT Advantages

PDT’s apparently simple methodology overcomes some serious deployment constraints of centralized approaches. The key features of PDT that make it more attractive in largescale deployments are as follows:

- Anomaly detection accuracy generally degrades as we move from the endpoints to the network core [14]. Since PDT relies on opinions of previous hop nodes, it can be inferred intuitively (and will be shown empirically later) that PDT’s detection accuracy should improves at each progressive node with the accuracy improvements becoming pronounced as the packet progresses along its path.
- PDT is generic and can be used with any ADS. In fact, since PDT allows different ADSs to be deployed at each hop, each of these ADSs can be customized for the traffic characteristics and attack vulnerabilities observed at a given point of network deployment.
- PDT is completely decentralized and does not introduce additional communication overhead because progressive nodes communicate using only a single bit which can be encoded in unused IP packet headers. Such low overhead encoding precludes the need for additional communication channels and protocols between nodes.
- PDT has very low processing complexity; empirical results substantiate this claim in the following section.

B. PDT Deployment in Our Experimental Setup

We now propose and evaluate a Progressive Differential Thresholding (PDT) algorithm under the PNC model. The first node on a packet’s path uses a fixed threshold value since it has no prior information to perform differential thresholding. The ADS operating at the next hop uses the score embedded by first hop as side information and applies different thresholds to different packets. For the proof-of-concept experiments in this paper, we use two thresholds for classification of packets. For packets marked benign by the previous hop, a relaxed threshold is used, whereas a strict threshold is used for packets marked as malicious by previous hop.

Fig. 2 shows a pictorial view of PDT deployment in our experimental setup. Our experimental setup of three cascaded nodes: endpoint, lab router, and research wing router. The first ADS, deployed at an endpoint, does not have any a priori information about the packets’ maliciousness, and therefore it uses a fixed classification threshold for all packets. The endpoint ADS embeds a binary anomaly score in the packet, so that packets classified as malicious are marked using unused IP header bits. The second node along the packet’s path (first hop/lab router) has additional information in the form of the binary packet score from the last node. Using this side information, the second node uses a strict threshold for packets marked as malicious and a relaxed threshold for the packets marked as benign by the previous hop. Using differential thresholding, the ADS at the second node marks packets detected as malicious and forwards them on the the packet’s path. Similarly, the third node (second hop/research wing router) along the packet path follows the same procedure of using differential thresholding and packet marking.

We now evaluate the performance of proposed PDT approach on the attack dataset collected for this study.

V. PERFORMANCE EVALUATION

In this section, we discuss performance evaluation of the proposed PDT approach on four performance metrics: detection rate, false alarm rate, communication overhead, and run-time computational complexity. Before proceeding with the performance evaluation, we briefly describe the anomaly detectors used for performance evaluation in this work.

A. Anomaly Detectors

While we evaluate the proposed PDT approach on Maximum Entropy [6], NETAD [5], TRW-CB [15] and Rate

Fig. 2. Deployment of the PDT framework in our experimental setup.
In this paper, we only report the results for Maximum Entropy [6] and NETAD [5]. Due to space constraints, we skip the qualitatively similar results of TRW-CB [15] and Rate Limiting [16] approaches. These anomaly detectors were chosen to ensure diversity because these detectors have very different detection principles and features, and operate at different traffic granularities. On the one hand, we employ an information-theoretic self-learning system like Maximum Entropy [6] and on the other hand we use NETAD [5] which is a packet-based rule-modeling system. We now provide a brief description of these ADSs. Interested readers are referred to the original papers [6], [5] for a detailed description of these ADSs.

The Maximum Entropy based detector [6] estimates the benign traffic’s baseline distribution using Maximum Entropy estimation. This ADS divides the traffic into 2,348 packet classes. These packet classes are defined based on destination ports and the transport protocols. The Kullback-Leibler (K-L) divergence measure is then used to flag anomalies if divergence from the baseline distribution exceeds a threshold from the baseline distribution. NETAD [5] operates on rule-based filtered traffic in a modeled subset of common protocols. It computes a packet score depending on the time and frequency of each byte of packet, and unusual header values are assigned high scores. A threshold is applied on a packet’s score to find anomalous packets. For performance evaluations, parameter tuning for these anomaly detectors is performed in the same fashion as in a recent evaluation study [14].

### B. Accuracy Evaluation

We performed all our analysis by using a range of differential thresholds and plotting their corresponding Receiver Operating Curves (ROCs). For brevity, we only report accuracy results at the best ROC points. The accuracy results on the first hop, for all attacks, are the same for both standalone and PDT-enabled ADSs because the ADS at the first node has no prior information about the maliciousness of the traffic.

Experimental results in Fig. 3 show that the detection rate of the PDT-based Maximum Entropy ADS significantly increases for all attack types. For portscan attack, PDT resulted in 32% and 23% accuracy improvements on the second and the third nodes, respectively. Similarly, for the TCP flood attack, PDT achieved 40% and 22% improvements in accuracy at the second and third nodes, respectively. The improvements for the UDP flood attack are even better with PDT achieving 45% accuracy improvements at the second and 54% at the third node. In addition to increased detection rate, PDT also results in decrease in false alarms as shown in Fig. 4. For the portscan attack, we observe that the false alarm rates of PDT-based ADSs are 3 times and 4 times less than that of standalone ADSs for the second and the third nodes, respectively. Similarly, for TCP and UDP flood attacks 2 times
tool. The complexity of PDT is
computed using the
additional communication overhead.

C. Communication Overhead and Computational Complexity

The proposed PDT technique embeds the packet anomaly score in the packet header. No extra field or new packet is needed to communicate the packet score of a packet to the next hop node. Therefore, the proposed PDT approach has no additional communication overhead.

Table II provides the computational complexity comparison of PDT-based and standalone ADS. These numbers are computed using the hprof tool. The complexity of PDT is almost equal to the standalone ADS. This is because PDT-based ADS only checks if a packet is marked and embeds its binary score in that packet. Similarly, the extra condition of checking whether the packet is marked or not and then deciding which threshold to use adds to its complexity. On the other hand, the small/strict value of PDT threshold results in quick threshold comparisons as compared to standalone ADS.

VI. LIMITATIONS AND COUNTERMEASURES

We now highlight some limitations of the proposed PDT technique and propose solutions to circumvent these limitations.

- We do not assume that all ADSs are deployed by the same operator, and we do indeed consider that a node in the PDT scheme may be untrusted (e.g., marking packets intentionally). If a node is not trusted it can exploit the design of PDT by intentionally marking a significant fraction of the traffic as malicious/benign.
- PDT is designed specifically for real-time anomaly detectors. Other anomaly detectors which operate on non-real-time measurements (e.g., file analysis algorithms, etc.) cannot use differential thresholding.

REFERENCES