Network Anomaly Detection Using Autonomous System Flow Aggregates

To appear, GLOBECOM 2014. Draft version

Thienne Johnson and Loukas Lazos
Department of Electrical and Computer Engineering
University of Arizona, Tucson, Arizona, 85721
{thienne, llazos}@email.arizona.edu

***Abstract*—Detecting malicious traffic streams in modern computer networks is a challenging task due to the growing traffic volume that must be analyzed. Traditional anomaly detection systems based on packet inspection face a scalability problem in terms of computational and storage capacity. One solution to this scalability problem is to analyze traffic based on IP flow aggregates. However, IP aggregates can still result in prohibitively large datasets for networks with heavy traffic loads. In this paper, *we investigate whether anomaly detection is still possible when traffic is aggregated at a coarser scale*. We propose a volumetric analysis methodology that aggregates traffic at the Autonomous System (AS) level. We show that our methodology reduces the number of flows to be analyzed by several orders of magnitude compared with IP flow level analysis, while still detecting traffic anomalies.**

I. Introduction

The severity and volume of network attacks launched against the network infrastructure have soared in recent years [2]. These attacks rapidly evolve to avoid detection from signature-based Intrusion Detection Systems (IDSs) that require a priori knowledge of the attacks' anomaly patterns (signatures) [4]. One solution to detecting unknown attacks, also referred to as *zero-day attacks*, is to employ network-based IDSs (NIDSs) [9]. The latter rely on statistical traffic analysis to timely detect abnormal network activities. Typical activity metrics include the traffic volume, the number of IP flows, counts for different packet classes, and packet size distributions, to name a few. The rapid growth of network traffic in modern networks poses several challenging problems for NIDSs [1]. Most notably, modern NIDSs must cope with vast storage and computational overheads to accurately maintain the network state and allow for timely anomaly detection.

Flow sampling and IP flow aggregation have been the primary methods for addressing the resource scalability problem [19]. Flow aggregation techniques reduce the amount of state and history information that is maintained by summarizing IP flows to statistical metrics, merging multiple flow records with similar properties, and discarding benign flows (using whitelists) [5]. Even when aggregating traffic at the IP flow level, the computation and storage requirements for an online NIDS can be prohibitively large. Moreover, per-IP flow statistical analysis can lead to high false positive rates [19]. To reduce the communication and storage overheads, we exploit the organization of the IP space to Autonomous Systems (ASes). An AS is an internetwork under a single administrative and often business authority that dictates a unified routing policy. Each AS represents a set of IP prefixes that are advertised to other ASes using the Border Gateway Protocol (BGP) [15]. At present, the entire IP space of approximately 4.2 billion addresses (not counting IPv6 addresses) is under the administrative control of about $40,000$ ASes [7]. Therefore, aggregating traffic at the AS level can drastically reduce the required state and history information storage and computational analysis.

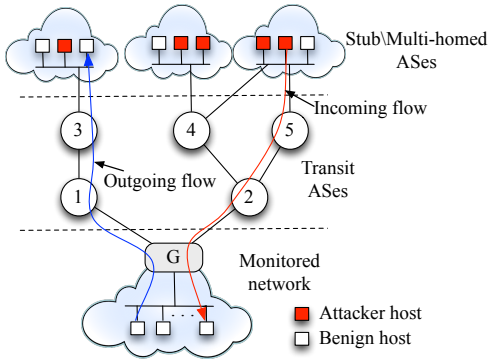


Fig. 1. The NIDS detects network anomalies by intercepting, aggregating, and analyzing IP traffic exchanged between the monitored network and the rest of the Internet at the gateway (G).

In this paper, *we investigate network anomaly detection techniques based on statistical traffic analysis of AS level traffic*. We hypothesize that aggregation at the AS level is capable of detecting large-scale network threats such as Distributed Denial-of-Service (DDoS) attacks that create substantial deviations in network activity compared with benign network conditions. At the same time, our method suppresses false alarms that are the result of large statistical variations of individual IP flows, as it operates at a much coarser scale. The false alarm suppression comes at the expense of missing the detection of anomalies which cause moderate to low network disturbances. Our NIDS is meant to operate at the border gateway servicing traffic between a monitored network and the rest of the Internet (see Fig. 1). The major contributions of this paper are as follows:

- We design an NIDS based on AS flow aggregates. By aggregating flows at the AS level, we achieve a significant reduction in storage and computation overhead compared to IP-flow based NIDSs. To our knowledge, this is the first work that uses AS aggregates for anomaly detection.

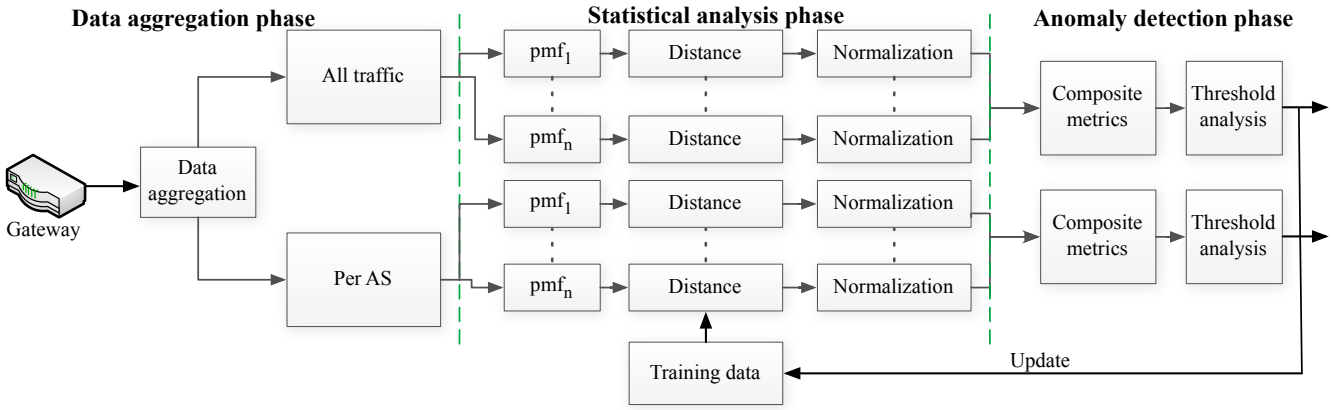


Fig. 2. Overview of the anomaly detection process.

- We adapt basic network anomaly detection metrics (packet count, traffic volume, flow count, etc.) to the AS domain and propose composite metrics of network activity that combine several basic metrics.
- We propose a new basic metric that counts the number of AS flows (packet flows between the monitored system and other ASes) for detecting anomalous events and show that this metric improves the overall detection capability.
- As a case study, we apply our methodology on a real dataset that contains annotated attacks. These attacks establish the ground truth.

The rest of the paper is organized as follows. In Section II, we present the NIDS design. In Section III, we apply our methodology on a dataset containing annotated attacks. Section IV presents related work. We conclude and present future work in Section V.

II. NIDS Design

The proposed NIDS is designed as the first line of defense between the monitored network and the rest of the Internet. Fig. 1 presents the application scenario under consideration. The NIDS operates at the border gateway (G), which connects the monitored network to one or several transit ASes (ISPs). The hosts of the monitored network exchange traffic with hosts that reside on other ASes through the gateway. Several hosts may be malicious and launch various types of attacks against the monitored network, either independently or in a coordinated fashion. The NIDS is responsible for identifying malicious traffic streams by analyzing IP traffic at the AS level.

The NIDS performs two types of analyses to identify malicious traffic streams: (a) volumetric analysis on the entire traffic stream incoming to and outgoing from the gateway and (b) volumetric analysis on traffic aggregated per AS. The analysis methodology is divided into three phases: the data aggregation phase, the statistical analysis phase, and the anomaly detection phase. In the data aggregation phase, traffic at at the gateway is summarized at the AS level to create AS flow metrics. In the statistical analysis phase, AS flow metrics are statistically compared to historic values obtained from training data to derive basic metrics of statistical deviation from the norm. Finally, in the anomaly detection phase, basic metrics are combined to composite metrics and are evaluated using threshold analysis. The phase sequence for the proposed NIDS is shown in Fig. 2.

A. Definitions and Metrics

We divide time into *intervals, epochs*, and *aggregation periods* defined as follows.

Definition 1 (Aggregation period A): The time period over which a metric of interest is aggregated. Aggregation over A provides one sample value for the computed metric.

Definition 2 (Epoch E): The time period over which k samples are collected. One epoch consists of k aggregation periods. The k samples create a statistical model for the metric of interest during the online phase.

Definition 3 (Interval I): The time period over which a single statistical model is assigned during the training phase. An interval consists of ℓ epochs.

The relationship between intervals, epochs and aggregation periods is shown in Fig. 3.

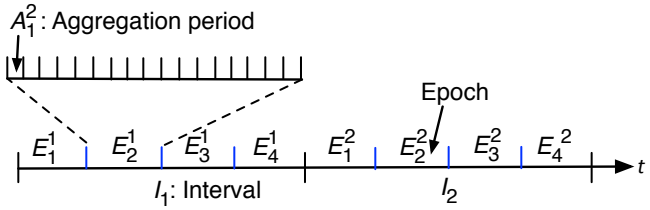


Fig. 3. Time is divided to intervals, epochs, and aggregation periods.

Definition 4 (IP flow): All IP packets containing the four-tuple <source IP, source port, destination IP, destination port> over an aggregation period are assigned to a unique IP flow.

Definition 5 (AS flow): All IP packets mapped to the four-tuple <source ASN, source port, destination ASN, destination port> over an aggregation period are assigned to a unique AS flow. Here, ASN is the unique AS number assigned to each AS by the Internet Assigned Numbers Authority (IANA). Every ASN is assigned a collection of IP prefixes. The IP to ASN mapping is done using the IP prefixes assigned to each AS.

B. Data Aggregation Phase

During the data aggregation phase, the NIDS intercepts the packets arriving at the gateway. The packet header of each packet is matched to a set of pre-defined rules that associate the packet with an AS flow. To cope with the high forwarding rates of modern gateways, the packet matching process is implemented in hardware. The matching rules are stored at ternary content accessible memory (TCAM), which allows for parallel rule processing and significantly reduces the lookup delay [11]. High-end routers support up to several tens of thousands of TCAM entries [18], which is in the same order as the total number of ASes [7].

We perform the packet header matching using an *IP prefix-to-ASN map*. This map associates every ASN with a collection of IP prefixes. Matching of an IP address (source or destination) with an IP prefix associates that IP address with an ASN. The IP prefix-to-ASN map is constructed from border gateway control (BGP) advertisements that are exchanged between BGP gateways for routing purposes. For our experimentations, we obtained the IP prefix-to-ASN map from the Cooperative Association for Internet Data Analysis (CAIDA) that maintains a continuously updated database based on a measurement infrastructure deployed worldwide [3]. Fig. 4 details the process of associating IP packets with AS flows. The IP prefix-to-ASN database populates the IP prefix matching rules in the TCAM. Packets intercepted at the gateway are associated with AS flows based on the matching rules.

We note that it is possible to receive packets whose IP address does not match any of the IP prefix rules installed on the TCAM. This is because some IP prefixes, although assigned to ASes, are not advertised by any AS. Moreover, the CAIDA measurement infrastructure does not receive all BGP advertisements exchanged by BGP routers. To address the case of unmapped IP addresses, we assign unknown IP prefixes to virtual ASNs. This assignment is done in /24 prefix blocks. Using the IP packet to AS flow association process, we compute the following basic metrics:

Definition 6 (Packet count $N(k, A_i^j)$): The number of packets associated with the k^{th} AS flow, over aggregation period A_i^j.

Definition 7 (Traffic volume $V(k, A_i^j)$): The traffic volume (in bytes) associated with the k^{th} AS flow, over A_i^j.

Definition 8 (IP Flow count $IP(k, A_i^j)$): The number of IP flows associated with the k^{th} AS flow over A_i^j.

Definition 9 (AS Flow count $F(A_i^j)$): The number of AS flows that are active during the aggregation period A_i^j.

The AS flow aggregate metrics are used for the statistical analysis of the traffic intercepted at the gateway, as described in the following section.

C. Statistical Analysis Phase

To detect malicious traffic streams, we perform statistical analysis. We emphasize that our goal is not to optimize the statistical methodology for detecting anomalies. Extensive research has already been performed on this domain [4], [20].

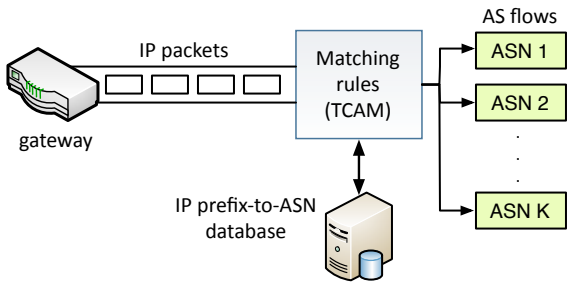


Fig. 4. Association of IP traffic with AS flows.

We aim at demonstrating that near real-time anomaly detection is possible by aggregating network traffic at the AS level. For this purpose, we adopt a semi-supervised statistical anomaly detection technique. Our technique operates in two phases; a training phase and an online phase.

During the training phase, we create stochastic models of normal network activity in the form of probability distributions. We use training datasets for this purpose. During the online phase, we perform the steps depicted in Fig. 2. We first create empirical probability distributions of the traffic intercepted at the gateway using count-based histograms. We then compare the online distributions with those obtained during the training phase and compute the "distance" between the respective distributions. Further, we normalize the distance to suppress the metric dynamic range between zero and one. Finally, we continuously update the training data using moving window techniques. We now describe each phase in detail.

Training Phase: In the training phase, we divide time to intervals $I_1, \ldots, I_m$. Traffic for each of the m intervals is represented by the same model. The idea here is that traffic is expected to follow different distributions at different periods (e.g. peak vs. non-peak). As an example, dividing each week day to six four-hour intervals yields a total of 42 models, one for every four-hour period of the week. For an interval I_j, we create empirical probability distributions $\mathcal{Q}_j(M)$, for each metric $M = \{N, V, IP, F\}$ defined in Section II-B. Distributions $\mathcal{Q}_j^M$ are computed using empirical histograms from the training dataset. Specifically, we divide each interval to $\ell * k$ aggregation periods. For an aggregation period, we compute each metric in M and obtain one sample value for $\mathcal{Q}_j(M)$. We collect $\ell * k$ samples for W occurrences of interval I_j, i.e., $\{I_j(t), I_j(t+1), \ldots, I_j(t+W)\}$, where W is a pre-defined time window. The $W * \ell * k$ samples are organized into an empirical probability histogram indicating the relative frequency for the metric. We apply a simple binning method by placing samples to bins of equal width. More elaborate density estimation methods can be explored [16]. An empirical probability histogram for the packet count metric N is shown in Fig. 5. The histogram creates a probability mass function (pmf) model for the traffic statistics over an interval I_j.

Online Phase: In the online phase, we model the metrics in M by also computing empirical pmfs. Because we are interested in near real-time anomaly detection, the pmf for the online phase is computed over an epoch, which is shorter

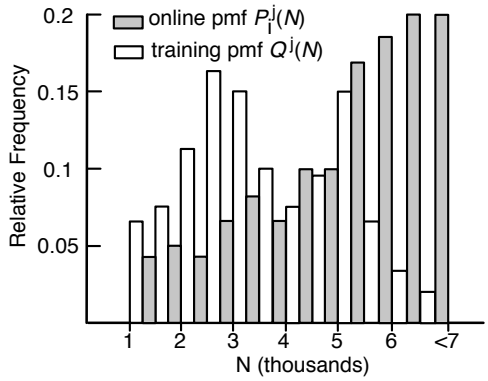


Fig. 5. An empirical probability histogram for the packet count metric N and for a single interval.

than an interval. Specifically, we collect k samples for each metric using the aggregate values over k aggregation periods (recall that each epoch is divided to k aggregation periods). For an epoch E_i^j, we create empirical probability distributions $\mathcal{P}_i^j(M)$, for every $M = \{N, V, IP, F\}$. These are compared with the distributions $\mathcal{Q}_j(M)$ that correspond to the interval that includes epoch E_i^j.

Statistical Divergence: We use statistical divergence to measure the deviation of the online phase model from the training phase model. Specifically, for an online pmf $\mathcal{P}_i^j(M)$ and a training pmf $\mathcal{Q}_j(M)$ corresponding to the same time period, we quantify the statistical divergence between $\mathcal{P}_i^j(M)$ and $\mathcal{Q}_j(M)$ using the *Jeffrey distance*, defined as follows.

Definition 10 (Jeffrey distance $\Lambda(\mathcal{P}, \mathcal{Q})$): For two discrete pmfs $\mathcal{P} = \{p_1, p_2, \ldots, p_k\}$ and $\mathcal{Q} = \{q_1, q_2, \ldots, q_k\}$ with a support set $\mathcal{S} = \{s_1, s_2, \ldots s_k\}$ (note that for our purposes, each s_i represents a bin), the Jeffrey distance Λ is,

$$\Lambda(\mathcal{P}, \mathcal{Q}) = \frac{1}{2}\left(KL(\mathcal{P}, \mathcal{Q}) + KL(\mathcal{Q}, \mathcal{P})\right), \tag{1}$$

where $KL(\mathcal{P}, \mathcal{Q})$ is the Kullback-Liebler divergence,

$$KL(\mathcal{P}, \mathcal{Q}) = \sum_{i=1}^{k} p_i \times log(\frac{p_i}{q_i}). \tag{2}$$

We use the Jeffrey distance because the KL divergence is not a true distance metric[1]. Distances are normalized to ensure equal distance scales when multiple metrics are combined to one. We define a normalization function J that produces the normalized metric $J(\mathcal{P}_i^j(M), \mathcal{Q}_j(M))$ as follows,

$$J(\mathcal{P}_i^j(M), \mathcal{Q}_j(M)) = \frac{\Lambda(\mathcal{P}_i^j(M), \mathcal{Q}_j(M))}{\Lambda(\mathcal{P}_i^j(M), \mathcal{Q}_j(M))_{95th}}, \tag{3}$$

where $\Lambda(\mathcal{P}_i^j(M), \mathcal{Q}_j(M))_{95th}$ is the value that falls on the 95^{th} percentile of the historical distance for a specific metric and AS node, accumulated over the moving window W. Values of $J(\mathcal{P}_i^j(M), \mathcal{Q}_j(M))$ higher than one are normalized to 1.

[1]For the KL distance, the divergence between two pmfs is generally not symmetric (i.e. $KL(p, q) \neq KL(q, p)$) and the triangle inequality is not satisfied. Jeffrey distance corrects the KL asymmetry [21].

One limitation of employing statistical divergence for detecting network abnormalities is that a scalar metric alone is not sufficient to indicate the nature of the divergence. As an example consider, the training pmf shown in Fig. 5 (white bars), for metric N. Assume that for an epoch E_i^j traffic was very low, causing the majority of the probability mass for the online pmf to be concentrated at low values ($N = 1$K and $N = 2$K). In this case, the Jeffrey distance between the training and online distributions will be high, due to the distribution dissimilarity. This will raise a false alarm for our system, despite the low traffic activity at the gateway. To suppress this type of false alarms, we set $J(\mathcal{P}_i^j(M), \mathcal{Q}_j(M)) = 0$ if the average aggregate metric value for a given epoch is lower than the average aggregate metric value for the training data.

D. Anomaly Detection Phase

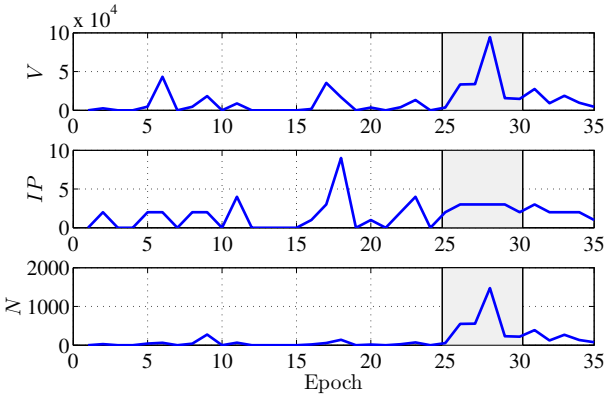
Network traffic analysis at the IP level has shown that network anomalies cause fluctuations in different metrics combinations [14], [19], [21], [22]. For example, (D)DoS attacks typically demonstrate a drastic increase in metrics N, V, and IP [19]. To capture the multi-dimensional nature of network behaviors, we create composite metrics by combining several basic metrics in M. In general, a composite metric C_i is expressed by applying a function $G_i(J(N), J(V), J(IP), J(F))$ on the normalized Jeffrey distances computed in the statistical analysis phase. For instance, function G could be a simple weighting formula among the different metrics. The weights could be adjusted to favor a subset of metrics, depending on the nature of the attack to be detected.

At the end of each epoch, composite metrics C_i are compared to threshold values θ_i. If any metric exceeds the predefined threshold, an alert is triggered to indicate abnormal behavior for the specific epoch. The threshold mechanism is also used in the training data update process to maintain a recent view of normal traffic. This is done as follows.

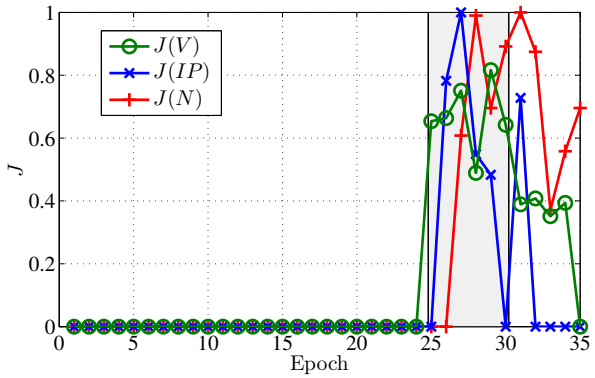
Training data update: We employ a moving window mechanism for maintaining the training data. Samples collected over the last W intervals $\{I_j(t), I_j(t+1), \ldots, I_j(t+W)\}$ are used to compute the empirical pmf for interval I_j. At the end of interval $I_j(t + W + 1)$ of the online phase, samples from epochs $E_i^j \in I_j(t + W + 1)$ for which all composite metrics C_i were below the corresponding thresholds θ_i, are admitted to the training data set. At the same time, samples from $I_j(t)$ are deleted from the training set. If none of the epochs can be admitted to the training set, the samples from $I_j(t)$ are maintained. With the update of the training set, the pmts for the corresponding metrics are also updated. Note that all operations are performed per AS node.

III. Experiments

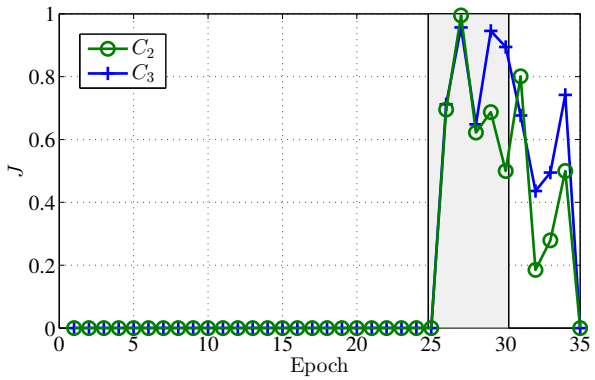
In this section, we evaluate the proposed NIDS using a popular annotated attack dataset [12]. The annotated attacks serve as ground truth for the feasibility of AS-based detection.



(a) Raw metrics.



(b) Basic distance metrics.



(c) Composite metrics.

Fig. 6. AS 1136, 6am-9am - TCP reset.

A. Dataset

We used the MIT LLS DDOS 1.0 intrusion dataset [12], which simulates several DoS attacks and background traffic. We chose this dataset because of the availability of both training and online data and because it has been analyzed by many prior works using IP flow based analysis [6], [14], [17], [22]. The dataset provides three weeks of training data and two weeks of online data. In our evaluation, we selected the traffic traces from Tuesday of the first three weeks as training data. This day was selected because numerous attacks occurred during Tuesday of week 5. Table I summarizes the three composite metrics used in our analysis, which were determined based on the findings in [10]. In our implementation, we used the CAIDA AS-to-IP prefix database [3] for the IP-to-AS matching process.

TABLE I
TRAFFIC ANOMALY PATTERNS

M	Definition	Attack Type
C_1	$0.3J(N)+0.3J(IP)$ $+0.35J(F)$	(D)DoS high packet rate # of IP/AS connections (all traffic)
C_2	$0.5J(N)+0.5J(IP)$	(D)DoS high packet rate # of IP connections (per AS)
C_3	$0.5J(N)+0.5J(V)$	(D)DoS high volume packet rate (per AS)

B. Results

In all experiments, weeks were divided to 42 intervals (6 four-hour intervals per day). For the online phase, we set each epoch to 5 minutes. The aggregation period for both the online and training phases was set to 6 seconds (50 samples per epoch, 2,400 samples per interval). According to [13], the attacks that occurred on Tuesday of week 5 are as follows:

- Attack 1: (*TCP reset*): a DoS attack that disrupts TCP connections by sending TCP RST packets to the victim machine.
- Attack 2: (*Teardop*): a DoS attack that exploits a flaw in the fragmentation/reassembly process of older TCP/IP stacks.
- Attack 3: (*Casesen*): a user-to-root attack that installs and executes three files on a vulnerable host.
- Attack 4: (*Selfping*): a DoS attack that remotely reboots the targeted host with a single ping command.

Although attacks 1-4 are not relevant in today's operating systems, from a network perspective, they exhibit similar characteristics with modern (D)DoS attacks such as increased traffic volume, packet count, and number of IP flows.

1) Per-AS Analysis: In this set of experiments, we aggregated traffic at the AS level and computed composite metrics C_1-C_3. Our analysis was performed over the 6AM-9AM time period and spanned two intervals; the 4AM-8AM interval and the 8AM-12PM interval. As a result, the training pmf model changes at epoch 24. Based on our analysis, we discovered AS 1136 to be the source of the *TCP reset* attack that occurs during epochs [26,30]. Fig. 6(a) shows the raw metrics for N, V, and IP (epochs [25,30] are gray shaded to highlight the attack period). We observe that the TCP reset attack causes a spike in V and N, but not in IP. This is because the attack was instrumented from a relatively small set of hosts.

Fig. 6(b) shows the normalized basic metrics for the same time period. Before epoch 26, all metrics are equal to zero because the online traffic (N, V, and IP) was lower on average than then one recorded in the training dataset. Note that $J(IP) = 0$ for epoch 18, despite the raw value of 9 IP flows per epoch. This is because the expected number of IP flows for the 4AM-8AM interval was higher than 9. On the other hand, when switching to the pmf model for the 8AM-12PM interval, a raw IP flow value of four yields a $J(V)$ larger than zero. Fig. 6(c) shows composite metrics C_2 and C_3 for the same time period (C_1 is not plotted, as it is used only for

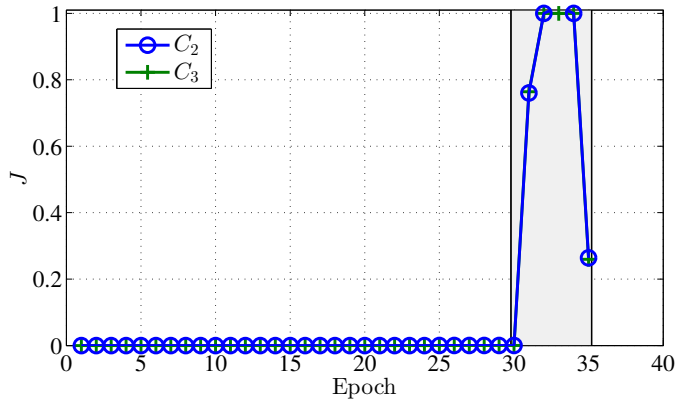


Fig. 7. AS 5511 - composite metrics for 6am-9am - Teardrop.

the collective analysis of all traffic). We observe that both C_2 and C_3 attain similarly high values during the attack.

During interval [30,33], an attack launched by AS 5511 is captured by the NIDS. This corresponds to the teardrop attack (attack 2) according to the annotated dataset [13]. Figure 7 shows C_2 and C_3 for AS 5511. Both metrics have almost identical values (within 1% to 2%) during the attack period and detect the sharp increase in the number of packets and traffic volume. Finally, AS 1136 originates attack 4 during epochs 8 and 9 between 9AM-12PM. Figure 8 shows metrics C_2 and C_3 between 9AM-12PM for AS 1136. We observe that C_3 indicates abnormal network behavior, while C_2 has a metric value below 0.5. The selfping attack caused a large increase in the number of IP flows, without significantly increasing the traffic volume received from AS 1136. This is because the selfping attack involves the flooding ping packets that are short in size.

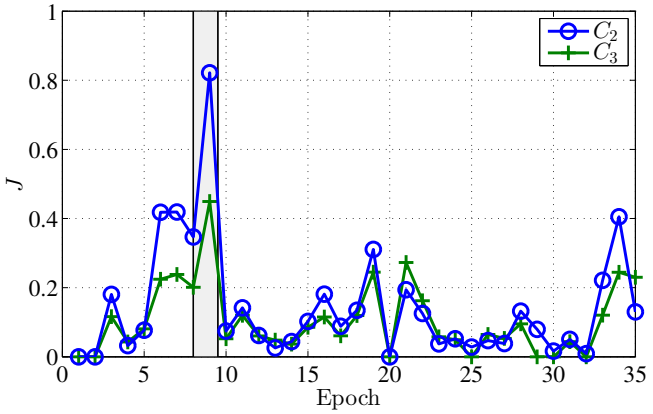


Fig. 8. AS 1136 - Composite distance metrics, 9am-12pm - Selfping.

As a final note, we measured a significant reduction in the state information maintained by our NIDS. The number of AS flows corresponded to approximately 31.5% of the number of IP flows and the number of AS nodes corresponded to approximately 22% of IP nodes.

2) All traffic analysis: In this set of experiments, we computed the composite metrics over the entire traffic stream, as opposed to aggregating per AS. Figs. 9(a) and 9(b) show the basic and composite metrics computed over the 6AM-9AM period, with annotated attacks 1,2, and 3. For attack 1, only the number of AS flows has a considerable distance value. The rest of the metrics indicate a normal behavior. This is because the abnormal traffic due to attack 1 was not significant compared to background traffic from other ASes. This behavior reveals the limitations of coarser metric aggregation. Aggregating all traffic to a single stream reduces the ability to detect attacks that moderately increase traffic relative to the overall background traffic. The best indicator of attack 1 is C_1, which takes into account the number of AS flows.

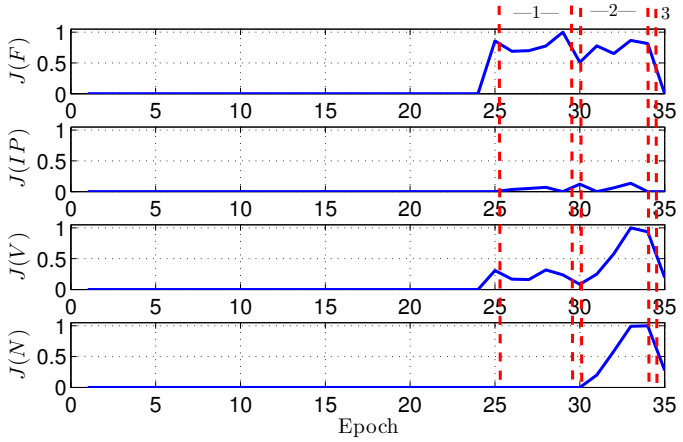
For attack 2, the increased number of AS flows is accompanied by an increase in $J(V)$ and $J(N)$. Hence, all three composite metrics capture this attack. Attack 2 could be characterized as severe, because it increases all metrics despite the averaging effect of background traffic. Finally, attack 3 stays largely undetected, as it only generated internal traffic that is not taken into account at the gateway.

IV. Related work

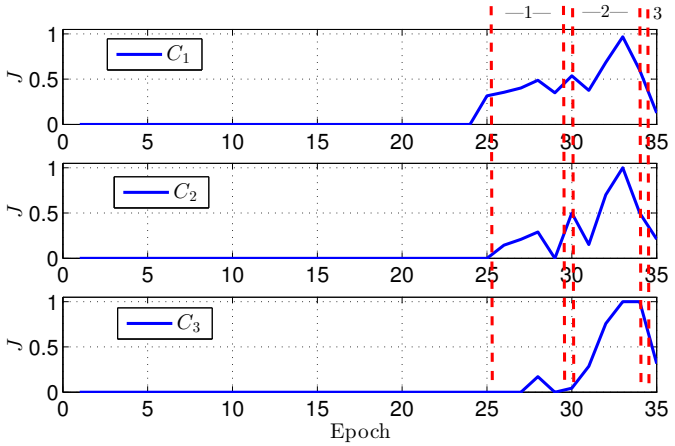
Network intrusion detection systems have been an area of active research for over two decades. Methods for detecting network anomalies can be broadly classified to statistical, classification-based, clustering-based, information theoretic, and spectral [4]. Interested readers are referred to several existing comprehensive surveys on NIDSs [1], [4], [20]. We focus our attention to statistical-based methods, as they are closely related to our work.

Xiang et al. used entropy and information distance metrics to detect low-rate distributed DoS attacks (DDoS) [21]. The proposed system used IP packet size distributions and source IP distributions to detect the DDoS attacks and traceback attaches to their sources. Distributions were compared in terms of entropy and information distance to detect statistically significant differences. All analysis was performed at the IP level. Ping and Abe presented a DoS detection methodology that uses the entropy of the IP packet size distribution. The authors showed that different applications (FTP, HTTP, etc.) cause specific packet size distribution due to the fixed sizes of control packets and an expected size for data packets. Using entropy, they detected distribution changes due to DoS attacks.

Yu et al. employed distance metrics to differentiate DDoS attack flows from flash crowds, by analyzing flow similarity [22]. The premise of their work is that DDoS attacks show a stronger flow similarity compared with the flow characteristics of flash crowds. Different distance metrics were compared in search of the optimal for differentiating DDoS attacks. Muraleedharan et al. presented methods for detecting anomalies on IPFIX (IP Flow Information Export) protocol flows [14]. IPFIX is the standard protocol for exporting IP flow information from routers, probes, and other devices. Flows were analyzed



(a) Basic distance metrics.



(b) Composite metrics.

Fig. 9. Monitored network, 6AM-9AM, Attacks 1, 2, and 3.

using chi-square statistical tests. Similar to other methods, this level required data at the IP level.

Thatte et al. proposed parametric methods for detecting network anomalies using aggregate statistics on IP flows [19]. The authors applied simple statistical models for describing anomalous and background traffic and estimate traffic statistics in real time. They demonstrated high detection rates using only traffic rate and packet-size statistics. Their work is different than ours in that aggregation occurs at a coarser level by considering all traffic that is intercepted at the monitored network (similar to our all-traffic analysis).

Yu et al. developed behavior based anomaly detection methods that detect network anomalies by comparing the current network traffic with a baseline distribution using maximum entropy [8]. Similar to our NIDS operation, their system proceeds in two phases. In the first phase, a baseline distribution model is built using density estimation techniques. In the second phase, anomalies are detected in real time by comparing the KL divergence in different packet classes. The baseline model is updated using a slide window approach. The major difference between the work in [8] and ours is that we operate on AS aggregates as opposed to performing packet-level classification.

V. Conclusions

We developed an NIDS that detects anomalous network behavior using AS flow aggregates. Our system mitigates the storage and computational scalability problems associated with increasing traffic loads at the gateways of monitored networks. By performing volumetric analysis at the AS level, we drastically reduced the state information that must be maintained by the NIDS. We adapted well-established volumetric analysis metrics to the AS domain. We further proposed a simple statistical analysis process that exploits information-theoretic metrics to detect deviations of the online traffic from a baseline model. We adopted a moving windowing technique for updating the baseline models with time. We applied our NIDS on a popular dataset containing annotated attacks and showed the feasibility of detecting various attack types when aggregate metrics are summarized at the AS level.

References

[1] M. Bhuyan, D. Bhattacharyya, and J. Kalita. Network anomaly detection: Methods, systems and tools. *Communications Surveys Tutorials*, 1(99):1–34, 2013.

[2] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita. An effective unsupervised network anomaly detection method. In *Proc. of the Conference on Advances in Computing, Communications, and Informatics*, pages 533–539, 2012.

[3] CAIDA. IP prefix-to-AS mapping comparison. http://www.caida.org/research/routing/prefix_as_comparison/, 2013.

[4] V. Chandola, A. Banerjee, and V. Kumar. Anomaly detection: A survey. *ACM Comput. Surv.*, 41(3):1–15, 2009.

[5] F. D. and G. M. Flexible flow aggregation for adaptive network monitoring. In *Proc. of the Workshop on Network Measurements*, 2006.

[6] P. D. and S. Abe. Detecting DoS attacks using packet size distribution. In *Proc. of the Bio-Inspired Models of Network, Information and Computing Systems Conference*, pages 93–96, 2007.

[7] E. Gregoru, A. Improta, L. Lenzini, L. Rossi, and L. Sani. Inferring geography from BGP raw data. In *Proc. of the Computer Communications Workshops INFOCOM*, pages 208–213, 2012.

[8] Y. Gu, A. McCallum, and D. Towsley. Detecting anomalies in network traffic using maximum entropy estimation. In *Proc. of the SIGCOMM conference*, pages 32–32, 2005.

[9] V. Jyothsna, V. V. R. Prasad, and K. M. Prasad. A review of anomaly based intrusion detection systems. *International Journal of Computer Applications*, pages 8975–8887, 2008.

[10] A. Lakhina, M. Crovella, and C. Diot. Characterization of network-wide anomalies in traffic flows. In *Proc. of the SIGCOMM conference*, pages 201–206, 2004.

[11] L. Luo, G. Xie, S. Uhlig, L. Mathy, K. Salamatian, and Y. Xie. Towards TCAM-based scalable virtual routers. In *Proc. of the conference on Emerging networking experiments and technologies*, pages 73–84, 2012.

[12] MIT Lincoln Laboratory. 1999 darpa intrusion detection evaluation. http://www.ll.mit.edu/mission/communications/cyber/CSTcorpora/ideval/data/1999data.html, 1999.

[13] MIT Lincoln Laboratory. Intrusion detection attacks database. http://www.ll.mit.edu/mission/communications/cyber/CSTcorpora/ideval/docs/attackDB.html, 1999.

[14] N. Muraleedharan, A. Parmar, and M. Kumar. A flow based anomaly detection system using chi-square technique. In *Proc. of the Advanced Computing Conference*, pages 285–289, 2010.

[15] Y. Rekhter, T. Li, and S. Hares. RFC 4271: Border gateway protocol 4, 2006.

[16] B. W. Silverman. *Density estimation for statistics and data analysis*. CRC Press, 1986.

[17] V. Siris and F. Papagalou. Application of anomaly detection algorithms for detecting syn flooding attacks. In *Proc. of GLOBECOM Conference*, 2004.

[18] Y. Sun, H. Liu, and M. S. Kim. Using TCAM efficiently for IP route lookup. In *Proc of the CCNC conference*, pages 816–817, 2011.

[19] G. Thatte, U. Mitra, and J. Heidemann. Parametric methods for anomaly detection in aggregate traffic. *IEEE/ACM Transactions on Networking*, 19(2):512–525, 2011.

[20] J. Wang, D. Rossell, C. Cassandras, and I. Paschalidis. Network anomaly detection: A survey and comparative analysis of stochastic and deterministic methods. *ArXiv e-prints*, 2013.

[21] Y. Xiang, K. Li, and W. Zhou. Low-rate DDoS attacks detection and traceback by using new information metrics. *IEEE Transactions on Information Forensics and Security*, 6(2), 2011.

[22] S. Yu, T. Thapngam, J. Liu, S. Wei, and W. Zhou. Discriminating DDoS flows from flash crowds using information distance. In *Proc.of the CNSS Conference*, pages 351–356, 2009.