Temporally Oblivious Anomaly Detection on Large Networks Using Functional Peers[*]

Kevin M. Carter, Richard P. Lippmann, and Stephen W. Boyer
MIT Lincoln Laboratory
Lexington, MA USA
kevin.carter@ll.mit.edu, lippmann@ll.mit.edu, boyer@ll.mit.edu

ABSTRACT

Previous methods of network anomaly detection have focused on defining a temporal model of what is "normal," and flagging the "abnormal" activity that does not fit into this pre-trained construct. When monitoring traffic to and from IP addresses on a large network, this problem can become computationally complex, and potentially intractable, as a state model must be maintained for each address. In this paper, we present a method of detecting anomalous network activity without providing any historical context. By exploiting the size of the network along with the minimal overhead of NetFlow data, we are able to model groups of hosts performing similar functions to discover anomalous behavior. As a collection, these anomalies can be further described with a few high-level characterizations and we provide a means for creating and labeling these categories. We demonstrate our method on a very large-scale network consisting of 30 million unique addresses, focusing specifically on traffic related to web servers.

Categories and Subject Descriptors

C.2.3 [**Network Operations**]: Network Monitoring

General Terms

Measurement, Security

Keywords

Anomaly detection, network security, machine learning

1. INTRODUCTION

To complement *signature-based* intrusion detection systems (IDS) such as Snort [17] and Bro-IDS [16], there has been much research in the area of *anomaly-based* systems [22, 11], which do not match pre-defined signatures, but rather identify behavior which is rare compared to a data-driven model. This approach provides the benefit of being able to detect zero-day attacks – or deviations of known attacks without a signature – as long as they deviate from the normal traffic patterns. Sommer and Paxson [20] provide a detailed breakdown of several of the reasons why machine learning based anomaly detection methods have failed to reach wide adoption. We address two important weaknesses and develop an anomaly detection method that scales to large networks and provides context for decisions.

In this paper we present a method for anomaly detection on very large-scale networks which is temporally oblivious, e.g. we do not utilize any historic information about a host to determine if it is anomalous. In complete isolation, it is impossible to determine if a host is behaving in an anomalous fashion during a given moment without the historical context that is typically used for anomaly detection. However, in the presence of numerous functional peers – hosts providing similar services – we can identify those hosts exhibiting significantly different behaviors over a static snapshot in time. This intuition follows from the fact that many functions and services have inherent properties which govern their behavior. In our approach, the model used to detect anomalies consists of measurements over the group of peers and automatically compensates for variations in traffic patterns over time. While abnormal by definition, the anomalies detected can be further characterized by a small set of descriptions, and we provide an automated method of providing these qualitative assessments.

The contributions of this paper to network anomaly detection are as follows:

- Temporal modeling is not required, instead we use on-the-fly data driven statistics from other active network hosts performing similar functions.
- Computation is low because only NetFlow [1] features are used and instead of building one reference model per host, only one reference model is required for each general function. This supports scaling to very-large networks.
- After training, anomalies are automatically categorized by a small set of descriptive characterizations, requiring minimal additional computation in a live environment.

[*]This work is sponsored under Air Force Contract FA8721-05-C-0002. Opinions, interpretations, conclusions and recommendations are those of the authors and are not necessarily endorsed by the United States Government.

Table 1: Anomaly Detection Features

Incoming	Outgoing	Ratio (Outgoing/Incoming)
Bytes / Packet (B/P)	B / P	Bytes
Packets / Flow (P/F)	P / F	Packets
Flows / Unique External Source IP (F/SIP)	F / DIP	Flows
# Unique External Source IP (SIP)	DIP	IP addresses

1.1 Previous Work

Some recently presented methods of network anomaly detection focus on detecting volume anomalies which show a sharp change (typically an increase) in the traffic volumes received by a network or host [2, 21, 22]. This approach may miss important traffic variations because the volume increases to an individual host are often masked by variations in normal network traffic [11]. There has been work to detect specific types of malicious volume activity such as port scanning [18], worm and botnet propagation [6, 8], and denial of service attacks [4, 23, 14]. These techniques use properties of machine learning, along with high-level network information (such as NetFlow records) to detect these very specific activities. Additional work has been done to classify internet traffic into specified applications [13]. While payload-oblivious techniques have their natural limitations [7], they scale to large networks. Statistical entropy has been used to detect a more general class of anomalies [10, 11, 12, 15], measuring the change in the distribution of network traffic. These methods have been shown to be very successful at identifying *when* an anomaly occurs on the network, but they require additional post-processing to identify the responsible parties, which has seen additional work [3].

2. DETECTING ANOMALOUS HOSTS

We define an anomalous host as one which exhibits behavior that is significantly dissimilar to other hosts on the monitored network performing similar function(s) during an observation window; stressing specifically that *anomalous* is not necessarily *malicious*. While network traffic is known to be highly variable over short periods for individual hosts, on large networks there are frequently many hosts performing similar functions. This makes it possible to detect anomalies among these functional peers by using the collection of hosts as a reference to detect the few which deviate from this behavior.

2.1 Data and Features

Given our goal for detection on large-scale networks, we utilize unsampled NetFlow data as full packet capture is often unfeasible. From the data, we extract a 12-dimensional feature vector x from each host on the monitored network during a T second observation window. These aggregated flow features are listed in Table 1, where SIP and DIP refer to the number of external unique source and destination IP addresses respectively, and the ratio of IP addresses is computed as DIP/SIP.

With the exception of the number of unique IP addresses communicating with the host, each feature is invariant to the scale of the network or service utilization. This was chosen specifically so that hosts with similar utilization patterns are observed as such, regardless of the number of external hosts accessing the service. However, it is still necessary to include the number of unique IP addresses connecting to the host in order to give some context to the access patterns.

2.2 Identifying Outliers

Measuring Dissimilarity

Given the collection of feature vectors for the N IP addresses on the network $\mathbf{X} = [x_1, x_2, \ldots, x_N]$, we identify anomalies by first defining a dissimilarity measure for each pair of hosts. As each measured feature from Table 1 covers a different range, we normalize each feature to the same relative scale. We choose to normalize each feature such that there is unit distance between the 10th and 90th percentiles of that feature. Specifically, let $f_p(y)$ be defined such that it returns the p^{th} percentile of the data in vector y. For example, $f_{50}(y)$ would be equal to the median of y. We normalize each feature i in $\mathbf{X}$ such that

$$\mathbf{X}(i) = \mathbf{X}(i)/(f_{90}(\mathbf{X}(i)) - f_{10}(\mathbf{X}(i))),$$

where $\mathbf{X}(i) = [x_1(i), \ldots, x_N(i)]$. This normalization ensures that 80% of the mass of data will lie in the same range for the various features, but the outliers will still stand out as we are linearly scaling the data. Once normalized, we calculate pairwise distances between hosts with a standard Euclidean (L_2) distance $D(x_i, x_j) = \| x_i - x_j \|_2$.

Hierarchical Clustering

After calculating the pairwise dissimilarities between hosts, we employ hierarchical clustering [9] to identify outliers. Clustering is performed by first assigning each host to its own cluster or *node*. The two nodes with the smallest linkage cost between them are merged to form a new node, and the process is repeated until all hosts belong to the same node. For this task we use single-linkage clustering, which defines the cost of merging nodes A and B as $\min\{D(a, b) : a \in A, b \in B\}$; the minimum distance between any two hosts in the nodes. This is a logical linkage criterion for anomaly detection as we aim to find those samples which are most dissimilar from others. This method results in a hierarchical cluster tree in which the top of the tree is a single cluster containing all hosts, and the bottom of the tree contains a unique cluster for each host.

Intuition suggests that in the presence of outliers, the final nodes to merge would contain the potential outliers, as the linkage cost will be among the largest of any nodes in the set. We develop a stopping criterion intentionally designed to identify these outliers. Note that the linkage cost $L(i)$ is strictly non-decreasing over iterations i; we stop merging clusters at the point where $L(i) > \alpha L(i-1), \alpha > 1$. For a large enough value of α, this stopping criterion will identify the first significant jump in the linkage cost. When the cost of merging two nodes is a significant gain over the previous merge, any remaining clusters are distinctly different and any sample belonging to a cluster with less than n

members is flagged as an outlier. The threshold n may be defined either as a constant value, or some function of the set size N. By our definition of anomaly, if there is a cluster of activity with $\geq n$ members, those hosts will not be flagged because there is a large enough contingent of hosts exhibiting similar behaviors. Hence, certain activities which are not historically normal to the monitored network may go undetected. This does not mean the activity is not malicious or temporally anomalous, it is simply not anomalous for the monitored network *during this window T*.

2.3 A Network Illustration

We tested our anomaly detection method on seven continuous days of traffic on a monitored network consisting of 30 million hosts. Our data comes from gateway border routers that observe all traffic entering and leaving the network and transmit all flow records to a central repository. We use the SiLK system to query this database and compute aggregate statistics [5]. This centralized system alleviates concerns of asymmetric routing, as we are guaranteed to see both sides of the flow.

As a proof-of-concept, we limit our analysis to incoming traffic with destination port 80 and outgoing traffic with source port 80. This corresponds primarily, although not exclusively, to those IP addresses hosting web services. Note that we do not make a distinction between TCP and UDP for this analysis. On this monitored network, there are roughly 750 hosts receiving applicable traffic at any given time. While we lack appropriate labels for the data during this period, we do have knowledge of two large-scale SYN-flood distributed denial of service (DDoS) attacks against two network web servers during the monitored period.

Our detection parameters are set with the goal of identifying the most egregious anomalies on the network. We apply five-minute aggregation windows (T = 300s) to the data; a window large enough to benefit from consistency due to aggregation, yet small enough to still identify brief anomalies. Defining the optimal value T is a network-specific task and an area for future work. We set our stopping criterion variable at $\alpha = 3$ and our cluster size threshold at $n = 3$, reiterating our goal to find the most significant outliers.

As an illustration, we randomly select a five-minute window and demonstrate the detection results. In Fig. 1, we plot the dendrogram of the hierarchical cluster tree calculated from the extracted features, which shows the manner in which the tree is formed by merging nodes of increasing linkage costs. Due to the size of the network, we do not illustrate all of the unique leaf nodes, beginning instead with nodes that have already been merged. What is clear from Fig. 1 is that there exists a significant jump in the linkage cost near the top of the tree. We plot the cutoff threshold determined by $\alpha = 3$ with the dashed line, resulting in 3 remaining clusters, 2 of which contain only one sample. These two hosts are flagged as anomalous during this process. When run over the entire seven day window, this method resulted in 1,658 flagged anomalies out of a possible 1.5 million host samples, flagging an average of 0.11% of monitored hosts (0.8 hosts) as anomalous during a given observation period. In 47% of the observation windows, no hosts were flagged as anomalous.

3. GROUPING THE ANOMALIES

It is of particular interest to see if common anomalies occur at different time intervals and over different hosts in the network. This is not obvious, as by definition the behavior is not within the network norms. To determine this, we gather the feature vectors of all 1,658 anomalies flagged during the analysis of Section 2.2 in a set $\mathbf{Z}$, and perform additional clustering on this set. Each individual anomaly was detected in a temporally oblivious manner, and follow-on analysis determines if there are common themes amongst those detected.

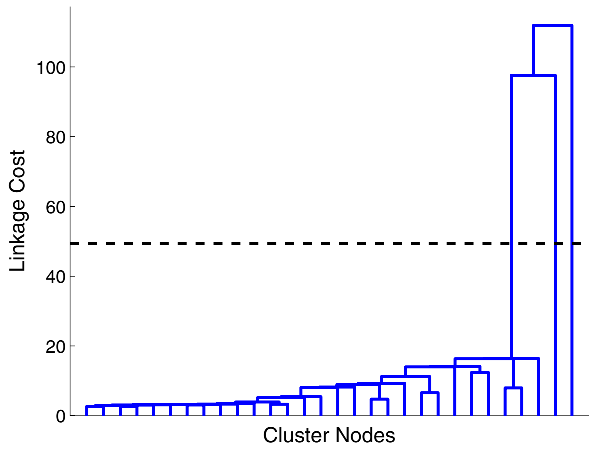

Figure 1: Dendrogram of the hierarchical cluster tree resultant from a single time window. The dashed line shows the stopping point, resulting in two detected outliers.

Normalizing the Data

To obtain categories for the collected outliers, it is necessary to normalize in a way that preserves the high-level description of the data. For example, if the vast majority of hosts receive less than F flows per unique external source IP, and two distinct hosts receive $100F$ and $500F$ flows in a five minute window, they should be treated as equals – they are anomalous due to large received F/SIP. To account for this issue, we normalize the data by a *sigmoid* function $s(x)$, which translates every value $x \in [-\infty, \infty]$ to $s(x) \in [0, 1]$:

$$s(x) = \frac{1}{1 + \exp(-(x - \mu)/b)}.$$

We omit the full details for brevity, but we set our parameters $\mu = f_{50}(\mathbf{Z}(i))$ and b such that the end of the linear portion of $s(x)$ (e.g. the bend point [19]) occurs at $x = f_{90}(\mathbf{Z}(i))$. We set the bend point to the 90th percentile of the data such that the extreme outliers will be truncated to near unity while still linearly scaling the majority of the data samples.

We demonstrate this scaling for the incoming B/P feature in Fig. 2, where we plot the scaled histogram of values alongside the sigmoid normalization function. This plot shows that the sample points far away from the mass of the distribution will be quantized to nearly 1, regardless of the magnitude of their distance.

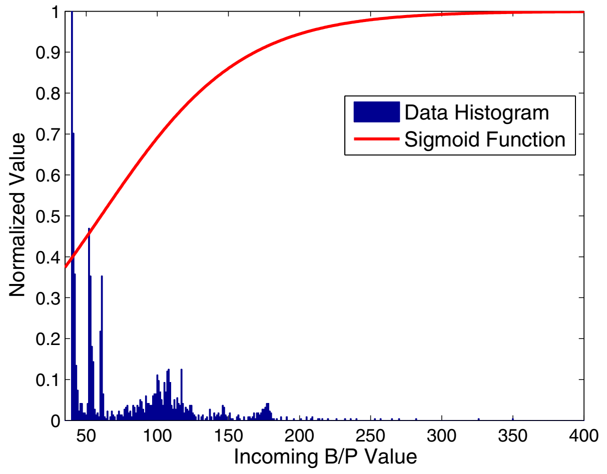

Figure 2: Normalizing the data features of the outliers using a sigmoid function. This offers linear scaling for the mass of the data while quantizing the extreme outliers.

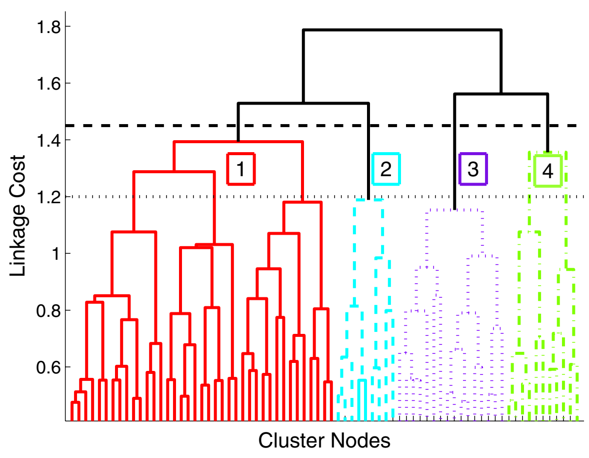

Figure 3: Dendrogram of the hierarchical cluster tree resultant from all anomalies over a 7 day period. The choice of cutoff threshold determines the cluster granularity.

Clustering the Anomalies

We proceed by performing hierarchical clustering on the normalized data using a complete-linkage criterion. The cost of merging nodes A and B is defined as $\max\{D(a,b) : a \in A, b \in B\}$, which is more useful for identifying similar clusters rather than outliers. Rather than assigning our cutoff threshold by identifying the jump in linkage cost as before, we qualitatively assess the dendrogram to determine the appropriate threshold. This has the benefit of enabling us to specify the level of detail which we are interested in describing the anomalies.

Visual inspection of the abbreviated dendrogram in Fig. 3 shows that we can glean a high-level description of the outliers by setting our threshold to 1.45, which results in 4 clusters covering all of the anomalies. In order to identify the common theme of each cluster in a quantitative sense we applied a 2-class decision tree for each cluster, in which members of the cluster belonged to the positive class, and all other vectors are given a negative label. While we omit the quantitative results, the decision rules were quite simple, accurately classifying the large amount of anomalies generally based on three primary features: byte ratio, IP ratio, and outgoing bytes/packet. We now provide a brief qualitative assessment of the type of traffic we see in each cluster:

Cluster 1: As the largest cluster, the hosts on this cluster exhibit several different patterns, and we noticed 3 distinct groups. However, the key feature is that 99% of the 1,008 host samples in the cluster sent traffic to more IP addresses than they received it from. The first group we observed in this cluster contains those hosts which received a large number of low-byte packets in a single flow from a single IP address, and responded with a much larger volume of traffic. An example of this would be a host that received a single flow with 26,000 packets each containing 40 bytes (1 MB total) from a single IP address, and responded to the same address with 30 MB of traffic.

The next group consists of hosts receiving a large amount of traffic from a single source IP through an abnormally large number of flow records, and responding in kind. An example of this would be a host which received 250 KB of data from a single IP across 350 flows. We can infer from this type of traffic that the source IP in question operates as a network address translation (NAT) or proxy server, and there are potentially numerous unique hosts using the same externally visible IP address.

Finally, the third group is clearly DDoS activity, as the hosts receive traffic from numerous unique source IP addresses each sending a significantly large number of flows containing only 2-3 packets. The servers are responding to more hosts than they are receiving traffic from, signifying that they are able to keep up for the time being. An interesting aspect about this cluster is that the DDoS attacks on the two servers mentioned earlier cluster together, even though they were carried out in different manners.

Cluster 2: The hosts in this cluster receive very low volumes of traffic from very few sources, yet send substantial volumes to more IP addresses than they received traffic from. In fact, 97% of the 94 hosts in this group sent outgoing traffic to more IP addresses than they received incoming traffic from. Example: A host received 735 bytes across 7 packets in a single flow from a single source and sent 195 MB of traffic across 140,000 packets in 2 flows to 2 unique IP addresses.

Cluster 3: The hosts in this cluster received high volumes, with a median of 595 MB of traffic during the observation period – for reference the median received traffic in the other clusters was 2.2 MB, 800 bytes, and 306 KB respectively. This could imply the hosts allow file uploads. Additionally, 99% of 402 hosts in this group sent outgoing traffic to less destinations than they received incoming traffic from. Within this cluster, we noticed subgroups containing high-volume unidirectional traffic entering the network which had no observed responses. This included DDoS traffic where the servers under attack had either crashed or stopped accepting traffic. There was additionally DDoS traffic for which there were responses, although the response volume was always less than the incoming traffic. This may imply a DDoS for which the server is unable to keep up with requests.

Cluster 4: The hosts in this cluster behaved similarly to those in Cluster 2; receiving low volumes of traffic while sending significantly larger ones. This cluster is differentiated by the fact that no host sent traffic to more IP addresses than it received traffic from. In fact, 66% of the 154 hosts sent traffic to fewer hosts. There was an additional group within this cluster of hosts that received traffic from a few sources, comprised of large numbers of small byte packets in a minimal number of flows, yet still sent a large volume of traffic in return, approaching the maximum transmission unit (MTU) size of the network (1,500 bytes/packet). This appears to be large file-transfer traffic, which would explain the large outgoing packet sizes, low flow counts, and small incoming packet sizes (which would simply by ACKs by the external host).

Our cluster descriptions provide a high-level understanding of the different types of detected anomalies. While there were many "repeat offenders", 172 of the 1,658 anomalies were attributed to unique IP addresses, and each cluster contained at minimum 39 unique IP addresses. A finer categorical granularity can be achieved by simply decreasing the cutoff threshold for the hierarchical cluster tree. For example, when reducing the threshold to 1.2, Cluster 1 is now split into the three clusters we previously observed as shown in Fig. 3

3.1 Classifying New Anomalies

If new anomalies could be detected without a new clustering analysis, the qualitative categorization presented would be of high value to analysts; abstracting them from the technical and statistical details concerning clustering. By training classifiers on anomalies detected over a period of time, we can provide a label along with a confidence for each newly detected anomaly. In the absence of ground truth, we develop our classifier by using the previously clustered anomalies as our training set. For this study, we train on the high-level cluster labels represented in Fig. 3 (four classes), and implement a simple linear classifier on the normalized anomaly data; choosing decision boundaries via linear discriminant analysis. This results in a 1.69% classification error on the training set. Note that this error is in classifying hosts which have already been flagged as anomalous, so there is a drastically lower cost for misclassification than with a traditional IDS.

We test this classifier over seven days of traffic occurring 8 weeks after the data we trained on. We normalize the newly flagged anomalous data in the same manner, still using the f_{50} and f_{90} values from the training data in order to preserve the appropriate scaling. There were 2,411 out of a possible 1.4 million hosts flagged during this testing period, yielding an average of 1.24 hosts being flagged as anomalous during an observation window. After classifying each anomaly, we note that the types of activity in each class are consistent with those during the training period. We can quantify the confidence $c(x)$ in these labels by using the posterior probability that a sample x_i belongs to a specific class C_j, which is denoted as $\mathrm{P}(C_j \mid x_i)$. As each sample is labeled according to the class C_j which maximizes this value, we can set the confidence of our classification label as $c(x_i) = \max_j \mathrm{P}(C_j \mid x_i)$. For the test data, 95.5% of the detected anomalies had $c(x) > 90\%$, confirming that the general categories of anomalies on the network did not change. It is worth noting that 66% of the anomalies detected during this testing phase were from IP addresses that were *never* flagged as anomalous during the training phase. It is not simply the same hosts consistently exhibiting the same anomalous behavior, but different hosts behaving in manners which fall under persistent categorizations. There was one sample with a confidence of $< 50\%$ ($c(x) = 42\%$). A sample with a confidence this low may represent a new category of anomaly, and a method for culling this information automatically is an area for future work.

4. DISCUSSION

We've presented what we deem to be very promising results, yet our work has a few shortcomings which we now address. First, while we provide intuition for our chosen parameter values, α and n, a more rigorous approach would be necessary for a full deployment. Given the nature of anomalies, and their lack of expert-defined labels, there is no straightforward way to define these parameters in an automated fashion. A network administrator deploying a system such as ours would have to train the parameters to yield an "acceptable" amount of alerts per observation period, as is done in many IDS and anomaly detection systems, and determine which classes of anomalous traffic are potentially benign. The detected anomalies could then be filtered based on their class and network knowledge (eg. NATs) to reduce the alerts an analyst must investigate.

There is a chance that a previously unseen anomaly could occur which is not accurately described by the previously trained classifier. This is the nature of anomaly detection, yet our system has the tools to account for this. Specifically, the classification confidence of such an anomaly would be predictably low (as shown in Section 3.1), and these types of anomalies could be saved for retraining the system. As with any classifier, periodic retraining is necessary if the statistics of the feature space change, and that will almost always be the case for network data. It is future work to determine how often this retraining would be necessary.

While our method is specifically *not* an IDS, we realize that anomaly detection is often used for that purpose and we want to detail how an adversary could circumvent this system. A sophisticated attack could have an adversary intentionally manipulate the traffic of multiple servers at the same time in the same manner, causing them to form a large enough group in the feature space that would not be deemed anomalous. This would require knowledge of the algorithm parameters, specifically the minimum cluster size n, and would also increase the required resources for the adversary.

5. CONCLUSIONS

In this paper we have described a temporally-oblivious approach for detecting anomalous hosts on large-scale networks. By modeling the behavior of functional peers, anomalies stand out and can be described by a small set of qualitative characterizations. The fact that anomalous activity, which by definition is significantly abnormal, is categorically consistent over time is intriguing. We utilized this discovery to develop a labeling system for anomalies discovered during new observation periods. This framework was trained over seven days and tested over an additional seven days nearly 2 months later, with very similar results.

This work is presented as a proof-of-concept of a novel method of detecting anomalies. We note that hierarchical clustering is an $O(N^2)$ operation, meaning that it will not purely scale to large N. This is why we operate our system on a *per service* basis, focusing on port 80 in this study. While port numbers do not restrict activity, many are reserved for specified functions, so this split is logical for anomaly detection. Since networks are made of a finite number of services of interest, this split also enables scaling to very large-scale networks. Determining a manner to cluster across services is an area for future work. Additionally, we would like to identify clusters of activity within the non-anomalous traffic, seeing whether these clusters remain consistent over time in a similar fashion as the clusters of anomalous activity. This naturally lends itself to many research areas in network behavioral analysis. The nature of the clusters themselves provides network situational awareness, and identifying the movement of hosts between clusters could provide additional information.

Acknowledgements

We would like to thank the reviewers for their thoughtful comments and William W. Streilein of MIT Lincoln Laboratory for his help with our normalization methods.

6. REFERENCES

[1] Cisco netflow. http://www.cisco.com.

[2] Barford, P., Kline, J., Plonka, D., and Ron, A. A signal analysis of network traffic anomalies. In *Internet Measurement Workshop* (2002), pp. 71–82.

[3] Brauckhoff, D., Wagner, A., and Salamatian, K. Anomaly extraction in backbone networks using association rules. In *Proceedings of 9th ACM SIGCOMM Internet Measurement Conference* (2009), pp. 28–34.

[4] Carl, G., Kesidis, G., Brooks, R. R., and Rai, S. Denial-of-service attack-detection techniques. In *IEEE Internet Computing* (Jan./Feb. 2006), vol. 10, pp. 82–89.

[5] CERT/NetSA at Carnegie Mellon University. *SiLK (System for Internet-Level Knowledge)*. [Online]. Available: http://tools.netsa.cert.org/silk.

[6] Collins, M. P., and Reiter, M. K. Hit-list worm detection and bot identification in large networks using protocol graphs. In *Proceedings ofthe 10th International Symposium on Recent Advances in Intrusion Detection (RAID)* (2007).

[7] Collins, M. P., and Reiter, M. K. On the limits of payload-oblivious network attack detection. In *Proceedings of 11th International Symposium on Recent Advances in Intrusion Detection (RAID)* (2008), pp. 251–270.

[8] Gu, G., Perdisci, R., Zhang, J., and Lee, W. Botminer: Clustering analysis of network traffic for protocol- and structure-independent botnet detection. In *Proceedings of 17th USENIX Security Symposium* (2008), pp. 139–154.

[9] Hastie, T., Tibshirani, R., and Friedman, J. *The Elements of Statistical Learning*. Springer, July 2003.

[10] Lakhina, A., Crovella, M., and Diot, C. Characterization of network-wide anomalies in traffic flows. In *Proceedings of ACM/SIGCOMM Internet Measurement Conference* (2004), pp. 201–206.

[11] Lakhina, A., Crovella, M., and Diot, C. Diagnosing network-wide traffic anomalies. In *Proceedings of ACM SIGCOMM* (2004), pp. 219–230.

[12] Li, X., Bian, F., Crovella, M., Diot, C., Govindan, R., Iannaccone, G., and Lakhina, A. Detection and identification of network anomalies using sketch subspaces. In *Proceedings of 6th ACM SIGCOMM Conference on Internet Measurement* (2006), pp. 147–152.

[13] Moore, A. W., and Zuev, D. Internet traffic classification using bayesian analysis techniques. In *Proceedings of 2005 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems* (2005), pp. 50–60.

[14] Moore, D., Voelker, G., and Savage, S. Inferring internet denial-of-service activity. In *Proceedings of the 10th Usenix Security Symposium* (2001), pp. 9–22.

[15] Nychis, G., Sekar, V., Andersen, D. G., Kim, H., and Zhang, H. An emperical evaluation of entropy-based traffic anomaly detection. In *Proceedings of 8th ACM SIGCOMM Conference on Internet Measurement* (Oct. 2008), pp. 151–156.

[16] Paxson, V. Bro: A system for detecting network intruders in real-time. *Computer Networks 31*, 23–24 (1999), 2435–2463.

[17] Roesch, M. Snort - lightweight intrusion detection for networks. In *Proceedings of 13th LISA Conference* (1999), pp. 229–238.

[18] Schechter, S. E., Jung, J., and Berger, A. W. Fast detection of scanning worm infections. In *Proceedings of the 7th International Symposium on Recent Advances in Intrusion Detection (RAID)* (2004), pp. 59–81.

[19] Sebaugh, J. L., and McCray, P. D. Defining the linear portion of a sigmoid-shaped curve: Bend points. In *Pharmaceutical Statistics*, vol. 2. 2003, pp. 167–174.

[20] Sommer, R., and Paxson, V. Outside the closed world: On using machine learning for network intrusion detection. In *Proceedings of 31st IEEE Symposium on Security and Privacy* (May 2010).

[21] Subhabrata, B. K., Krishnamurthy, E., Sen, S., Zhang, Y., and Chen, Y. Sketch-based change detection: Methods, evaluation, and applications. In *Proceedings of ACM SIGCOMM Internet Measurement Conference* (2003), pp. 234–247.

[22] Thottan, M., and Ji, C. Anomaly detection in ip networks. *IEEE Transactions on Signal Processing 51* 8 (Aug. 2003), 2191–2204.

[23] Wang, H., Zhang, D., and Shin, K. G. Detecting syn flooding attacks. In *Proceedings of the IEEE Infocom* (2002), pp. 1530–1539.