The Need For Simulation In Evaluating Anomaly Detectors

Haakon Ringberg
Princeton University

Matthew Roughan
University of Adelaide

Jennifer Rexford
Princeton University

This article is an editorial note submitted to CCR. It has NOT been peer reviewed.
Authors take full responsibility for this article's technical content.
Comments can be posted through CCR Online.

ABSTRACT

Anomalous events that affect the performance of networks are a fact of life. It is therefore not surprising that recent years have seen an explosion in research on network anomaly detection. What is quite surprising, however, is the lack of controlled evaluation of these detectors. In this paper we argue that there are numerous important questions regarding the effectiveness of anomaly detectors that cannot be answered by the evaluation techniques employed today. We present four central requirements of a rigorous evaluation that can only be met by *simulating* both the anomaly and its surrounding environment. While simulation is necessary, it is not sufficient. We therefore present an outline of an evaluation methodology that leverages both simulation *and* traces from operational networks.

Categories and Subject Descriptors

C.2.3 [**Computer-Communication Networks**]: Network Operations; C.4 [**Performance of Systems**]:

General Terms

Experimentation, Performance, Measurement

1. INTRODUCTION

Automated detection of anomalies in computer networks has been a hot topic for several years. Approaches have been presented [2, 8, 6] to detect anomalies such as worms, DDoS attacks, and IP scans. Despite this flurry of anomaly-detection papers, effective ways to validate and compare the proposed solutions have remained elusive. This is particularly important for statistical techniques, where one typically aims to detect variations from "normal" behavior, rather than a specific signature (as is sometimes the case in the security arena). In this paper we argue that existing evaluation techniques are unable to answer numerous important questions regarding the effectiveness of anomaly detectors; only by simulating both the anomalies and the environment within which they occur will we be able to do so.

In order to quantify the accuracy of a detector it is necessary to first identify a set of "true" anomalies that *ought* to be found by the detector. This set must obviously be identified by a procedure that is independent of the detector being evaluated. By far the most common way to accomplish this identification is to rely on manual labeling of traces by domain experts [1, 8, 11, 5]. In this procedure, the human domain expert inspects a trace and certifies some events as

being true-positive anomalies. The detector is then evaluated based on its ability to identify this set of events. Manual labeling is a natural first evaluation technique in a new field because it builds expertise with a new source of data and this knowledge of the properties of anomalies is essential in order to later automate detection.

Network anomaly detection is now a more mature field, however, with more related publications than we have space to enumerate. Unfortunately, existing evaluation techniques do not allow us to adequately compare the relative strengths of detection algorithms. Manual labeling, for example, is an insufficient evaluation methodology for many reasons: (1) it may be impossible to share the underlying traces (e.g., due to proprietary data or privacy concerns), (2) even domain experts are flawed and may miss true anomalies, (3) manual processes do not scale to the magnitude necessary to identify rare but important anomalies, and (4) the reliance on a fixed trace prevents us from performing sensitivity analysis, for example to determine how "large" an anomaly must be for it to be detected. The central argument of this paper is that anomaly detection research should improve the standard of rigor in future publications, and we identify simulation as a way this can be achieved.

The rest of this paper is organized as follows. In Section 2 we enumerate some high-level requirements of a thorough evaluation of anomaly detectors that are not always met by currently employed evaluation methodologies. In Section 3 we show that simulation can satisfy each of these requirements. This fact notwithstanding, simulation is not sufficient in and of itself, and thus we present an outline of a complete evaluation methodology and the roles that both simulation and real traces have to play. We conclude in Section 4.

2. REQUIREMENTS

In this section we will enumerate four requirements of a complete evaluation methodology for anomaly detectors.

2.1 Ground Truth

"Ground truth" in the context of network anomaly detection requires a *complete* list of all anomalies that exist in a given data set. While the requirement itself might seems obvious, it is much less clear how to obtain this ground truth. Identifying the true-positive anomalies requires combing through vast amounts of data that are sometimes of poor quality due to data-reduction techniques such as sampling. In addition, the anomalies themselves are a moving target (e.g., new worms are discovered on a regular basis), and often hard

to distinguish (e.g., DDoS attacks versus flow crowds). The challenges of obtaining high-quality data have led to many compromises in the evaluation of anomaly detectors, which in turn leads to "partial" ground truth.

However, there are important problems with partial ground truth. Given only a partial list of anomalies, it is impossible to calculate an accurate False-Negative Probability (FNP) because without a complete list of anomalies we cannot know the number of missed detections. Similarly, we cannot accurately quantify the False-Positive Probability (FPP) with only partial ground truth. Without a complete list of anomalies, it is impossible to distinguish between a false alarm (i.e., a false-positive) and an anomaly that your partial list didn't capture.

Moreover, it is important to realize that the FPP and FNP are mutually dependent, i.e., providing either one without the other is meaningless even when they are calculated from a complete list of anomalies. The reason for this is that most detection methods allow a tradeoff between the FNP and FPP over the complete range $[0, 1]$. Reporting that an algorithm has a false-negative probability of 10% is therefore meaningless without a corresponding false-positive probability. For example, it is trivial to construct an algorithm that does not miss a single true anomaly (i.e., has a false-negative probability of 0%) by asserting everything to be anomalous. Clearly this would be a poor algorithm since it would produce unbearably many false alarms.

Even providing both the FNP and FPP from a complete list of anomalies is insufficient for a thorough comparison of two anomaly detection systems. The reason for this is that a pair of (FNP, FPP) measurements represent only a single point on a curve that captures the complete tradeoff between the two performance metrics. The curve is sometimes called the Receiver Operating Characteristic, or ROC curve, and it and its derivatives have long been preferred in many other fields [4]. Figure 1 demonstrates the arbitrariness of comparing detectors using a single point along the ROC curve (and its associated FPP and FNP [1]). The fact that the two ROC curves intersect means that neither algorithm is absolutely superior to the other; the choice between the two algorithms depends on one's preferred tradeoff between false alarms and missed detections.

In addition to requiring a complete list of all anomalies, ground truth also requires detailed information about each anomaly, viz. its location (both spatial and temporal), magnitude, and type. Location information allows one to evaluate whether a detector is able to properly diagnose anomalies that might appear to be separate but are in fact correlated, such as DDoS attacks. The time and duration of an anomaly is necessary in order to calculate the detection delay. It is also important to consider the magnitude of an event: firstly, because it may be that larger events are more important to detect. More generally, it is important to investigate the sensitivity of a given detector to the magnitude of anomalies. Finally, it is natural to assume that different detectors will excel at finding various types of anomalies, which means that it is important to know the exact type or classification of the anomalies in a trace.

Automated injection of anomalies into traces is one technique that has been employed by some projects in order to provide a list of authoritative anomalies [8, 15]. Anomaly

---

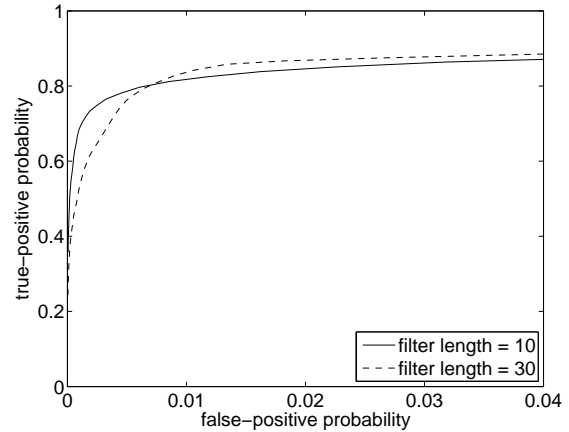[1]True-Positive Probability = 1 - False-Negative Probability

Figure 1: Example ROC curves. Signal: white Gaussian noise with a single injected anomaly (a rectangular pulse of random location, height, and duration). Detection by smoothing the data using a moving average (of two different lengths) and then thresholding at a range of values.

injection leverages models of anomalies in order to introduce them into traces taken from operational networks. It cannot guarantee accurate FPP and FNP measurements due to its reliance on existing traces. That is, these existing traces presumably came with an unknown number of anomalies. As we have already argued, neither automated algorithms nor human domain experts can identify all these anomalies with complete confidence—not even network operators [1, 5]. It will therefore be impossible for any evaluation technique that relies on traces from operational networks to guarantee an accurate FNP.

2.2 Reproducibility

The ability to reproduce an experiment is a central tenet of the scientific method. A researcher might wish to (1) verify published results by evaluating the same algorithm on the same data, (2) investigate the robustness of a published algorithm by applying it to different data, or (3) compare a novel algorithm against the published one by using the same data.

Reproducibility is particularly important in the context of evaluating network anomaly detectors because some of the events we wish to detect are exceedingly rare. Also, for a practical anomaly detection algorithm the FPP must be very small. Consider a typical scenario: a network operator runs an anomaly detection algorithm on each of the 1000 edge links in her network (a large network can easily have 10's of thousands edge links), each monitored using standard five-minute SNMP (Simple Network Management Protocol) traffic measurements. This represents 288,000 samples per day. If the FPP is only 10^{-4}, this will still result in around 28 false alarms per day. Even this number of false alarms may be prohibitive: the observed reaction of network operators to a system that generates too many false alarms is that they either ignore it, or turn it off. If it takes on the order of 20 minutes, say, to determine the truth (or otherwise) of an alarm, then an operator could handle on the order of 50 alarms of any sort in a day (assuming they had no other tasks). Many alarms would take much longer than 20

minutes to sort out. The result is that the FPP must be very small. Typical goals may be of the order of 10^{-4}, 10^{-5}, or even 10^{-6}. However, measuring small probabilities has challenges, as we shall show here.

Small occurrence probabilities require that we must be able to reproduce experiments many times over in order to gain confidence in detection performance. That is, one cannot draw general conclusions about the FNP or FPP from single experiments if the chance of seeing a given alarm is exceedingly small. For example, the fact that a detector returns the one-and-only relevant anomaly in a given trace does not mean that it has FPP = FNP = 0%; it may simply mean that we have not evaluated the detector on enough data.

Consider the simple scenario where we wish to estimate the FPP p of our "anomalous packet detector". We therefore evaluate the detector on a trace with n samples (packets) that we assume to be independent. The number of false alarms generated would therefore follow a binomial distribution, i.e., the probability of k false alarms is

$$p(k) = \binom{n}{k} p^k (1-p)^{(n-k)}.$$

The maximum likelihood estimator for p, given a binomial distribution is simply $\hat{p} = k/n$, and $E\{\hat{p}\} = p$ (so the estimator is unbiased). There are a number of approximations one can make to the binomial distribution, a common one being a Gaussian distribution with mean np, and variance $np(1-p)$. Gaussianity is a convenient simplifying assumption for calculating errors: the 95th percentile confidence intervals for the estimate can be easily calculated to be $\pm 1.96\sqrt{p(1-p)/n}$. However, this approximation is only reasonable for large n. A common rule of thumb is that $np(1-p) > 9$ is required to make the approximation reasonable. Clearly, as p becomes small, the factor of $(1-p)$ can be ignored, and we require $n > 9/p$, which for $p = 10^{-4}$ would require $n = 90,000$ trials.

This inverse relationship between the number of trials and p requires many more trials than is often performed. For instance, it is common to examine a week of five-minute SNMP data, which would contain 1440 trials. However, the real number of trials one may require is even worse. Consider that we may aim to make enough measurements to bound the error. For instance, let us set the apparently reasonable goal to bound the errors in $\hat{p}$ by $\pm 10\%$. The relative errors in estimates (as estimated from the confidence intervals) will be

$$\text{error} = 1.96\sqrt{p(1-p)/n}/p \simeq 2/\sqrt{np}.$$

So we require $n > 4/(p \times \text{error}^2)$. Our apparently modest requirement that error ≤ 0.1 results in $n > 400/p$. Again using $p = 10^{-4}$ this results in four million trials. Figure 2 illustrates the problem with a set of simulated $\hat{p}$ measurements (for $p = 10^{-4}$), where the number of trials n is shown on the y-axis. The plots show histograms of the values of $\hat{p}$ over 100 simulated experiments, and we can see that for $n \leq 10^4$ the resulting distribution is discrete and often provides an inaccurate measure of p (e.g., the probability of estimating that $p = 0$ is significant). When $n = 10^5$, the distribution of values somewhat resembles a Gaussian distribution but errors are roughly $\pm 100\%$. It is not until $n = 10^6$ that we obtain a reasonably accurate measure of p.
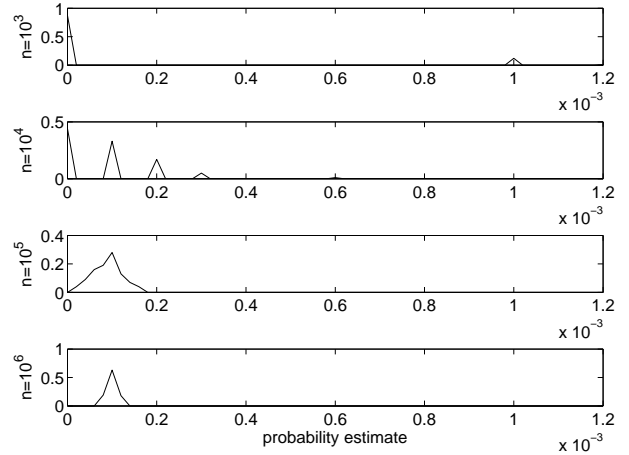

Figure 2: A set of simulated $\hat{p}$ measurements (for $p = 10^{-4}$), where the number of trials n is shown on the y-axis. The plots show histograms of the values of $\hat{p}$ over 100 simulated experiments.

There are many more sophisticated statistical techniques for estimating or bounding such probability estimates (e.g., Neyman-Pearson tests), but these do not avoid the basic point that many samples are needed to provide accurate bounds on FPP estimates. Typical data sets will not have enough data points to find the FPP to the degree of accuracy we require, simply because providing ground truth data (by manual classification) is so time consuming. On the other hand, simulation provides virtually unlimited data sets.

The problem of reproducibility in network anomaly detection is particularly dire due to a general lack of public data sets (particularly those that have some annotation as to ground truth). While some researchers have made commendable efforts to share both their software and evaluation traces (e.g., Lakhina *et al.*), most have not. The problem is exacerbated because most detectors are evaluated using traces from operational networks, and there are numerous valid reasons why such traces cannot be shared with the community. A significant fraction of these traces come from commercial networks, which means that both the data and software is likely proprietary. Even for traces from educational networks, there will be privacy concerns. Furthermore, there are stringent laws that restrict the distribution of certain types of telecommunications data. Finally, traces can often be on the order of many terabytes and it may be practically infeasible to share them. Traces can grow to this size because modern networks carry vast amounts of traffic and therefore any algorithm that is claimed to be able to operate in an online setting should be evaluated on representative traces. Simulation avoids this issue by sharing code to generate the large sample sets required.

2.3 Anomaly Definitions

Network anomaly detectors can be split into two groups: (1) those that aim to find very specific anomalies such as worms or DDoS attacks [18, 13], and (2) those detectors that aim more broadly to to find all anomalies that deviate from what is expected or "normal". Examples of the latter category include detectors that define anomalies as

being hierarchical heavy hitters [2, 17] and those that build a statistical model of expected traffic behavior [1, 8, 15].

The tension between the above two classes is inherent to network anomaly detection. That is, on the one hand, anomalies themselves are a moving target because fundamentally new attacks are sometimes discovered and old attacks mutate. One might therefore want a flexible detector that can adapt to the changing environment. On the other hand, it would not be unexpected if a specialized detector, e.g., focusing only on identifying DDoS attacks, were to outperform general-purpose detectors for their specific tasks.

While one can legitimately debate the merits of each class of detectors, it is impossible to *evaluate* a detector unless one specifies what it is one is looking for. In fact, this requirement is a precondition for both ground truth and reproducibility. That is, an evaluation needs a specification of what ought to be found in order to be *reproducible* or otherwise one will not be able to perform the same evaluation using other traces. Likewise, *ground truth* requires that one find all the anomalies that exist in a given trace, which also implies that we need a description of what constitutes a bona fide anomaly.

2.4 Experimental Control

There are important questions about the effectiveness of network anomaly detectors that cannot be answered without having complete control of the entire evaluation experiment. Having complete control requires that one has the power to change the location, magnitude, and type of individual anomalies as well as for the background traffic. Sample questions that require this type of control include:

- How many destination IP addresses must be targeted before a host scan is identified by the detector?
- If the detector requires training in order to build a model of normal network behavior, what is the impact of anomalies occurring during this training period?
- What would be the impact on a detector's FNP if IP flows were sampled at a rate of 1-in-100 instead of 1-in-1000?
- Can an adversary evade detection if she knows the thresholds/parameters used by the algorithm?

Each of these are important questions, and they all require control over the experiment. Manual labeling would be unable to provide such control due to the fixed nature of the underlying trace. The only previously employed evaluation methodology that provides some measure of control is automated anomaly injection into existing traces. For anomaly injection the control extends only to the anomaly itself, however; not the background traffic, which is static and included in the existing trace. This restricted control precludes study of the side-effects of an anomaly, such as the delay or congestion it causes. Anomalies do not occur in a vacuum, and it is therefore essential to capture and study these interactions between anomalies and the background traffic.

Complete control over the evaluation experiment is also necessary in order to train and test the detector on clean data. That is, an evaluation methodology that leverages existing traces cannot guarantee that all anomalies in those traces have been identified. This can obscure the true performance of an anomaly detection algorithm both because the algorithm *trains* on unclean data (which can degrade its performance [11]) and because it is *evaluated* on unclean data.

Only through simulation can one ensure that all anomalies in an evaluation trace are known.

3. EVALUATION METHODOLOGY

In this section we will describe what we mean by simulation, and how it satisfies the requirements laid out earlier. In fact it is the only approach we are aware of that satisfies all of these requirements. However, although necessary, simulation is not sufficient for rigorous assessment of anomaly detection.

3.1 Simulation

Simulation has played a vital role in networking research over the years. Unmitigated access and control over real networks is presently only a dream for networking researchers, which is why simulation is necessary in order to evaluate some aspects of complex ideas. Simulation environments in networking are typically event based, i.e., they model discrete events such as the arrival and departure of packets. Some anomaly detection algorithms may require us to perform this level of simulation. For instance, consider the case of trying to detect traffic spikes by measuring RTTs (round-trip times): networks buffer packets during congestion, which cause RTTs to increase. A simulation environment for this type of detector would need to account for packet dynamics.

On the other hand, many anomaly detectors analyze aggregated measurements, such as SNMP traffic data collected at five-minute intervals, or NetFlow records. In this case, it may be simpler, faster, and even more accurate[2] to simulate the aggregates directly. It is certainly true that simpler simulation environments provide for more straightforward causality determinations. That is, the diagnosis required to find the cause of some set of observations is certainly made easier if the simulation environment has fewer parameters and more intuitive output.

There are two parts to simulation in this context: (1) simulation of that which is "normal" (e.g., background traffic) (2) simulation of an anomaly (e.g., a port scan). In general we do not know how to do either, but in specific cases we have very good ideas. For instance, measurements have now shown that fractional-Gaussian Noise (fGN) is a reasonable model for highly-aggregated backbone traffic. Several studies (e.g., see [12, 9]) have considered the important parameters of such traffic (the mean and how it varies over time, variance, and Hurst parameter). Likewise, there are specific instances where the idea of an anomaly is well defined (e.g., traffic spikes resulting from DoS attacks).

One of the primary advantages of using simulation for evaluation is the ability to perform multi-factor experiments across several dimensions. One can perform sensitivity analysis on a detector in terms of anomaly size, data-reduction techniques, burstiness of background traffic, etc. The complete control that simulation offers also allows one to investigate the impact of change to the data collection infrastructure. Evaluating the impact of fundamental changes to sampling, filtering, or data aggregation on network anomaly detection is only presently possible through simulation.

---

[2]Higher aggregates of traffic often display simpler (to model) characteristics, and as such it is easier to generate these directly than by creating many inaccurate sub-models.

The scale and heterogeneity of the Internet lead to well-known challenges in simulation [3]. For the purposes of evaluating network anomaly detectors, however, it is important to realize that we often need to simulate only a relatively small part of this large system. In fact, in many cases it will not even be necessary to simulate the detector. That is to say, the simulation environment could be leveraged exclusively in order to produce high-quality (i.e., ground truth) measurement data/traces, which are subsequently fed to the detector for evaluation. The research community will obviously need to play a central role, both by designing and sharing simulators for specific domains and purposes, and in sharing the traces that are produced [14, 10].

3.2 Hypothetical Evaluation

Admittedly, simulation is not the cure-all answer to the problem of reliably and completely evaluating network anomaly detectors—it is necessary but not sufficient. Simulation provides a method to test an algorithm, but not to truly validate it. For instance, a simulation may show that a certain type of anomaly can be detected with 0 false positives or negatives. However, if that type of anomaly never occurs in real data, the algorithm cannot be considered to be useful.

We propose the following framework:

1. *Problem formulation.* We start by formulating the scenario we wish to consider (e.g., detecting traffic spikes in backbone traffic). We need an idea of the characteristics of both normal and anomalous traffic in order to (i) simulate these and (ii) choose an appropriate anomaly detection scheme.

2. *Training.* Simulation is used to train the anomaly detection algorithms to be tested or compared. Optimizing an algorithm's parameters requires large data sets. Simulation can provide this data.

3. *Test.* Simulations (independent of the first set) are used to test the algorithms, and understand their advantages and limitations in detail. Simulations satisfy our requirements for ground truth (as we specify this), repeatability, definition and control.

4. *Validation.* Once the characteristics of an algorithm are understood, it can be applied to real data to understand how it performs in a realistic setting (e.g., while processing vast amounts of data or subject to network operator requirements) [10]

This is not a closed framework. Measurements inform our problem formulation and our simulation models. Without such an iterative approach, anomalies pollute the data, which both prevents the construction of accurate tests and can profoundly degrade the performance of detectors that build their models of "normal" network operation based on historical data that is presumed to be clean [11].

4. CONCLUSION

The position we take in this paper is that anomaly detection research should improve its standard of rigor in future publications. We have presented four central requirements of an evaluation methodology that can only be met by simulating both the anomalies themselves and the background traffic. While simulation is a necessary evaluation technique, it is not sufficient. We have therefore presented an outline of a complete evaluation methodology that specifies the role that both simulation and traces from operational networks

has to play. We believe the details of how this simulation should be performed in specific contexts is an important area of future research.

5. REFERENCES

[1] BARFORD, P., KLINE, J., PLONKA, D., AND RON, A. A signal analysis of network traffic anomalies. In *ACM Internet Measurement Workshop* (Marseille, France, 2002).

[2] ESTAN, C., SAVAGE, S., AND VARGHESE, G. Automatically inferring patterns of resource consumption in network traffic. In *ACM SIGCOMM* (Karlsruhe, Germany, 2003), pp. 137–148.

[3] FLOYD, S., AND PAXSON, V. Difficulties in simulating the Internet. *IEEE/ACM Transactions on Networking 9*, 4 (2001), 392–403.

[4] HANLEY, J. A., AND MCNEIL, B. J. The meaning and use of the area under a receiver operating characteristic (ROC) curve. *Radiology 143*, 1 (1982), 29–36.

[5] HUANG, Y., FEAMSTER, N., LAKHINA, A., AND XU, J. J. Diagnosing network disruptions with network-wide analysis. In *ACM SIGMETRICS* (San Diego, CA, USA, 2007).

[6] KOMPELLA, R. R., SINGH, S., AND VARGHESE, G. On scalable attack detection in the network. In *ACM Internet Measurement Conference* (New York, NY, USA, 2004), pp. 187–200.

[7] KOTZ, D., AND HENDERSON, T. Crawdad: A community resource for archiving wireless data at Dartmouth. *IEEE Pervasive Computing 04*, 4 (2005), 12–14.

[8] LAKHINA, A., CROVELLA, M., AND DIOT, C. Diagnosing network-wide traffic anomalies. In *ACM SIGCOMM* (Portland, Oregon, USA, 2004), pp. 219–230.

[9] LELAND, W. E., TAQQU, M. S., WILLINGER, W., AND WILSON, D. V. On the self-similar nature of Ethernet traffic (extended version). *IEEE/ACM Transactions on Networking 2*, 1 (Feb 1994), 1–15.

[10] RINGBERG, H., SOULE, A., AND REXFORD, J. WebClass: Adding rigor to manual labeling of traffic anomalies. *SIGCOMM Comput. Commun. Rev.* (2008).

[11] RINGBERG, H., SOULE, A., REXFORD, J., AND DIOT, C. Sensitivity of PCA for traffic anomaly detection. In *ACM SIGMETRICS* (San Diego, CA, USA, 2007).

[12] ROUGHAN, M., GREENBERG, A., KALMANEK, C., RUMSEWICZ, M., YATES, J., AND ZHANG, Y. Experience in measuring Internet backbone traffic variability: Models, metrics, measurements and meaning. In *Proceedings of the International Teletraffic Congress (ITC-18)* (Berlin, Germany, 2003), pp. 379–388.

[13] SEKAR, V., DUFFIELD, N. G., SPATSCHECK, O., VAN DER MERWE, J. E., AND ZHANG, H. LADS: Large-scale automated DDoS detection system. In *USENIX Technical Conference* (2006), pp. 171–184.

[14] SHANNON, C., MOORE, D., KEYS, K., FOMENKOV, M., HUFFAKER, B., AND K CLAFFY. The Internet measurement data catalog. *SIGCOMM Comput. Commun. Rev. 35*, 5 (2005), 97–100.

[15] SOULE, A., SALAMATIAN, K., AND TAFT, N. Combining filtering and statistical methods for anomaly detection. In *ACM Internet Measurement Conference* (Berkeley, California, USA, October 2005).

[16] ZHANG, Y., GE, Z., GREENBERG, A., AND ROUGHAN, M. Network anomography. In *ACM Internet Measurement Conference* (Berkeley, California, USA, October 2005).

[17] ZHANG, Y., SINGH, S., SEN, S., DUFFIELD, N., AND LUND, C. Online identification of hierarchical heavy hitters: Algorithms, evaluation, and applications. In *ACM Internet Measurement Conference* (Taormina, Sicily, Italy, 2004), pp. 101–114.

[18] ZOU, C. C., GAO, L., GONG, W., AND TOWSLEY, D. Monitoring and early warning for Internet worms. In *Conference on Computer and Communications Security* (New York, NY, USA, 2003), pp. 190–199.