Journal of Network and Systems Management, Vol. 14, No. 1, March 2006 (© 2006)
DOI: 10.1007/s10922-005-9016-3

Detecting Distributed Denial-of-Service Attacks Using Kolmogorov Complexity Metrics[1]

Amit Kulkarni[2] and Stephen Bush[2]

Published online: 29 April 2006

Abstract This paper describes an approach to detecting distributed denial of service (DDoS) attacks that is based on fundamentals of Information Theory, specifically Kolmogorov Complexity. A theorem derived using principles of Kolmogorov Complexity states that the joint complexity measure of random strings is lower than the sum of the complexities of the individual strings when the strings exhibit some correlation. Furthermore, the joint complexity measure varies inversely with the amount of correlation. We propose a distributed active network-based algorithm that exploits this property to correlate arbitrary traffic flows in the network to detect possible denial-of-service attacks. One of the strengths of this algorithm is that it does not require special filtering rules and hence it can be used to detect any type of DDoS attack. We implement and investigate the performance of the algorithm in an active network. Our results show that DDoS attacks can be detected in a manner that is not sensitive to legitimate background traffic.

KEY WORDS: Kolmogorov Complexity; denial-of-service attack; active network; entropy; complexity probes.

1. INTRODUCTION

Distributed denial-of-service (DDoS) attacks are caused by an attacker flooding the target machine with a torrent of packets originating from a number of machines under the attacker's control. These machines are called "zombies." Tools that control and launch attacks from these zombies against the target

[1]This research has been funded by the Defense Advanced Research Projects Agency (DARPA) contract F30602-01-C-0182 and managed by the Air Force Research Laboratory (AFRL) Information Directorate.

[2] To whom correspondence should be addressed at Rensselaer Polytechnic Institute (RPI), New York,; Email: To whom correspondence should be addressed at Rensselaer Polytechnic Institute (RPI), New York. E-mail: bushsf@research.ge.com; Home Page: http://www.research.ge.com/∼bushsf.

[3] General Electric Global Research Center, Niskayuna, New York.

perform the attacks. The attacks can cause networks to be disabled for extended periods of time during which customers, employees, and business partners, are unable to access information or perform transactions. Several different tools for performing distributed denial of service attacks have been identified including Trinoo, Tribe Flood Network 2K (TFN2K), and StachelDraht [1]. The tools use a master system to control damons installed on other compromised systems to launch distributed parallel attacks against a targeted system. The types of attack can vary, but up to this point have consisted of denial of service attacks such as smurf, synflood, icmpflood, and udpflood. Furthermore, new DDoS attack tools use encryption to make detection and tracking of the attacker even harder.

Detection techniques are classified into anomaly-based detection and misuse-based detection. Anomaly-based detection looks for deviation from the pattern of normal traffic using techniques such as statistical measures. In contrast to anomaly-based detection, misuse-based detection maintains a database of signatures; suspicious flows are matched for their signature against this database. Typical detection techniques [2, 3] rely on filtering based on packet type and rate. Essentially, the detection software attempts to correlate the type of packet used for the attack, be it ICMP or UDP, with the destination. While these techniques have reasonable success, they have several limitations. Firstly, they are not very flexible because they are typically customized for known attack patterns. Therefore, these techniques are likely to fail if a new type of packet is used or if the attack consists of a traffic pattern that is a combination of different types of packets, e.g., ICMP and UDP. In such cases, packet profiling is defeated. Secondly, a large number of detection techniques use traffic logs to identify attacks. However, traffic logs generate a large amount of data even during normal operation so it is difficult and time-consuming to scan traffic logs looking for patterns when the network is under attack.

This paper describes an anomaly-based detection approach that leverages fundamentals of information complexity to provide a flexible and effective detection method. Stated as simply and succinctly as possible, we hypothesize that information, comprising observations of actions with a single root cause, whether they are faults or attacks, is highly correlated. Highly correlated data has a high compression ratio. The remainder of the paper is organized as follows. Section 2 describes the concept of information complexity, specifically Kolmogorov Complexity, as a measure of detecting and analyzing vulnerabilities in a system or network. Section 3 describes a specific approach for detecting distributed denial of service attacks in a network. Section 4 presents an experimental setup and results of using our method for DDoS attack detection. Section 5 describes other research and commercial efforts in the area. The paper ends with a summary of our work and future directions in Section 6 and acknowledgments and references in Sections 7 and 8.

2. BACKGROUND

The Kolmogorov Complexity of a string x, written as $K(x)$, gives the size of the smallest program written for a universal Turing Machine that is capable of generating x [4]. It measures the degree of randomness for the given data. In the case of a completely random string, the length of the shortest program that can generate it is equal to the size of the string itself. For all other cases, it is strictly smaller than the size of the string; the program size becomes smaller as more regularity or pattern is discernible from the string. A side effect of this complexity measure is its ability to represent the correlation between disparate pieces of data. This side effect is exploited to design an effective method for detecting DDoS attacks. This paper examines the behavior of complexity and suggests a complexity-based information assurance technique that identifies denial-of-service attacks as new vulnerabilities in an information system. The benefits of such a technique are its ubiquitous application; the technique analyzes fundamental properties of information without requiring *a priori* attack information such as an ontology or database of known attack mechanisms.

Vulnerabilities and attacks in a dynamic system are demonstrated by means of a complexity grid, which we call a K-Map, in which low complexity paths are identified. The concept of a K-Map is shown in Fig. 1. The overall complexity of the system is illustrated by the surface contour of the map and is comprised of the complexity of the information itself and the complexity of the system in which the information resides. The premise is that when the network is in a healthy state, its behavior is highly complex given many simultaneous, independent users, which appear as areas of high complexity in the K-Map. Specific vulnerabilities and attacks will appear as areas of low complexity. Specifically, attacks such as

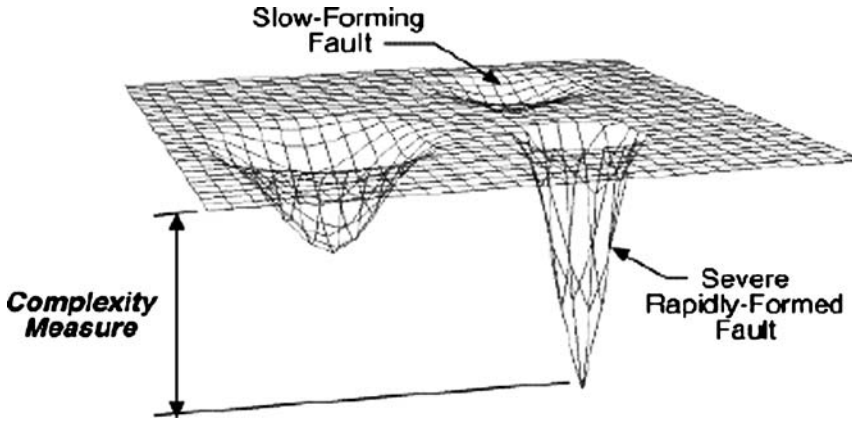

Fig. 1. Conceptual System K-Map.

Distributed Denial of Service (DDoS) will appear as warps in the complexity grid. This is due to the inherent system correlation in DDoS attack-streams.

3. APPROACH

The DDoS attack detection algorithm makes use of a fundamental theorem of Kolmogorov Complexity that states: for any two random strings X and Y,

$$K(XY) \leq K(X) + K(Y) + c, \tag{1}$$

where $K(X)$ and $K(Y)$ are the complexities of the respective strings, c is a constant and $K(XY)$ is the joint complexity of the concatenation of the strings. Proof for the above theorem is described in [3]. Simply put, the joint Kolmogorov Complexity of two strings is less than or equal to the sum of the complexities of the individual strings. The equivalence holds when the two strings X and Y are totally random, i.e., they are completely unrelated to each other. Another effect of this relationship is that the joint complexity of the strings decreases as the correlation between the strings increases. Intuitively, if two strings are related, they share common characteristics and thus common patterns. That knowledge can be harnessed to generate a smaller program that can represent the combined string.

In terms of detection of DDoS attacks, the property given by Inequality (1) is exploited to distinguish between concerted denial-of-service attacks and cases of traffic overload. The assumption is that an attacker performs an attack using large numbers of similar packets (in terms of their type, destination address, execution pattern, timing, etc.) sourced from different locations but intended for the same destination. Thus, there is a high degree of similarity in the traffic pattern. A Kolmogorov Complexity based detection algorithm can quickly identify such a pattern. On the other hand, a case of legitimate traffic overload in the network tends to have many different traffic types. The traffic flows are not highly correlated and appear to be random. Therefore, our algorithm samples every distinct flow of packets (distinguished by their source and destination addresses) to determine if there is a large amount of correlation between the packets in a flow. If it is determined to be so, then all suspicious flows at the node are again correlated with each other to determine that it is indeed an attack and not a case of a traffic overload.

3.1. Architecture

The architecture for DDoS detection has been implemented in an active network for ease of deployment and flexibility in testing. As shown in Fig. 2, packet complexity probes (described in detail in the next section), associated with

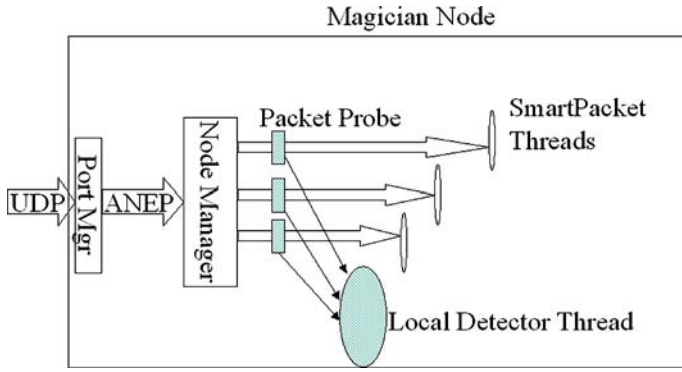

Fig. 2. Locations of complexity probes in an active node.

every traffic flow through a node, periodically sample packets in the flow. For the collected samples, the probe calculates a complexity differential over the samples.

Complexity differential is defined as the difference between the cumulative complexities of individual packets and the total complexity computed when those packets are concatenated to form a single packet. If packets x_1, x_2, $x_3 \ldots x_n$ have complexities $K(x_1)$, $K(x_2)$, $K(x_3)$,..., $K(x_n)$, then the complexity differential is computed as:

$$[K(x_1) + K(x_2) + K(x_3) + \ldots + K(x_n)] - K(x_1 x_2 x_3 \ldots x_n), \tag{2}$$

where $K(x_1 x_2 x_3 \ldots x_n)$ is the complexity of the packets concatenated together as measured in a finite time interval window (shown in Fig. 3). If packets x_1, x_2, x_3,..., x_n are completely random, $K(x_1 x_2 x_3 \ldots x_n)$ will be equal to the sum of the individual complexities and the complexity differential will therefore be zero. However, if the packets are highly correlated, i.e., some pattern emerges in their concatenation, then the concatenated packet can be represented by a smaller program and hence its complexity, i.e., $K(x_1 x_2 x_3 \ldots x_n)$ will be smaller than the cumulative complexity.

In effect, we use the measure of the compressibility of the packets accumulated in a given time interval to determine correlation. If the complexity differential is greater than a preset threshold for the flow, the flow is marked as suspect and the collected sample is referred to a Local Detector running on the node.

The Local Detector receives all such samples from various suspicious flows and correlates all the samples together using the same complexity differential calculation. If there is only one suspect flow, no correlation is performed. If the complexity differential again exceeds the threshold, all suspect flows (including the case of a single flow) are referred to as a Domain Detector that is running on some other node on the local network domain. The Domain Detector, in turn, correlates

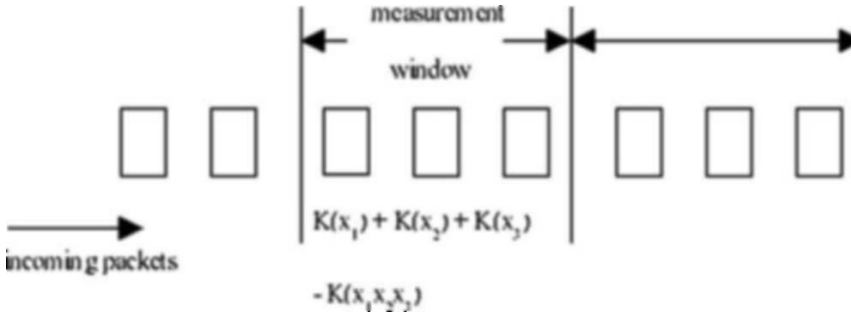

Fig. 3. Complexity differential measurement.

flows received from the Local Detectors to determine if the attack is localized or distributed using the same technique as described for the Local Detectors. Thus, a hierarchy of detectors cooperates to detect distributed denial of service attacks in the network itself. This hierarchy is shown in Fig. 4.

3.2. Estimation of Complexity

While it is known that, in general, Kolmogorov Complexity is not computable, various methods exist to compute estimates of the complexity. The packet complexity probe described in the previous section uses an entropy calculation technique for estimation of complexity. The estimator, currently implemented as

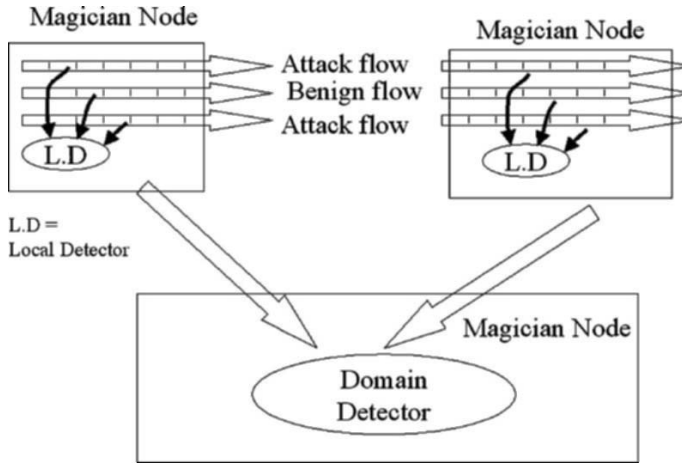

Fig. 4. DDoS detection architecture for a node.

a simple compression estimation method, returns an estimate of the smallest compressed size of a string. The complexity *K(x)* is computed using the entropy *H(p)* of the weight of ones ("1") in a string. Specifically, $K(x)$ is defined in Eq. (3) where *x#1* is the number of "1" bits and *x#0* is the number of "0" bits in the string whose complexity is to be determined. Entropy *H(p)* is calculated using Eq. (4). The expected complexity is asymptotically related to entropy as shown in Eq. (5). See [4] for other measures of empirical entropy and their relationship to Kolmogorov Complexity.

$$\widehat{K}(x) \approx l(x)H\left(\frac{x\#1}{x\#1 + x\#0}\right) + \log_2(l(x)) \tag{3}$$

$$H(p) = -p\log_2 p - (1-p)\log_2(1-p) \tag{4}$$

$$H(X) = \sum_{l(x)=n} P(X = x)C(X) \tag{5}$$

The complexity estimation technique used here is not the best because empirical entropy is actually a very poor method of complexity estimation. For example, the estimate for the string 10101010101010101010 and a completely random string with equal numbers of 1's and 0's is the same under empirical entropy. While it is true that one case does not prove anything, it illustrates the point that empirical entropy does not account for any regular patterns that may occur in the data. However, this simple case illustrates that improvement can be made. More accurate estimates for complexity will only serve to improve our method for DDoS detection. Future work will incorporate improved estimation techniques for the complexity probe and the performance of the algorithm will be compared with respect to the two techniques.

4. EXPERIMENTAL RESULTS

We compared our technique to a prototype packet counting algorithm for DDoS detection and found that our technique is better at discriminating traffic patterns. We use an active network testbed to embed the probes and perform our experiments. Active networking [5] is a new paradigm of networking in which code is sent along with the data in the packets that traverse the network. This code can execute on the nodes of the network enabling new functionality and services to be implanted in the network. Magician [6] is an active network emulation environment written in Java that can run on any workstation and create an overlay network that acts as an active network. We used our Magician-based active network testbed for the experiment. The programmable testbed makes it quite easy to set

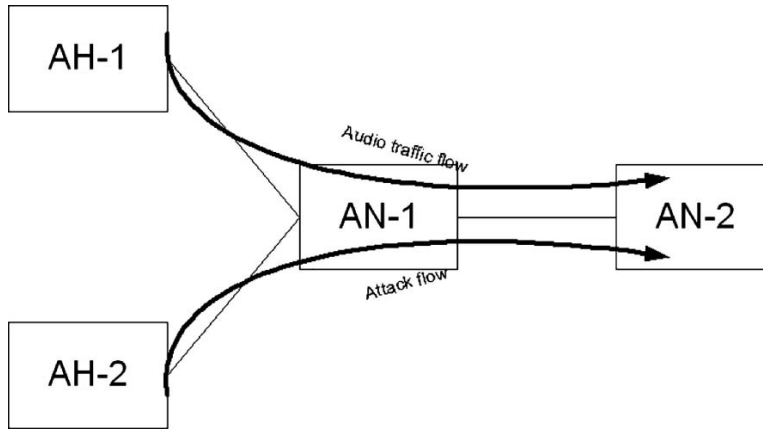

Fig. 5. Topology for experiment.

up a desired topology for the network, as well as control and measure performance using an active network. The Kolmogorov complexity probes, which are written in Java, are embedded inside the Java-based Magician kernel.

The experimental setup consisted of a set of active nodes arranged in the topology shown in Fig. 5. Node AH-1 continuously generates traffic consisting of audio packets destined for node AN-2. The load induced by this traffic is high enough that it is registered at node AN-1 as a "suspicious" flow, i.e., a traffic flow whose complexity differential exceeds the threshold. The load induced by this traffic flow is kept constant throughout the experiment. Node AH-2 generates the attack flow. The load induced by the attack flow is varied to determine the performance of the algorithms. The experiment is run twice, once with only the attack source on (node AH-2 transmitting only) and the next time with both sources on (both node AH-1 and node AH-2 transmitting). The rationale is that an attack is essentially a sustained overload induced for some time interval. The purpose of the experiment is to determine the effectiveness of the two techniques in separating and identifying an attack in the presence of background traffic.

Figures 6 and 7 show the performance of the packet-counting and complexity-based approaches as measured against the load induced by the two sources (in packets per second) described above. Figure 6 shows that the packet-counting metric cannot discriminate between an attack and a true overload. When the audio source is transmitting in conjunction with the attack source, any threshold set by the packet-counting algorithm running on node AN-1 will be exceeded leading to the false conclusion that the node is under attack. For example, based on the attack pattern only (dashed curve), we decide to set the threshold at

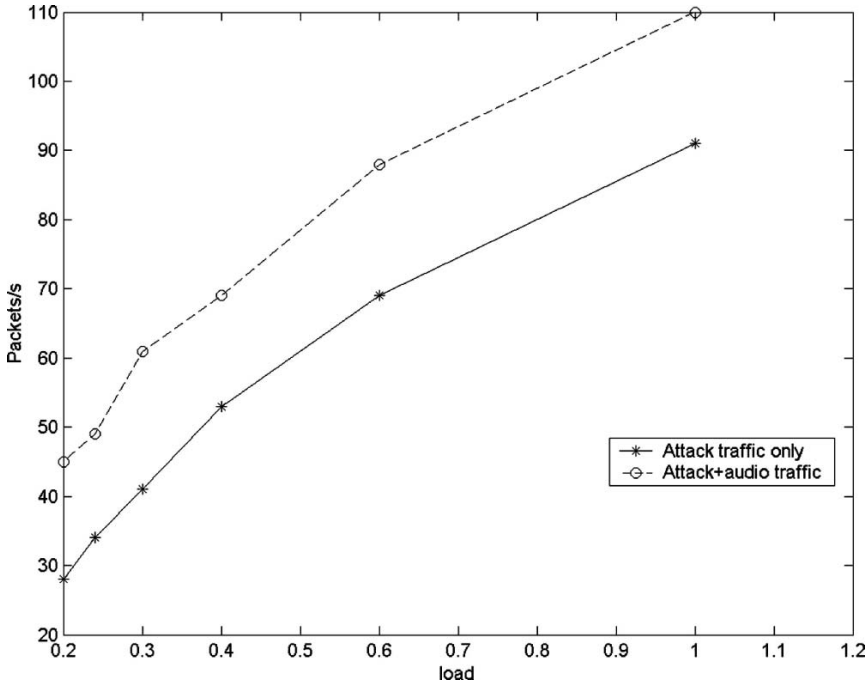

Fig. 6. Performance of packet counting metric.

70 packets/s for a load of 0.6. When the audio source is introduced, the combined traffic trips the same threshold at a load of only 0.4, which is a false positive.

Figure 7 shows the complexity differential versus load curve for a given sampled time interval, which in this case was 10 s. Note that *higher differential* complexity corresponds to *reduced* complexity of the flow. In effect, the higher differential complexity estimates the deviation from the randomness inherent in a healthy network with a mix of different traffic flows.

Thus, the experiment shows that the attack flow is estimated to be less complex over time than the ambient legitimate traffic. Also it is to be noted that the complexity-based metric does not change its behavior when a combination of attack and traffic sources is used. This is because the attack traffic dominates the combined flow and hence the complexity differential is roughly equal to that observed if only the attack flow existed. Therefore, the complexity-based approach is more accurate in separating false alarms from true attacks because it can conserve salient patterns of a traffic flow.

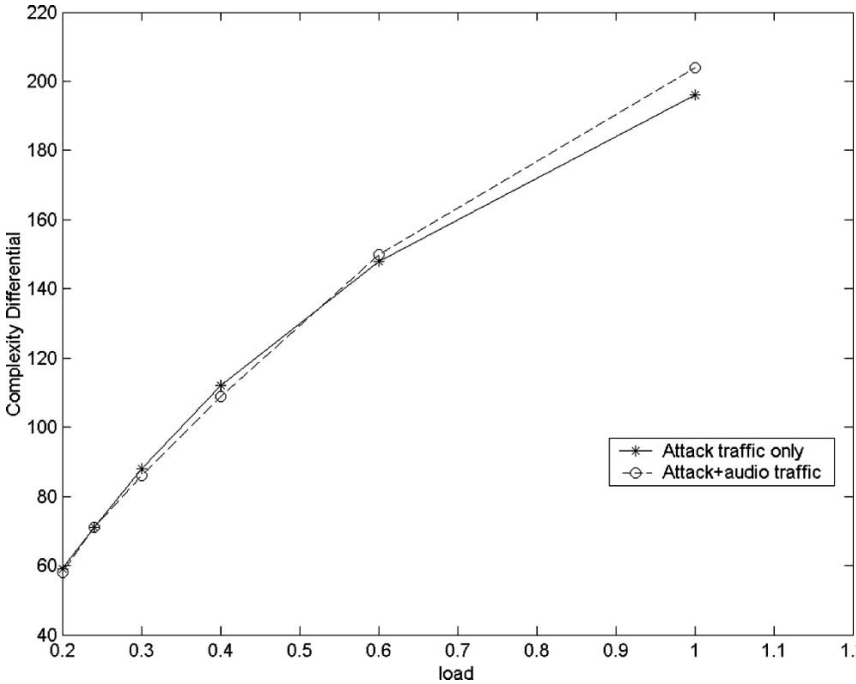

Fig. 7. Performance of complexity-based metric.

5. RELATED WORK

The route-based distributed packet filtering (DPF) [7] exploits the power-law property of Internet autonomous-systems (AS) connectivity to filter out significant portion of the spoofed packet flows to prevent attack packets from reaching their target. Route-based DPF's main strength lies in the fact that with partial coverage or deployment—about 18% in Internet AS topologies—a synergistic filtering effect is achieved whose collective filtering action proactively prevents spoofed IP flows from reaching other autonomous systems in the first place. However, the scheme is not effective when the attack is distributed with the attacking zombie machines being part of multiple AS'es.

In [8], the authors propose an anomaly-based detection technique in which an abnormal increase in the number of flows or packets within a time window that is not followed by a similar increase in the number of octets is considered to be strong evidence for an attack. The idea is that under normal circumstances the time derivatives of the number of flows, packets, and octets coming through a router over a short time interval (window) have the same sign; this practically

means that when one of them increases or decreases so do the others. However, the authors concede that this may not apply to short- term observations.

The anomaly-based detection technique proposed in [3] employs statistical analysis of data obtained from multiple layers of the protocol stack for detection of very subtle changes in traffic, which is considered typical of attacks. The idea of the approach is based on the observation that an attack leads to relatively abrupt changes in statistical models of traffic compared to the "normal mode." These changes occur at unknown points in time and need to be detected instantly. Therefore, the problem of detecting an attack is formulated and solved as a change-point detection problem: detect a change in the distribution (model) with a fixed delay (batch approach) or minimal average delay (sequential approach), controlling the rate of false detections. In addition, the authors combine both methods (batch and sequential) in one unit to develop a multistage (batch sequential) detection algorithm, which turns out to be more robust and reliable to some expense of the detection delay. The authors claim that this method can be used for detection of generalized denial-of-service attacks.

Commercial tools such as FloodGuard [9] from Reactive Solutions are designed to stop so-called SynFlood attacks and other DDoS techniques in which the IP addresses of the attacking machines are spoofed. FloodGuard comprises two components: detectors and actuators. The detectors sit downstream of the network bottleneck, viewing an attack as it happens. Once the software identifies an incoming DDoS attack using traffic analysis searching for abnormal activity, the detectors send a message to the actuators, which, in turn, communicate with the network routers and switches and instruct them to install filters to choke off the attack traffic. The software can install the filters automatically or send an alert to the administrator warning of the incoming attack and suggesting a remedy. And, unlike other anti-DDoS solutions, FloodGuard does not rely on a database of known attack signatures. Because packets sent during DDoS attacks can be encrypted or given a junk payload that throws off signature-based detection, Reactive's software looks for unusual traffic patterns and packet behavior in relation to their protocols to identify attacks.

6. SUMMARY AND FUTURE WORK

This paper describes an attempt at bridging the gap between the promise of the theory of information complexity, particularly Kolmogorov Complexity, and the application of the theory to hard problems. We embarked on the effort to help us gain a deeper understanding of the strengths, weaknesses, and challenges of the measurement, representation, and calculation of Kolmogorov Complexity estimates, using the DDoS attack detection problem as a model application. Obviously, good estimation of Kolmogorov Complexity is key to its usefulness in identifying correlation between attack flows. Although our simple entropy

calculation technique served as a useful metric and it is computationally efficient, we are investigating and benchmarking more estimators for the Kolmogorov Complexity of a given string.

With respect to the DDoS detection technique, its performance needs to be compared to more intelligent detection algorithms that are currently in use. In particular, its performance has to be measured in terms of resource tradeoffs, detection, and false-alarm probability and response time. For example, the current technique performs its evaluation on the entire contents of the packet. Anecdotal evidence has shown that the performance degrades if the payload of the packet is encrypted and the size of the payload dominates the size of the packet. Techniques that adapt to payload size are being formulated and tested.

In terms of next steps, the next challenge is identifying or developing theory using Kolmogorov Complexity for controlling the DDoS attacks and tracing the attack back to the attacker. The fundamental hypothesis is that the attacker can be traced back using a complexity-based approach because the attacks must have a common pattern since they originated from a common source.

7. ACKNOWLEDGMENTS

The work discussed in this paper was funded by DARPA, under the auspices of the Fault Tolerant Networks program. Our thanks go to Doug Maughan, the Manager for the Fault-Tolerant Networks program and Scott Shyne, Air Force Rome Labs for their generous support.

REFERENCES

1.. J Mirkovic, S. Dietrich, D. Dittrich, and P. Reiher. *Internet Denial of Service: Attack and Defense Mechanisms*, Prentice-Hall , ISBN 0131475738.

2.. T. Gil and M. Poletto, *MULTOPS: A Data Structure for Bandwidth Attack Detection*, USENIX 2001.

3.. R. Bezeq, H. Kim, B. Rozovskii, and A. Tartakovsky, *A Novel Approach to Detection of Denial-of-Service Attacks via Adaptive Sequential and Batch-Sequential Change-Point Methods*, IEEE Systems, Man and Cybernetics Information Assurance Workshop, June 2001.

4.. M. Li and P. Vitanyi, *An Introduction to Kolmogorov Complexity and its Applications*, Springer-Verlag, Berlin, 1997.

5.. D. L. Tennenhouse, J. M. Smith, W. D. Sincoskie, D. J. Wetherall, and G. J. Minden, A survey of active network research, *IEEE Communications Magazine*, Vol. 35, No. 1, pp. 80–86, January 1997.

6.. A. Kulkarni, G. Minden, R. Hill, Y. Wijata, S. Sheth, H. Pindi, F. Wahhab, A. Gopinath, and A. Nagarajan, *Implementation of a Prototype Active Network*, IEEE OpenArch, San Francisco, 1998.

7.. K. Park and H. Lee, *On the Effectiveness of Route-Based Packet Filtering for Distributed DoS Attack Prevention in Power-Law Internets*, Proceedings of ACM SIGCOMM ‘01, 2001.

8.. C. Kotsokalis, D. Kalogeras, and B. Maglaris, *Router-Based Detection of DoS and DDoS Attacks*, Eighth Workshop of the HP OpenView University Association, Berlin, Germany, June 2001.

9.. *FloodGuard: A Distributed Solution for Detecting and Mitigating Flooding Attacks Leading to Denial of Service*, White Paper, Reactive Solutions.