GE Research &
Development Center

Detecting Distributed Denial-of-Service Attacks Using Kolmogorov Complexity Metrics

A.B. Kulkarni, S.F. Bush, and S.C. Evans

2001CRD176, December 2001
Class 1

Technical Information Series

Corporate Research and Development

Technical Report Abstract Page

Title	Detecting Distributed Denial-of-Service Attacks using Kolmogorov Complexity Metrics		
Author(s)	A.B. Kulkarni S.F. Bush S.C. Evans	**Phone**	(518)387-4291 8*833-4291
Component	Information and Decision Technologies		
Report Number	2001CRD176	**Date**	December 2001
Number of Page	8	**Class**	1
Key Words	Kolmogorov complexity, denial-of-service, active network, entropy		

This paper describes an approach to detecting distributed denial of service (DDoS) attacks that is based on fundamentals of information theory, specifically Kolmogorov complexity. The algorithm is based on a concept of Kolmogorov complexity that states that the joint complexity measure of random strings is lower than the sum of the complexities of the individual strings if the strings exhibit some correlation. Furthermore, the joint complexity measure varies inversely with the amount of correlation. The proposed algorithm exploits this feature to correlate traffic flows in the network and 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. This algorithm is shown to perform better than simple packet-counting or load-measuring approaches.

Manuscript received November 12, 2001

Detecting Distributed Denial-of-Service Attacks Using Kolmogorov Complexity Metrics

A.B. Kulkarni, S.F. Bush and S.C. Evans

Introduction

Distributed denial-of-service attacks are caused by the 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'. The attacker typically uses ICMP or UDP packets for the attack. Typical detection techniques [1,2] for these types of attacks 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 are not very flexible. For example, these techniques will fail if a new type of packet is used for attack or if the attack consists of a traffic pattern that is a combination of ICMP and UDP packets. In such cases, packet profiling is defeated. This paper describes an approach based on fundamentals of information complexity that is both flexible and effective.

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 Kolmogorov Complexity, K(x), of a string of data measures the size of the smallest program capable of representing the given piece of data [3]. It measures the degree of randomness for the given data. The length of the shortest program to generate a completely random string is equal to the size of the string itself. For all other cases, it is smaller than the size of the string and the program size becomes smaller as more regularity or pattern is discernible from the string. A side effect of this 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.

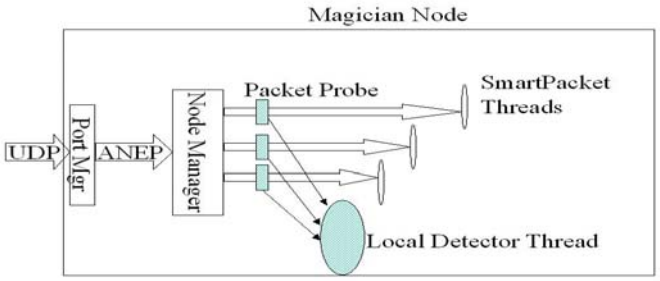

Figure 1. Implementation in Magician Active Node

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.

The concept of "Conservation of Complexity" was introduced in [6] and explored in detail using simulation [8]. This concept relates to the ability to discern an attack by monitoring the complexity change due to processes occurring in the system and imposing bounds that identify unauthorized processes – noted by complexity changes that are either too great or too small to be from authorized processes. Figure 2 below describes the concept of conservation of complexity. This concept was first applied to a closed system, where the processes are known and able to be monitored by complexity probes. In the distributed case of a denial of service attack, the process is not known, but bounds on the differential complexity allowed by the distributed processes are still able to be enforced.

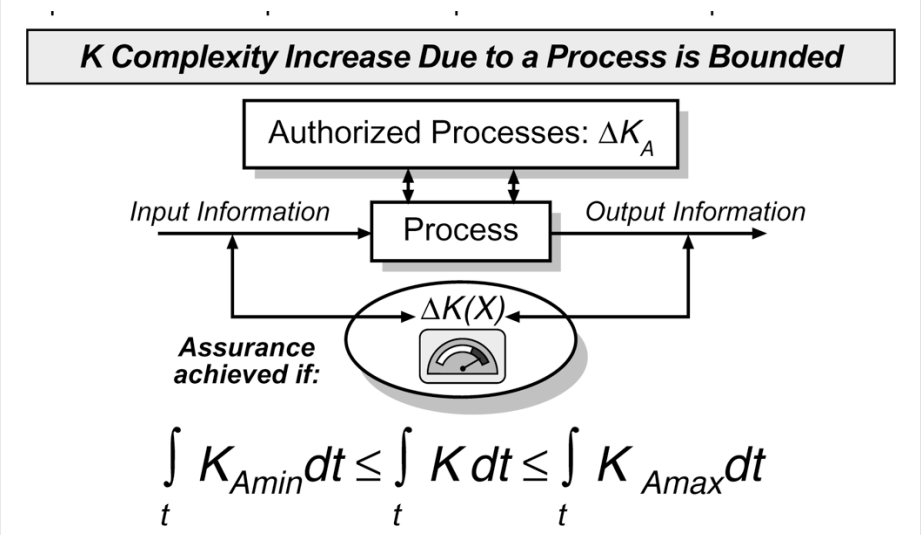

Figure 2. Principle of Conservation of Complexity

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 etc.) sourced from different locations but intended for the same destination. Thus, there is a lot 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.

The architecture for DDoS detection has been implemented in an active network for ease of deployment and flexibility in testing. As shown in Figure 1, packet complexity probes (described in detail in the next section), associated with 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,…,x_n have complexities $K(x_1)$, $K(x_2)$, $K(x_3)$,…, $K(x_n)$, then complexity differential is computed as:

$$[K(x_1) + K(x_2) + K(x_3) +\ldots+ K(x_n)] - K(x_1x_2x_3\ldots x_n), \qquad 2)$$

where $K(x_1x_2x_3\ldots x_n)$ is the complexity of the packets concatenated together. If packets x_1, x_2, x_3,…,x_n are completely random, $K(x_1x_2x_3\ldots x_n)$ 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. $K(x_1x_2x_3\ldots x_n)$ 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 a Domain Detector that is running on some other node on the local network domain. The Domain Detector in turn correlates 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 Figure 3.

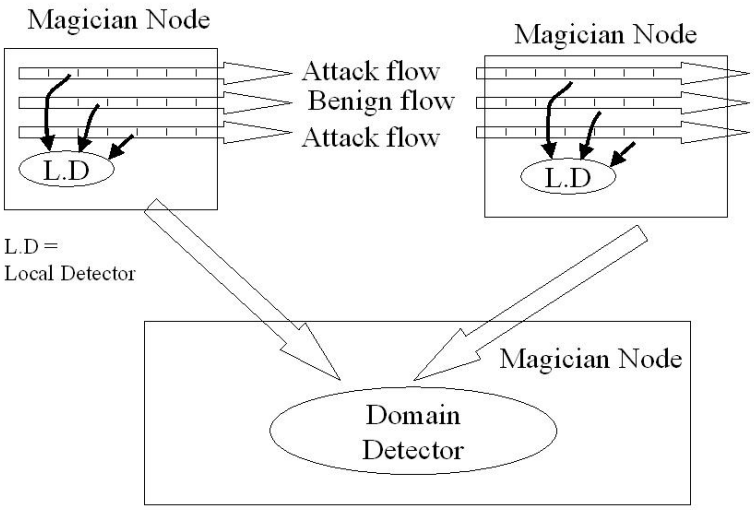

Figure 3. DDoS detection Architecture

Complexity Estimates

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 Kolmogorov Complexity estimator, currently implemented as 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 in a string. Specifically, *K(x)* is defined in Equation 4 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 defined in Equation 5. The expected complexity is asymptotically related to entropy as shown in Equation 6. See [3] for other measures of empirical entropy and their relationship to Kolmogorov complexity.

$$\hat{K}(x) \approx l(x)H(\frac{x\#1}{x\#1 + x\#0}) + \log_2(l(x)) \tag{4}$$

$$H(p) = -p\log_2 p - (1-p)\log_2(1-p). \tag{5}$$

$$H(X) = \sum_{l(x)=n} P(X = x)C(X) \tag{6}$$

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 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. See [7] for an innovative and improved method for complexity measurement. The

improvement of the technique described in [7] over current methods is part of ongoing research. In future work, this technique will be used in the complexity probe and the performance of the algorithm will be compared with respected to the two techniques.

Experimental Results

We compared our technique to a prototype packet counting algorithm for DDoS detection and found that our technique is better discriminates traffic patterns. We used our Magician-based [4] active network [5] testbed for the experiment for two reasons. Firstly, it is quite easy to set up a desired topology for the network, as well as control and measure performance using an active network. Secondly, it is easier to embed our complexity probes, which are written in Java, inside the Java-based Magician kernel as opposed to embedding them inside commercial routers. The results, however, can be extrapolated to real traffic settings.

The experimental setup consisted of a set of active nodes arranged in the topology shown in Figure 4. 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.

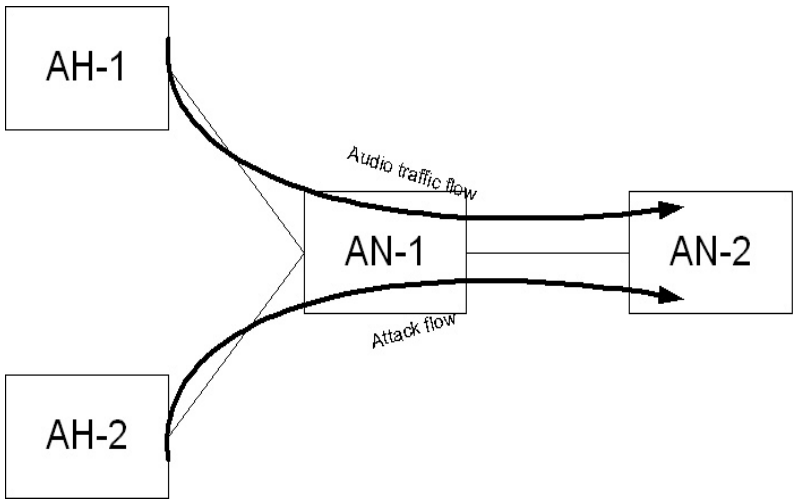

Figure 4. Topology for experiment

Figures 5 and 6 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 5 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 (blue curve), we decide to set the threshold at 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 6 below shows the complexity differential versus load curve for a given sampled time interval, which in this case was 10 seconds. 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 when 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.

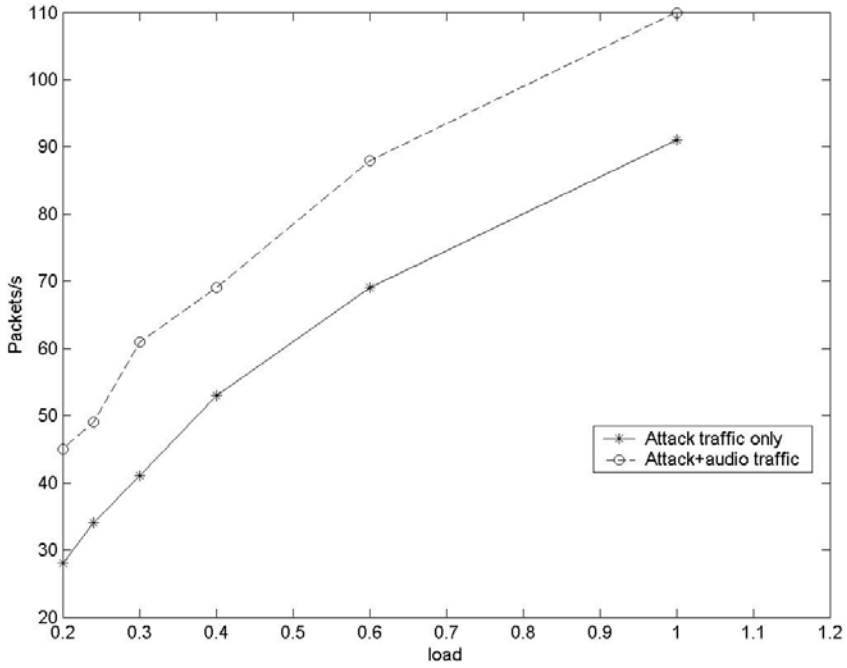

Figure 5. Performance of packet-counting metric

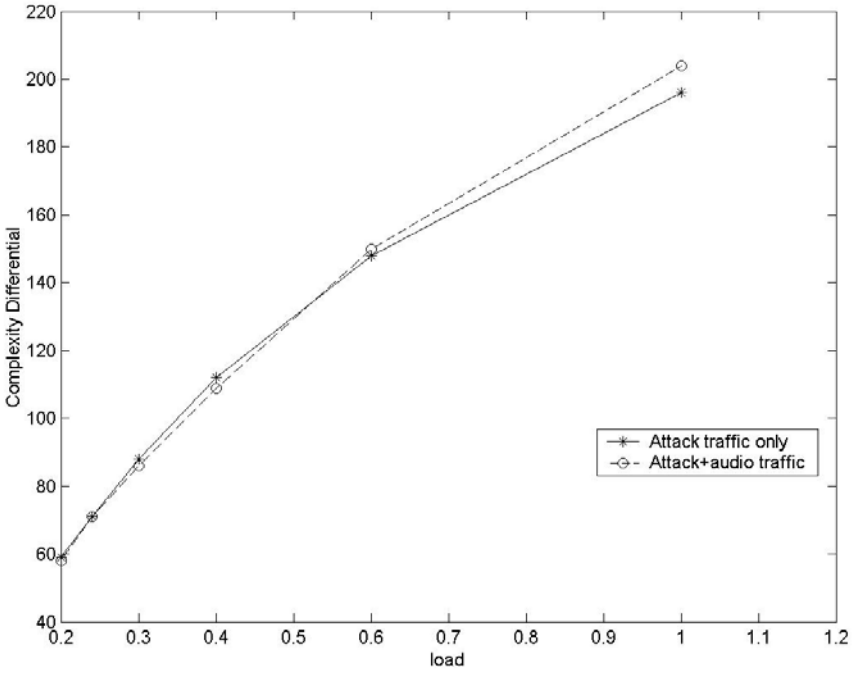

Figure 6. Performance of complexity-based metric

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 K(x).

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 content of the packet. Anecdotal evidence has shown that 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.

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] Gil, T. and Poletto, M. "MULTOPS: a data structure for bandwidth attack detection, " USENIX 2001.

[2] Bazek, R., Kim, H., Rozovskii, B., and Tartakovsky, A. "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.

[3] Li, M. and Vitanyi, P. *An Introduction to Kolmogorov Complexity and Its Applications*, Springer-Verlag, 1997.

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

[5] Bush, Stephen F. and Kulkarni, Amit B. *Active Networks and Active Virtual Network Management Prediction: A Proactive Management Framework*, ISBN 0-306-46560-4, Kluwer Academic/Plenum Publishers. Spring 2001.

[6] Evans, S. C., Bush, S. F., and Hershey, J., "Information Assurance through Kolmogorov Complexity", DARPA Information Survivability Conference & Exposition II, 2001, Proceedings Vol 2, pp 322-331.

[7] Evans, S. C. and Bush, S. F. "Symbol Compression Ratio for String Compression and Estimation of Kolmogorov Complexity", submitted to 2002 IEEE International Symposium on Information Theory, to be held June 30 – July 5, 2002.

[8] Bush, S. F. and Evans, S. C. "Complexity-based Information Assurance," General Electric Corporate Research and Development Technical Report 2001CRD084, October 2001.

A.B. Kulkarni
S.F. Bush
S.C. Evans

Detecting Distributed Denial-of-Service Attacks Using Kolmogorov Complexity Metrics

2001CRD176
December 2001