# DDoS Tolerant Networks*

Laura Feinstein, Dan Schnackenberg
*The Boeing Company, Phantom Works*
*Laura.C.Feinstein@boeing.com*
*Daniel.D.Schnackenberg@boeing.com*

Ravindra Balupari, Darrell Kindred
*Network Associates Laboratories*
*Ravindra_Balupari@nai.com*
*Darrell_Kindred@nai.com*

## Abstract

*The nature of the threats posed by Distributed Denial of Service (DDoS) attacks on large networks, such as the Internet, demands effective detection and response methods. These methods must be deployed not only at the edge but also at the core of the network. The DDoS Tolerant Networks technology incorporates methods to detect, characterize, and respond to DDoS attacks by computing entropy and frequency-sorted distributions of selected packet attributes. Preliminary results indicate that these methods can be effective against current attacks and suggest directions for improving detection of more stealthy attacks.*

## 1. Introduction

Powerful DDoS toolkits are now available to potential attackers, and essential networks are ill prepared for defense. Individual DDoS attacks are often too short-lived for manual diagnosis and response to be effective. To meet the increasing need for detection and response to DDoS attacks, researchers face these major issues:

- A stand-alone router on the attack path should automatically recognize that the network is under attack and adjust its traffic flow to ease the attack impact downstream.
- The defense techniques should be adaptable to a wide range of network environments, preferably without significant manual tuning.
- Attack detection should be as accurate as possible. False positives can lead to inappropriate responses that cause denial of service to legitimate users. False negatives result in attacks going unnoticed.
- Attack response should employ intelligent packet discard mechanisms to reduce the downstream impact of the flood while preserving and routing the non-attack packets.
- The detection method should be effective against a variety of attack tools available today and also robust against future attempts by attackers to evade detection.

These are demanding goals, but we contend that there are several reasons to believe that satisfactory detection and response methods can be designed. DDoS traffic generated by today's tools often has packet-crafting characteristics that make it possible to distinguish from normal traffic. For example, source addresses and destination TCP and UDP ports are often chosen at random to evade simple detection and filtering techniques, while normal traffic exhibits different distributions. Future DDoS tools may attempt to blend in better, but we claim that attack floods will still distort statistical measurements of the composition of the traffic at some points in the network. Our hypothesis is that relatively simple statistical measures can be used to discriminate DDoS traffic from legitimate traffic in core routers with sufficient accuracy to mitigate the effect of the attack downstream.

Our DDoS defense approach requires no explicit coordination (e.g., *pushback* [3]) between defending network components, no built-in knowledge of applications or protocols, and no instrumentation at end hosts. It can complement other approaches using these techniques in a comprehensive DDoS defense solution.

In the demonstration, a prototype implementation of these detection and response methods is deployed on routers within a test network. Background traffic obtained from a live network is injected on all network segments, to simulate a realistic detection environment. A DDoS flood is generated from attack agents on multiple hosts, with the intent of disrupting a network application. DDoS detectors identify the flood and apply automatic responses to squelch the flood traffic while permitting most legitimate traffic to pass. The network application is shown to survive the flood, which would have disabled it in the absence of DDoS defenses.

* This research was supported by DARPA under contract N66001-01-C-8048.

Proceedings of the DARPA Information Survivability Conference and Exposition (DISCEX'03)
0-7695-1897-4/03 $17.00 © 2003 IEEE

## 2. Detection Algorithms

Our detection algorithms measure statistical properties of specific fields in the packet headers at various points in the Internet. For instance, a detector on a router can monitor source IP addresses of arriving packets to build a model of the current distribution of source addresses. Through further computation with this distribution, it can measure the randomness or uniformity of the addresses and compare the current distribution to baseline observations. Significant deviations may indicate a DDoS attack in progress.

### 2.1. Entropy

Let an information source have $n$ independent symbols each with probability of choice $p_i$. Then, the entropy $H$ is defined as [6]:

$$H = -\sum_{i=1}^{n} p_i \log_2 p_i$$

Hence, entropy can be computed on a sample of consecutive packets. Comparing the value for entropy of some sample of packet header fields to that of another sample of packet header fields from the same observation point provides a mechanism for detecting changes in the randomness. We have observed through experimentation that while a network is not under attack, the entropy values for various header fields each fall in a narrow range. While the network is under attack with current attack tools, these entropy values exceed these ranges in a detectable manner.

A sophisticated attacker might attempt to defeat the detection algorithm by creating *stealthy* traffic floods that mimic the legitimate traffic the detector would expect. An attacker who knew that the entropy of various packet attributes was being monitored could build an attack tool that generates floods with tunable entropy levels. Through guesswork, penetration, or trial and error, the attacker could determine typical entropy levels seen at the detector and tune the flood to match. This may not be as easy as it sounds, particularly if there are multiple detectors deployed between the flood sources and the targets, as the typical entropy values seen by detectors in different network environments are likely to differ. Stealthy attacks are explored further in the full paper [1].

### 2.2. Chi-Square Statistic

Pearson's chi-square ($\chi^2$) Test is used to compare distributions. In the chi-square DDoS detector, the current distribution of values for some packet attribute (e.g., source address) is compared against a baseline measurement representing typical traffic seen in that detector's environment. When the chi-square statistic indicates a substantial discrepancy between the baseline and current distributions, the detector concludes that a DDoS attack may have begun.

The chi-square statistic is most useful in cases where the measurements involved have a small number of possible values, such as TCP SYN flag values (0 or 1) or protocol numbers. However, this can often be achieved through "binning", that is combining a set or range of possible values and treating them as one. For example, the chi-square test can be applied to service ports by considering four values: HTTP, FTP, DNS, and "other." Similarly, packet lengths can be binned into ranges such as 0-64 bytes, 65-128 bytes, 129-255 bytes, etc.

In practice, we have found that defining bins dynamically based on the frequency-sorted distribution of values is often best. For example, five bins for the IP source address attribute might be defined as follows: (1) the most frequent address, (2) the next four most frequent addresses, (3) the next 16, (4) the next 64, and (5) the remainder. This approach tends to generate a more stable baseline distribution, increasing detector sensitivity while keeping false alarms in check.

## 3. Response

Our defense approach involves *response modules* that use a characterization of the attack provided by a detection module to take defensive measures. The response module classifies individual packets as benign or suspect based on the attack characteristics provided by the detector. Once identified, the suspect packets are subjected to rate limiting or packet-filtering methods based on the intensity of the attack or pre-defined response policies.

When a detector module detects a DDoS attack, it constructs a description of the flood traffic by identifying attributes that distinguish the anomalous traffic from the more typical flows. For example, in a chi-square detector, attack detection implies that the frequency of one or more traffic "bins" differs substantially from the baseline. The goal of the response is then to bring these frequencies back towards baseline levels, which requires applying rate limits to the over-represented bins. For example, if source address distribution is unusually dispersed due to a random-source-address flood, the detector will note that the bin corresponding to the least common source addresses has a high frequency, and will impose rate limits to drop some of these packets. The detector can further focus response by identifying other dominant attributes (e.g., specific target addresses or ports) of the anomalous traffic and narrowing the rate limits appropriately.

## 4. Prototype Implementations

To evaluate the DDoS attack detection methods under realistic conditions, we implemented prototype detector modules as plug-ins for Snort, the popular, open-source network intrusion detection system [4]. In addition to real-time traffic monitoring, Snort supports off-line processing of previously captured network traffic, making it possible to conduct reproducible detection experiments with traffic data from a variety of environments.

In addition, we implemented a prototype response module for Linux routers as a kernel module. It uses netfilter and Linux Advanced Routing and Traffic Control (LARTC) to filter and rate-limit packets [2],[5]. It can classify packets using specific attribute values (e.g., a given destination TCP port), and it can also be used to filter packets with "random" attribute values using a simple frequency-threshold scheme. An API is provided to take alerts from the detection module and generate filter rules to be issued to the response module. We also produced an extension to the Linux *iptables* mechanism that provides similar functionality, for better integration with iptables-based router/firewall configurations.

A prototype based on the Intel IXP-1200 network processor is currently under development. We consider this processor representative of the next generation of network hardware in that it is a highly programmable device with the capability of forwarding network traffic at high bandwidth. Prototyping on this platform will help to validate the claim that these methods are appropriate for deployment in core network infrastructure.

## 5. Future Extensions

The focus thus far has been on detection and response algorithms and the implementation of these algorithms in software. At issue is whether these algorithms can reliably detect and respond to DDoS attacks.

Against today's relatively unsophisticated DDoS toolkits, our prototype detector is able to determine that the network is under attack and deploy accurate filtering rules. Because baseline measurements and thresholds can be established automatically, and because detectors can generate filtering rules automatically based on the traffic statistics they gather, the system is adaptable to a wide range of network environments with minimal manual tuning. Our initial goal was to provide effective defense against existing DDoS tools, but we are continuing to explore techniques for better defense against future stealthy attacks.

Future research and development will focus on tighter integration of detection and response modules. In the initial implementation, detectors generate concise recommended rules for responders to impose, and no further coordination between the two. In a more tightly coupled detection/response system, the individual packet classification decisions made by the responder could make use of the rich data structures maintained by the detector. Furthermore, detectors inspecting different packet attributes could work together to build more precise flood characterizations. These enhancements would enable more focused filtering and rate limiting, and reduce the possible impact of responses on legitimate traffic.

Another approach to providing more narrowly targeted response while avoiding computationally expensive analysis would be to enable detectors to dynamically tune themselves and "drill down" to investigate detected anomalies more closely. A detector with these capabilities could more effectively allocate its limited computational resources where they are most needed. Such drill-down could be triggered by a vague or uncertain detection by a quick analysis, or by complaints received from downstream network devices.

For further details on these detection and response methods and results of a preliminary evaluation of their potential effectiveness in different network environments, see our full paper [1].

## 6. References


[1] L. Feinstein, D. Schnackenberg, R. Balupari, and D. Kindred, "Statistical Approaches to DDoS Attack Detection and Response," to appear in *Proc. of DISCEX III*, April 2003.

[2] B. Hubert, "Linux Advanced Routing and Traffic Control HOWTO", http://lartc.org/howto/.

[3] R. Manajan, et al., "Controlling High Bandwidth Aggregates in the Network", *SIGCOMM Computer Communications Review*, 32(3), July 2002.

[4] M. Roesch, "Snort - Lightweight Intrusion Detection for Networks" *Proceedings of the 13th Systems Administration Conference (LISA'99)*, USENIX Association, 1999, pp. 229-238, http://www.snort.org/docs/lisapaper.txt.

[5] R. Russell and H. Welte, "Linux Netfilter Hacking HOWTO", http://cvs.netfilter.org/cgi-bin/cvsweb/netfilter/documentation/HOWTO/.

[6] C.E. Shannon, and W. Weaver, *The Mathematical Theory of Communication*, University of Illinois Press, 1963.