
©2005 IEEE. Personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution to servers or lists, or to reuse any copyrighted component of this work in other works must be obtained from the IEEE.
Abstract—Currently Distributed Denial of Service (DDoS) attacks have been identified as one of the most serious problems on the Internet. The aim of DDoS attacks is to prevent legitimate users from accessing desired resources, such as network bandwidth. Hence the immediate task of DDoS defense is to provide as much resources as possible to legitimate users when there is an attack. Unfortunately most current defense approaches cannot efficiently detect and filter out attack traffic. Our approach is to find the network anomalies by using neural network, deploy a system at distributed routers, identify the attack packets, and then filter them. The marks in the IP header that are generated by a group of IP traceback schemes, Deterministic Packet Marking (DPM)/Flexible Deterministic Packet Marking (FDPM), assist this process of identifying attack packets. The experimental results show that this approach can be used to defend against both intensive and subtle DDoS attacks, and can catch DDoS attackers’ characteristic of starting from multiple sources to a single victim. According to results, we find this system can sensitively and accurately detect anomalies caused by DDoS, thus provides high legitimate traffic throughput and reduce attack traffic throughput. Therefore, it can perform well in filtering DDoS attack traffic precisely and effectively.

Keywords—Filtering; DDoS; neural network; traceback; packet marking

I. INTRODUCTION

Distributed Denial of Service (DDoS) attacks have been a serious problem on today’s Internet. A DDoS attack is characterized by an explicit attempt from an attacker to prevent legitimate users from using the desired resources [8]. Many defense approaches have been proposed to fight against DDoS attacks, such as filtering [9], traceback [1], congestion control [11] and replication [27]. However, it is still difficult to solve the notorious problem fundamentally. The reasons lie in two facts. One is the DDoS tools are too easy to get and use, thus even an inexperienced hacker can launch the attack effortlessly. The other reason is that it is difficult to separate the attack traffic from legitimate traffic, and then remove the attack traffic.

Recently many researchers focus their interests on IP traceback. IP traceback is the ability to trace IP packets to their origins without relying on the source address field in the IP header; it provides a system with the ability to identify true sources of the IP packets. This ability is beneficial to locate the attackers and provide judicial evidences for forensics. Many traceback schemes have been proposed, such as link testing [5] [22], messaging [3], logging [20] and packet marking [19] [4]. Packet marking traceback overwrites some fields in the IP header, which are called marks. In particular, an improved DPM scheme, Flexible Deterministic Packet Marking (FDPM) [25], requires a small number of IP packets to find out more sources than other schemes, and has a built-in overload prevention mechanism to intelligently mark packets when system is overloaded in high-speed networks. Instead of investigating traceback schemes, we present a mark-aided distributed filtering approach in this paper, which utilizes the marks of DPM/FDPM to filter out attack traffic. According to experimental results, we find this system can sensitively and accurately detect anomalies caused by DDoS, thus provides high legitimate traffic throughput and reduce attack traffic throughput.

Our contributions in this paper are that we propose an effective DDoS defense mechanism to differentiate the legitimate traffic and attack traffic (which can offer most of network resources to legitimate users); explore the effectiveness of using traceback technique to filter DDoS attack traffic; and apply neural network in finding network anomalies.

II. SYSTEM OUTLINE

Before the system design is described, we have a brief look at the FDPM [25]. As it is shown in figure 1, the encoding module is deployed at the edge routers that are close to the attack source end. When packets enter the network, they are dynamically marked by the encoding modules, which will change marking rate depending on the load of routers. The marks comprise of protocol control bits, address bits, address digest bits and segment number. When the packets reach the victim end, the source IP addresses of entry points can be reconstructed.

In the marks that FDPM uses, the address digest bits in different IP packets are always the same for one entry point. If the attacker sends attack packets, in a large traffic volume, or in a certain rate (eg. 300Kbps), through one entry point, there will be a special pattern of marked packets with the same destination IP address and address digest bits. Therefore, in a global view, there will be a pattern with several groups of packets with corresponding address digest bits, and the same
destination IP address. The pattern reflects clearly the character of DDoS traffic that come from multiple sources and aggregate at one destination. This information is especially beneficial to

- Find out attack traffic; and
- Remove them from legitimate traffic.

![System Architecture](image)

**Figure 1. System architecture.**

Our system utilizes the marks of FDPM, and is deployed between the source end (one hop behind FDPM encoding module) and victim end. The system has two parts, Offline Training System (OTS) and Online Filtering System (OFS). The Offline Training System (OTS) is a lightweight neural network with back-propagation algorithm [10]. This offline system collects traffic characteristics and trains the neural network without influencing the normal operation of the network. It is usually deployed close to the victim end, in order to obtain better training result. Actually, it can be deployed at any point in the protected network. To save the computation time, trained neural networks can also be serialized and be shared for different Online Filtering Systems (OFSs). In this neural network program, a serialized file is from 100kb to 330kb, which is convenient to be exchanged periodically to other OFSs.

The OFS provides the fast decision making function to find the attack signals. Just as the OTS, it can be deployed at any point in the protected network. If it is deployed close to the attack source end, it can prevent even better the rest of network from it to victim, because the attack traffic has been removed before it travels to the victim, without causing overall network congestion. When the attack is confirmed, those packets with the same address digest bits are filtered out.

### III. System Design

#### A. Design of neural network

Although current Internet traffic resembles a stochastic manner, we may still have prior knowledge to the traffic status relevant to traffic classification problems. This prior knowledge has also been applied in many anomaly detection methods, such as statistical method [16] and CUSUM [18]. The high nonlinearity of the Internet traffic makes it difficult to apply these methods to describe precisely and regulate the traffic. Neural network is one of the tools that allow an anomaly detection system to learn the nonlinearity and at the same time, implement linear discriminants. We apply one of the most popular methods for training based on gradient descent in error, back-propagation algorithm to detect traffic anomalies.

There are 3 layers in this neural network, input layer, hidden layer and output layer. The number of the units in the input layer is dictated by the dimensionality of the input vectors (features of traffic). There is one unit in the output layer, representing a value between 0 and 1 (legitimate and attack traffic, respectively). The number of hidden units that governs the expressive power of the net will be introduced in the later part. Here we only apply one hidden layer because it suffices the requirement of preciseness and efficiency. The input layer is a linear layer and the other two are sigmoid layer with transfer function

\[ y = 1/(1 + e^{-x}), \quad y \in (0,1) \]  

In the training phase the desired output must be 0 or 1, and in the test phase the output is between 0 and 1. We use cross entropy as the error criterion function to control the iteration. The cross entropy for \( n \) patterns can be written as

\[ J(w) = \sum_{m=1}^{n} t_m \ln(t_m / z_m) \]  

Where \( t_m \) and \( z_m \) are the target and the actual value of output unit for pattern \( m \), when there is 1 output unit; \( w \) is the weight. The optimal learning rate \( \eta_{opt} \) that satisfies the requirement of convergence and minimum training error can be written as

\[ \eta_{opt} = \left( \frac{\partial^2 J}{\partial w} \right)^{-1} \]  

#### B. Features input

**TABLE I. FEATURES USED (NUMBER PER CERTAIN PERIOD)**

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
<th>Protocol</th>
</tr>
</thead>
<tbody>
<tr>
<td>SrcIP</td>
<td>Number of source IP address</td>
<td>Any</td>
</tr>
<tr>
<td>DestIP</td>
<td>Number of destination IP address</td>
<td>Any</td>
</tr>
<tr>
<td>SrcPort</td>
<td>Number of source port</td>
<td>Any</td>
</tr>
<tr>
<td>DestPort</td>
<td>Number of destination port</td>
<td>Any</td>
</tr>
<tr>
<td>Length</td>
<td>Total length of packets</td>
<td>Any</td>
</tr>
<tr>
<td>Chksum</td>
<td>Number of wrong checksum</td>
<td>Any</td>
</tr>
<tr>
<td>SYN</td>
<td>Number of SYN flag</td>
<td>TCP</td>
</tr>
<tr>
<td>FIN</td>
<td>Number of FIN flag</td>
<td>TCP</td>
</tr>
<tr>
<td>ACK</td>
<td>Number of ACK flag</td>
<td>TCP</td>
</tr>
<tr>
<td>Mark</td>
<td>Concentration of the packets with same digest bits</td>
<td>Any</td>
</tr>
</tbody>
</table>

We use several extracted network traffic features as the input of the neural network for training and test, as shown in table 1, and let the output as the likelihood of attack packets. We apply time window to collect the information of network traffic. Besides the common packet features, the mark (address digest bits) that the FDPM writes into the IP header, is also concerned. Let

\[ X_{mark} = \frac{\text{Number of Packets}}{\text{Number of digests}} \]
This feature means the concentration of the packets that have same digest bits. In practice, we adjust the scale of this special feature, to let the neural network adjust weights from it more than other features during training, because if the neural network prefers this feature over the others, it will be more sensitive to DDoS attacks, according to our experiments. Let

$$x_{mark}' = \beta x_{mark}$$  \hspace{1cm} (5)$$

where $x_{mark}'$ and $x_{mark}$ are the adjusted mark feature and the original mark feature respectively, and $\beta$ is the scaling ratio.

C. Tuning neural network

In this section, we will introduce how to tune parameters of the neural network by experiments. The details of experiments including data and simulation will be presented later. The neural network parameters are learning rate, momentum, and number of hidden units. The learning rate will affect the speed at which the neural network attains a minimum in the criterion function $J(w)$. By experiences we choose this value 0.78. Momentum allows the neural network to learn more quickly by altering the learning rate in stochastic back-propagation to include some faction of the previous weight update. By experiences we choose this value 0.32.

Theoretically, more hidden units can deal with more complex nonlinear problem. However, the training error and test error should be small enough while moderate number of hidden units is chosen. After tuning in the experiments, we found the optimal value of number of hidden units is between 19 and 21, as it is shown in figure 2.

![Figure 2. Choose number of hidden units.](image)

D. Online Filtering System (OFS)

The Online Filtering System (OFS) detects network anomalies and find the attack traffic according to the trained neural network. When the attack is confirmed, those packets with specific marks as the attack packets are filtered out. We test the incoming packets by the trained neural network. If the output indicates anomalies, we further investigate the composition of marked packets. If the number of packets that have the same address digest bits exceeds a threshold $N_{drop}$ (this value is decided by experience), this flow of packets will be filtered. Here flow means the packets have the same destination IP address and digest bits.

This two-step design can not only protect legitimate traffic that shares a large portion of bandwidth but also punish entirely the attack traffic. First, because the anomaly detection is performed by a nonlinear neural network classifier with the assistance of concentration of the packets of same digest bits, the legitimate traffic will be less likely decided as an anomaly than by other coarse granite classifier such as statistical model. Second, once the attack traffic flow is identified, this flow can be totally filtered by differentiating the identity – digest bits that FDPM marks.

IV. Experiments and Evaluation

A. Finding anomalies by neural network

In order to test the capability of the neural network to find anomalies, we conduct experiments by using two public data sets. One is 1998 DARPA Intrusion Detection Evaluation Data Set at Lincoln laboratory, MIT [17]. The other is sanitized UCLA CSD traffic traces from D-WARD project [23]. We extract the features of interest in MIT data sets with time window of 10 seconds. The training data include one week data and a four-hour subset of training data. The features include all the features in table 1 except Mark. The features extracted in UCLA data sets are SrcIP, DestIP, SrcPort, DestPort, and Length. We test different types of attacks (maximum attack rate is 300KBps) such as constant rate attack, pulsing attack and increasing rate attack.

By using different training data set and testing data set, we obtain the fitted ROC curves as figure 3. A ROC curve is a plot with the false positive rate on the $X$ axis and the true positive rate on the $Y$ axis. It can reflect the sensitivity of the neural network by measuring the area below the curve. The point (0, 1) is the perfect classifier; it classifies all positive cases and negative cases correctly. It is (0, 1) because the false positive rate is 0 (none), and the true positive rate is 1 (all). From the figures we can see under each situation the area below the curve is nearly equal to 1, which proofs the neural network approach can detect anomalies sensitively and accurately.

![Figure 3. ROC curves for different types of attacks.](image)

B. DDoS simulation

Currently there is very few data that can describe the whole profile of a DDoS attack. Therefore, besides the MIT and UCLA data sets, we also use the data generated by SSFNet [21] simulator and the embedded DDoS tools [7] in project Distributed Denial of Service Simulators at Deakin University. The reasons of choosing simulator to generate data are first, to obtain control hardware resources such as hosts and networks could be very expensive; second, although to launch DDoS attacks in a laboratory or in real world network and collect data could be direct, it might not be legal; third, it is not easy to change real network topology to create different scenarios; and finally, it is difficult to control the attack process in a real environment because there are too many factors that can affect the result.

In the above project, two DDoS tools, TFN2K and Trinoo, are adopted and integrated into SSFNet to create virtual DDoS networks to simulate the attacks. The TFN2K and Trinoo are originally written in C language. They are ported to Java to be embedded into SSFNet. By the DDoS simulators, we can
launch any DDoS attack with different features such as duration, protocol, attack rate, etc.

In order to simulate the DDoS attack as real as possible, we also use the real Internet topology from Cooperative Association for Internet Data Analysis (CAIDA)'s Skitter project [6]. The data set used is generated from server aroot ipv4.20040120 on 09-Jan-2004. To simplify the problem, we connect all routers by 100M network interfaces. We randomly choose the 1000 attack hosts and let the rest be legitimate clients, and let the Skitter server be the victim. Constant rate attack of 300KBps is applied to all attack hosts. According to the hop distribution (number of routers between the victim and its clients), most of the clients locate in the distance between 10 hops and 25 hops. Therefore, we deploy the FDPM encoding module at routers 10 hops from the victim, and the mark-aided distributed filtering systems at routers from 1 to 9 hops from the victim.

To test the sensitivity and accuracy of neural network in the simulation, we apply TFN2K to launch the attacks and obtain the ROC curve. From the comparison in the last of figure 3 we can see that there is no mark-aided feature used to train the neural network, the area below the ROC curve is smaller than the mark-aided approaches. We also adjust the scaling ratio $\beta$ in equation (5) to get different results. When $\beta=10$ the neural network can obtain better detection result than $\beta=1$ (no input is scaled) and $\beta=100$. It is shown that the parameter $\beta$ can affect the sensitivity and accuracy of neural network. However, how to set a best value still needs more research.

C. Performance

The ultimate goals of our system are to find out the attack traffic as accurately as possible, and to filter out the attack traffic as much as possible and at the mean time let as much legitimate traffic pass through as possible (but not to detect anomalies). Therefore, the performance metrics are average value of legitimate traffic passed rate (LTPR) and attack traffic passed rate (ATPR) of distributed filtering systems. We deploy the mark-aided distributed filtering system at different distances from the victim and conduct experiments based on both TFN2K and Trinoo DDoS tools. Random algorithms in SSFNet are used to generate legitimate traffic. After the neural network is trained, the DDoS tools are initiated to start the attack with 300KBps attack rate. Then the traffic on the deployment points is monitored. Figure 4 shows the average values of LTPR and ATPR at routers that locate at different hops from the victim. From the figures we can see our scheme can filter out most of the attack traffic and let most of the legitimate traffic pass through. These two figures also show that both LTPR and ATPR decrease slightly if the defense systems are deployed close to the attack source end. This proofs this system can be deployed at any place in the protect network. Actually, if the filtering system is deployed close to the attack source end, it can protect the rest of the network from congestion.

FDPM can change its marking rate dynamically at its encoding modules according to the load of participating routers. This ability can intelligently find the most possible attack packets to be marked. From figure 5 we can see that the performance of LTPR and ATPR change according to the marking rate. Moreover, if attacking packet rate increases, our scheme can let even more legitimate packets pass through, and filter more attacking packets.

V. RELATED WORK

A. Anomaly detection methods

DDoS attacks usually cause network anomalies. Statistical method [16] is a straight forward method to detect anomalies. However, it requires a strong assumption that the network traffic variables obey a Normal Distribution. Another popular
method to detect anomalies is nonparametric Cumulative Sum (CUSUM) method [18]. It is stateless, lightweight, and sensitive to persistent sudden changes caused by DDoS attacks instead of Internet flash crowd. This method has been used to detect many DDoS anomalies such as SYN flood [24]. However, this method can only consider one network feature, and can only deal with the change point problem. If the network anomaly is not an intensive flood, this method may not discover the attack timely.

Rather than analyzing the change of features, multivariate correlation analysis [14] [28] that is proposed to detect subtle DDoS attacks considers the correlations among the features. Loss of self-similarity [2] is also a representation of relationships between features. However, there is no theoretical proof to decide which features are valid for the correlation models and how important each feature is. Additionally, those methods can only represent the changes of correlation, but not the causality between those changes and attacks.

B. Filtering methods

Ingress filtering [9] is proposed to be deployed on the external interface of a network and drops all spoofed incoming packets. It requires a global deployment and also a knowledge base of legitimate IP addresses that can be very large. Other filtering methods such as router-based Distributed Packet Filtering (DPF) [22] and Hop-Count Filtering (HCF) [13] are proposed to drop spoofed IP packets by detecting the networking features such as the number of hops a packet takes to reach its destination. Some filtering approaches [11] [12] depend on network congestion, which means only intensive congestion can trigger the filtering mechanisms. Both Path Identifier (PI) [26] and Deterministic Bit Marking (DBM) [15] create a path signature for all the packets originating from the same location upon arriving at a destination. By identifying this signature it isolates and filters DDoS traffic. However, the detection is affected by the distance in number of router hops, resulting in low detection rate if the attacks come from hosts that are far away from the victim.

VI. CONCLUSION

In this paper, we present a distributed filtering system by utilizing the marks that DPM/FDPM writes into the IP header. Neural network is applied to detect the network anomalies. The experimental results show that our approach is sensitive and accurate in finding DDoS attacks. It can filter out most of attack traffic, and let most of legitimate traffic pass through.

REFERENCES


