DDoS Attacks: Intelligent Detection Technique using ANNIFS

Rakesh Suryawanshi[#1], Vaibhav Survase[*2]

[#]*Department Of Computer Engineering, University Of Mumbai*
A.C. Patil College of Engineering, Kharghar, Maharashtra, India
[1]rakeshsuryawanshi@gmail.com
[*]*Student at A.C. Patil College of Engineering*
Kharghar, navi Mumbai, Maharashtra, India
[2]*vvsurvase@gmai.com*

Abstract— **The occurrence of distributed denial of service (DDoS) attacks has become more frequent in today's network environment. Detecting these attacks would prevent the unnecessary utilization of resources which otherwise could have been used to service legitimate users. Adaptive Neuro Fuzzy Inference System (ANFIS) has been used as the hybrid intelligent system for the detection of DDoS attacks. The aim is to provide a proactive DDoS detection and defense mechanism by proposing knowledge based systems in ANFIS by training the data over true and false data packets in a netork. ANFIS trains the data and the attack data can be detected from a large datasets. It is found that ANFIS is able to classify the TCP SYN DDoS data with very good precision. Use of hybrid intelligent systems provides effective detection of DDoS attack.**

Keywords - ***DDoS, ANFIS, Defence mechanisms***

INTRODUCTION

Distributed denial of service (DDoS) is a type of Denial of service (DoS) attack where multiple compromised systems-usually infected, are used to target a single system causing a Denial of Service (DoS) attack. A distributed denial of service (DDoS) attack is a large scale, coordinated attack on the availability of services of a victim system or network resources, launched indirectly through many compromised computers on the Internet. According to [1], in August 1999 Trinoo was deployed in at least 227 computer systems to flood a single computer of University of Minnesota, was first documented DDoS attack. The system was unresponsive for more than two days. On February 7, Yahoo was the victim of DDoS attack, as a result its Internet portal was inaccessible for three hours. Next day, Amazon, Buy.com, CNN and eBay were all hit by DDoS attacks that slowed them down significantly. The main goal is to inflict damage on the victim. The motives can be personal reasons or prestige. Some of them can be performed for material gain or for political reasons. The victim might not be the actual target of the attack, but others who rely on the target's operations. Attacks are increasing in numbers, size and in complexity as new types of DDoS attacks are emerging.

DDOS ATTACKS AND THEIR ARCHITECTURE

While designing the Internet, the prime concern was to provide for functionality, not security and this is what makes attackers and the attack tools powerful. The design of internet raises security issues concerning opportunities for DDoS attacks.

- *Internet security is highly interdependent.* Regardless of how well secured the victim system may be, it is susceptibility to DDoS attack depends on security of the rest of the Internet.
- *Limited Internet Resouces.* Every Internet host has limited resources which can be exhausted by a large number of users.
- *Intelligence and resources are not collocated.* End-to-end communication paradigm led to storing most of the intelligence needed for service guarantees with end hosts, limiting the amount of processing in the intermediate network so that packets could be forwarded at minimal cost. A desire for large throughput led to higher bandwidth pathways in the intermediate network. Thus, malicious clients can misuse the abundant resources of intermediate network for delivery of numerous messages.
- *Accountability is not enforced.* IP spoofing gives attackers a powerful mechanism to escape accountability for their actions, and sometimes even the means to perpetrate attacks.
- *Control is distributed.* Internet management is distributed. There is no way to enforce global deployment of a particular security mechanism or security policy, and due to privacy concerns, it is often impossible to investigate cross network traffic behavior.

A. *DDoS Strategy*

A DDoS attack is composed of several steps:

1. *Selection of Agents:* The attacker employs a large number of vulnerable hosts to launch an attack instead of using a single server, which is less effective and gets easily detected.

2. *Compromise:* The attacker exploits the security holes of the agent machines and inserts the malicious code. It also protects the code from identification and deactivation. These agent programs are cost effective in terms of memory and bandwidth, therefore does not affect performance of the system effectively. In *direct DDoS attack* strategy, large numbers of compromised nodes called zombies are employed through high bandwidth Internet.

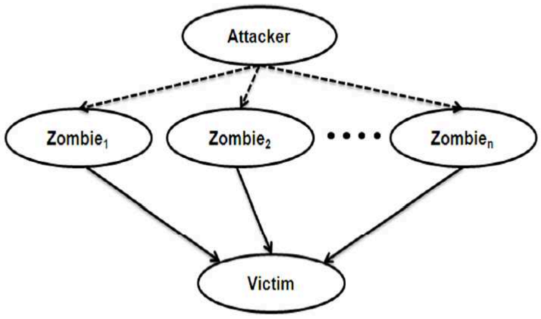


Figure 1: Direct DDoS Attack

On the other hand, *Indirect DDoS attack* strategy is more complex due to inclusion of reflectors between zombies and victims further complicating the traceback. Self-propagating tools like Ramen worm and Code Red are used to automate this phase. It is usually difficult for an agent systems to realize that they have become a part of DDoS attack system.

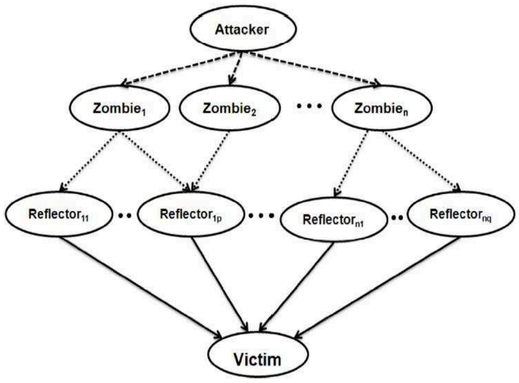


Figure 2: Indirect DDoS Attack

3. *Communication:* The attacker regularly communicates with handlers to identify which of the agents are up and running, when to schedule attacks, or when to upgrade agents. These communications are carried out on network protocols such as TCP, ICMP, or UDP. Agents can also communicate with single or multiple handlers. During direct communications, the agent and the handler needs to know each other's identity in order to communicate. This is usually done by hardcoding IP address of handler machines in attack code which is later installed in agent machines. Drawback of this approach for attacker is that discovery of one compromised machine can expose whole DDoS network. Also, as they actively listen to the communication channel, they can be identified by network scanners. Whereas *Indirect* approach the agents do not actively listen to the network connections, but instead use a legitimate IRC service as a result of which their control packets cannot be easily differentiated from legitimate chat traffic. To avoid detection further, attackers frequently deploy channel hopping, using random IRC service for short period of time.
4. *Attack:* The attacker initiates the attack. The victim, the duration of attack, attack type, length, TTL, and port numbers can be adjusted. Variations in the attack packets are necessary as they complicate the detection.

B. Recruiting the Vulnerable Machines

Attackers can use different kinds of scanning techniques to find vulnerable machines. Some of these are:

1. *Random Scanning*: In this technique, infected machine probes IP addresses randomly from the IP address space and checks their vulnerability. Whenever it finds a vulnerable machine, it breaks into it and tries to infect it, installing on it the same malicious code that is installed on itself.

2. *Hit-list scanning:* Before scanning, attackers collect a list of a large number of potentially vulnerable machines. In order to find vulnerable machines they scan the list. When they find one, they install the malicious code on it and divide the list into half. They share the other half with the newly compromised system, keep the remaining half and continue scanning the remaining list. The process repeat itself whenever a new vulnerable machine is found.

3. *Topological Scanning*: This method uses information contained on the victim machine in order to find new targets. An already compromised host looks for URLs in the disk of machine that it wants to infect. It renders these URLs target and checks their vulnerability. The fact that these URLs are valid web servers means that the compromised host scans possible targets directly from the beginning of the scanning phase. Therefore, the accuracy of this technique is extremely good.

4. *Local subnet scanning*: This type of scanning acts behind a firewall in an area that is considered to be infected by the malicious scanning program. The compromised host looks for targets in its own local networks. More specifically, a single copy of the scanning program is running behind a firewall and tries to break into all vulnerable machines that would otherwise be protected by the firewall.

5. *Permutation scanning*: In this type of scanning all machines share a common pseudorandom permutation list of IP addresses which is constructed using block cipher of 32 bits with a pre-selected key. If a compromised host has been infected during either hit list

scanning or local subnet scanning, it starts scanning just after its point in the permutation list and scans through this list to find new targets. If it has been infected during permutation scanning, it starts scanning at random point. Whenever it encounters an already infected machine, it chooses a new random start point in the permutation list and proceeds from there. The process of scanning stops when the compromised host encounters infected machines sequentially.

C. Propagaion of Malicious code

There are three ways of propagation of malicious code given below:

1. *Central source propagation.* In this mechanism, after the discovery of agents, instructions are given to a central source so that a copy of attack toolkit is transferred from a central location to the newly compromised systems. After the transfer of toolkit, automatic installation of the attack tools takes place on agents, controlled by scripting mechanism. This initiates another attack cycle, in which the infected system looks for other vulnerable systems on which they can install the attack toolkit. This mechanism commonly uses HTTP, FTP and RPC protocols for communicating.
2. *Back-chaining propagation.* In this mechanism, the attack tools that installed on the attacker include special methods for accepting a connection from a newly compromised system and sending a file to it containing attack toolkit. This back-channel file copy can be supported by simple port listeners that copy file contents or by full intruder-installed web servers, both of which use the Trivial File Transfer Protocol (TFTP).
3. *Autonomous propagation.* In this mechanism, the attackers transfer the attack tools to the newly compromised system at the exact moment that it breaks into that system. This mechanism differs from the above as in this the attack tools are planted into the compromised hosts by the attackers themselves and not by external file source.

Once the attack network is set up, the intruders use handler machines to specify attack type and victim's address and wait to mount the attack. The agent machines then floods the victim's system with useless load and exhausting its resources. In this way, attackers makes the victim's machine unavailable to legitimate users and obtain unlimited access to it. The volume of traffic may be too high that the network suffers from low performance thereby, denying the services to other users.

TYPES OF DDoS ATTACKS

DDoS attacks can be classified as *flood attacks* and *logic or software attacks*. In *flood attacks*, the attacker continuously sends large amount of data to target machine to consume its network bandwidth, processing power, memory. Whereas in other type of attack, the attacker sends malformed packets to exploit the vulnerability of the software loaded in the target machine.

1. Flood Attacks

1.1. TCP-SYN Flood: This attack exploits the three way handshaking of TCP connections. The attacker initiates a connection request with spoofed IP address to target machine. The target machine replies and waits for a reply that never comes. This holds the resources of target machine like memory and processing power.

1.2. UDP Flood: In this type of attack, attacker can send a tempered packet to a random port of the target machine. When victim finds no application waiting on the port after receiving the packet, it generates destination unreachable ICMP packet. This packet is then sent to the sources address of the received packet. If a packet sent by the attacker are in large numbers, then most of the resources of victim will be held up and machine will go down.

1.3. ICMP Flood: In this type of attack, the victim is bombarded by number of pings and UDP packets which results in slow network eventually leads to loss of connectivity.

1.4. Smurf Attack: In this attack type, attacker sends forged ICMP echo packets to broadcast addresses of vulnerable networks. All the systems on these networks reply to the victim with ICMP echo replies. Bandwidth available to target machines will exhaust rapidly if packets are generated by attacker is large in numbers.

2. *Logic or Software Attacks*

2.1. *Ping of Death*: Attacker sends an ICMP echo request with IP packet size greater than the maximum size allowed, the victim cannot reassemble the packets. As a result the operating system may crash or reboot. This will deny users from accessing the resources found on the victim.

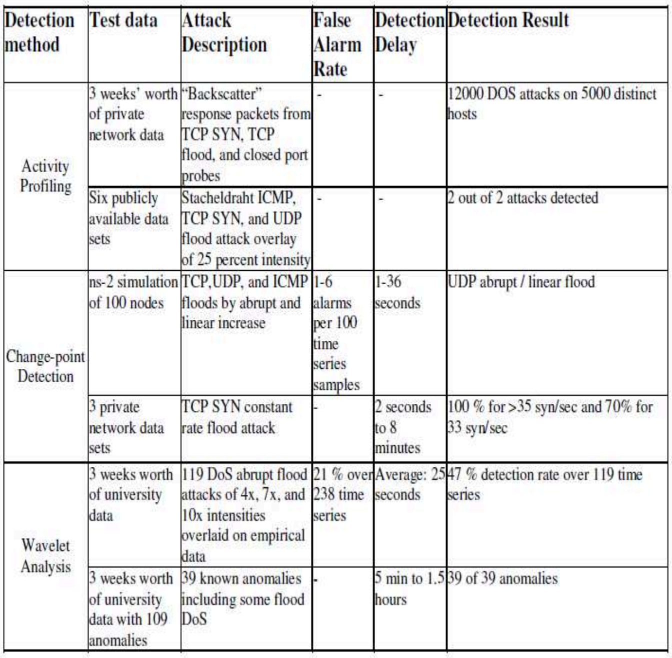
2.2. *Teardrop*: Teardrop attack exploits an overlapping IP fragment problem in some common operating systems. In this method, two fragments of a packet that cannot be reassembled using the offset value of the packet are sent. These packets will crash or reboot the target machine.

2.3. *Land*: A forged packet with the same source and destination IP address is sent to the victim. This raises confusion and may crash or force reboot of the system.

2.4. *ECHO/CHARGEN*: A character generation service generates a series of characters whenever it receives a UDP packet, while an echo service echoes any character it receives. Exploiting these two services, the attacker sends a packet with the source spoofed to be that of the victim to another machine. Echo service of the former machine echoes the data of that packet back to the victim's machine and the victim's machine, responds in same way. Hence, a constant stream of useless load is created that burdens the network.

DETECTION OF DDOS ATTACKS

According to the survey done by [3], there are three general ways of DoS detection methods. These are activity profiling, sequential change-point detection and wavelet analysis. This survey is summarized in table 1. It was found that average time for detection using change point detection is smaller than that of wavelet analysis.



Detection method	Test data	Attack Description	False Alarm Rate	Detection Delay	Detection Result
Activity Profiling	3 weeks' worth of private network data	"Backscatter" response packets from TCP SYN, TCP flood, and closed port probes	-	-	12000 DOS attacks on 5000 distinct hosts
	Six publicly available data sets	Stacheldraht ICMP, TCP SYN, and UDP flood attack overlay of 25 percent intensity	-	-	2 out of 2 attacks detected
Change-point Detection	ns-2 simulation of 100 nodes	TCP,UDP, and ICMP floods by abrupt and linear increase	1-6 alarms per 100 time series samples	1-36 seconds	UDP abrupt / linear flood
	3 private network data sets	TCP SYN constant rate flood attack	-	2 seconds to 8 minutes	100 % for >35 syn/sec and 70% for 33 syn/sec
Wavelet Analysis	3 weeks worth of university data	119 DoS abrupt flood attacks of 4x, 7x, and 10x intensities overlaid on empirical data	21 % over 238 time series	Average: 25 seconds	47 % detection rate over 119 time series
	3 weeks worth of university data with 109 anomalies	39 known anomalies including some flood DoS	-	5 min to 1.5 hours	39 of 39 anomalies

Table 1: Ways to detect DoS Attacks

A. *DDoS Attacks Defense mechanism*

According to [1], DDoS Defense mechanism schemes can be divided into three classes, based on their locality of deployment. These are victim-end, source end, and intermediate router defense mechanism.

1. *Victim-end defense mechanism*
   This mechanism is employed in the routers of victim network. A generic architecture of such schemes is shown in Figure 3. The detection engine is used to detect intrusion either online or offline, using either misuse or anomaly based intrusion detection. The reference data stores information about known intrusion signatures or profiles of normal behavior. This information is updated whenever a new knowledge about the observed behavior becomes available. The security manager often updates the stored intrusion signatures and checks for critical events like false alarms. Detecting DDoS attacks at victim end is relatively easy because of the high rate of resource consumption. During DDoS attacks, victim resources such as network bandwidth, memory, often gets overwhelmed and these approaches cannot stop the flow beyond victim routers. Also, these approaches detect the attack only after it reaches the victim and detecting attacks when legitimate users have already been denied is not useful.

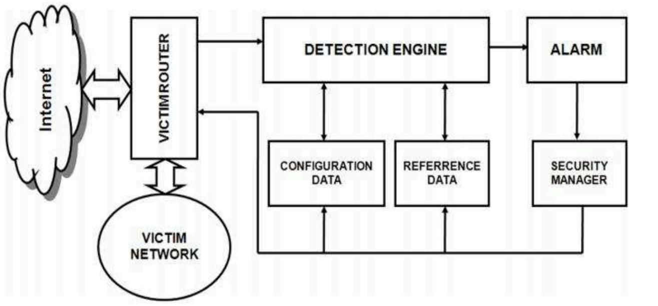


Figure 3: Generic architecture for victim-end DDoS defense mechanism

2. *Source end defense mechanism*

This architecture is similar to victim end defense mechanism. Additionally, a throttling component is added to impose rate limit on outgoing connections. The observation engine compares both incoming and outgoing traffic statistics with some predefined normal profiles. A generic architecture of this defense mechanism is shown in Figure 4.

Detecting and stopping a DDoS attack at source is best defense mechanism. Detecting DDoS attacks at source end is not easy as sources are widely spread and a single source behaves almost similarly as in normal traffic. Also, deployment of defense mechanism at source end is difficult.

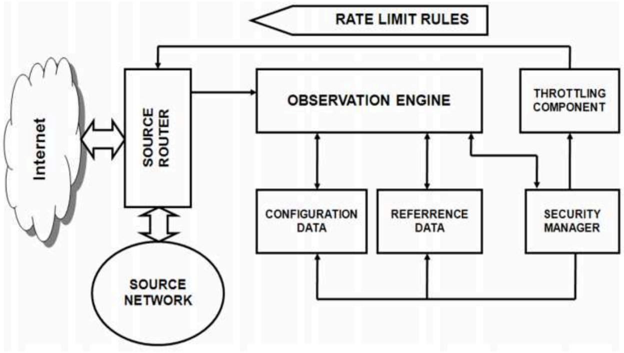


Figure 4: Generic architecture for source-end based DDoS defense mechanism

3. *Intermediate network defense mechanism*

This kind of schemes is generally collaborative in nature and the routers share their observations with other routers. It balances the tradeoffs between detection accuracy and attack bandwidth consumption. A generic architecture is present in Figure 5.

Detection and traceback of attack sources are easy in this approach due to collaborative operation. To achieve full detection accuracy, all routers on the internet should employ this detection scheme, because of unavailability of this scheme in only few routers leads to failure of detection and traceback.

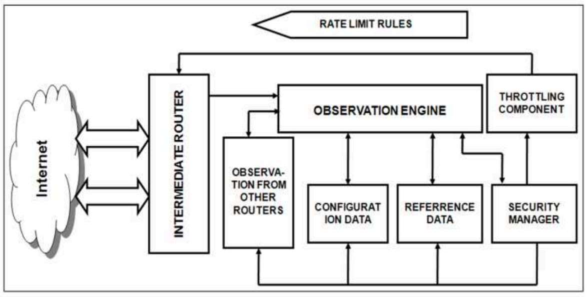


Figure 5: Generic architecture for intermediate network based DDoS defense mechanism

B. Hybrid Intelligent System

A hybrid intelligent system is the one which combines at least two intelligent techniques. The combination of realistic reasoning, fuzzy logic, neural networks and other forms of core soft computing.

Soft computing is capable of operating with uncertain, imprecise and incomplete information in a manner that reflects human thinking. Humans use words, and soft computing attempts to model our sense of words in decision making. Soft computing exploits the tolerance for uncertainty and imprecision to achieve greater traceability and robustness and lower the cost of solutions.

A hybrid intelligent system can be good or bad – it depends on which components constitute the hybrid. Each component has its strength and weaknesses. A good hybrid system brings the advantages of these technologies together. Their synergy allows a hybrid system to accommodate common sense, extract knowledge from raw data, use human-like reasoning mechanisms, deal with uncertainty and imprecision, and learn to adapt to a rapidly changing and unknown environment.

A neuro-fuzzy system is a neural network that is functionally equivalent with fuzzy inference model. It can be trained to model to develop If-then fuzzy rules and determine membership function from input and output variables of the system. Expert knowledge can be easily incorporated into the structure of the neuro-fuzzy system while the connectionist structure avoids fuzzy inference, which entails a substantial computational burden. In fuzzy systems there are two commonly used fuzzy inference models; the Mamdani Fuzzy Inference (Mamdani *et al*,1975) model and the Sugeno (Sugeno, 1985) Fuzzy Inference model. The Mamdani-style inference, requires the calculation of the centroid of a two dimensional shape by integrating across a varying function. This process is not computationally efficient. To make neuro-fuzzy systems computationally effective, a neural network that is functionally equal to a Sugeno fuzzy inference model was proposed by Roger Jang (Jang, 1993).This model is called Adaptive Neuro-Fuzzy Inference System.

C. *ANFIS*

The Sugeno fuzzy model is used to generate fuzzy rules from a given input output dataset. For example, Sugeno fuzzy rule:
IF x_1 is A_1
AND x_2 is A_2
……
AND x_1 is A_m
THEN $y = f(x_1, x_2,\ldots.., x_m)$
where x_1 , x_2 ,…, x_m are input variables; A_1 , A_2 ,…, A_m are fuzzy sets; and y is either a constant or a linear function of the input variables. When y is a constant, we obtain a zero-order Sugeno fuzzy model in which the consequent of a rule is specified by a singleton. When y is a first-order polynomial, i.e.
$y = k_0 + k_1x_1 + k_2\, x_2 + \ldots + k_m\, x_m$
The ANFIS has six layers of feed forward neural network.
Assumptions for the first order Sugeno fuzzy model in

- Two input x1 and x2
- One output y
- Each input is represented by two fuzzy sets
- The output is represented by a first order polynomial
- Figure 9 ANFIS implements the following four rules

Rule 1:
IF x_1 is A1 AND x_2 is B1
THEN $y=f_1=k_{10}+k_{11}x_1+k_{12}x_2$

Rule 2:
IF x_1 is A2 AND x_2 is B2
THEN $y=f_2=k_{20}+k_{21}x_1+k_{22}x_2$

Rule 3:
IF x_1 is A2 AND x_2 is B1
THEN $y=f_3=k_{30}+k_{31}x_1+k_{32}x_2$

Rule 4:
IF x_1 is A1 AND x_2 is B2
THEN $y=f_4=k_{40}+k_{41}x_1+k_{42}x_2$

Where,
A1 and A2 are fuzzy sets in the universe of discourse X1
B1 and B2 are fuzzy sets in the universe of discourse X2
k_{i0}, k_{i1}, and k_{i2} is a set of parameters specified for rule *i*
ANFIS has several layers,
Layer 1, the input neurons pass external crisp signals to Layer 2.
$y_i^{(1)} = x_i^{(1)}$
where $x_i^{(1)}$ is input and $y_i^{(1)}$ is the output of the neuron I in Layer 1.
Layer 2 is the fuzzification layer. In ANFIS, fuzzification neurons have a bell activation function. A bell activation function is specified as
$y_i^{(2)} = 1 /[1+((x_i^{(2)} -a_i)/c_i)^{2b_i}]$,

where a_i, b_i and c_i are parameters that control the centre, width and slope of the bell activation function of neuron *I* and $x_i^{(2)}$ and $y_i^{(2)}$ are input and output neurons in Layer 2.
Layer 3 is rule layer. The output of neuron *i* in this layer is obtained as,
$y_i^{(3)} = {}^{k}{}_{j=1} \Pi\, x_{ij}^{(3)}$

Layer 4 is Normalization layer. Each neuron in this layer receives input from all neurons in the rule layer, and calculates the normalized firing strength of a given rule to the final result.

$y_i^{(4)} = x_{ii}^{(4)} / ({}_{j=1}{}^{n}\sum x_{ji}^{(4)}) = \mu_i / ({}_{j=1}{}^{n}\sum \mu_j)$

Layer 5 is defuzzification layer. Each neuron in this layer is connected to the respective normalization neuron and also receives intial inputs x1 and x2.

$y_i^{(5)} = x_i^{(5)} [k_{i0}+k_{i1}x_1+k_{i2}x_2]$
Layer 6 is represented by a single summation neuron. This neuron calculates the sum of outputs of all defuzzification neurons and produces the overall ANFIS output, i.e., y,

$y = {}_{i=1}{}^{n}\sum x_i^{(6)}$

It is very difficult for one to specify a rule consequent in a polynomial form. So, it is not necessary to have prior knowledge of them for ANFIS to deal with problem. ANFIS learns these parameters and tunes membership functions.

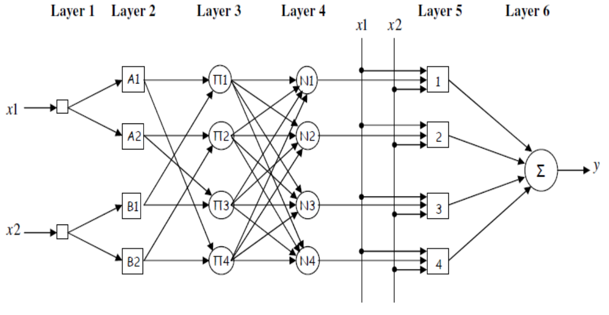


Figure 6 Addaptive Neuro –Fuzzy Inference System

Learning Algorithm
ANFIS uses hybrid learning algorithm that combines the least-squares estimator and the gradient descent method. In ANFIS training algorithm, each epoch is composed of forward pass and backward pass. In the forward pass, a training set of input patterns is ANFIS, neuron outputs are calculated on the layer basis, and rule consequent parameters are identified by the least-square estimator. In the backward pass, backward propagation algorithm is applied. The error signals are propagated back, and the antecedent parameters are updated according to the chain rule. In ANFIS training algorithm, both antecedent parameters and consequent parameters are optimized. In forward pass, the consequent parameters are adjusted whereas in backward pass, the antecedent parameters are tuned while the other remains fixed. In case of small input and output data, membership functions can be decided by human experts.

D. Analysis, Design and Implementation

DDoS detection using an intelligent system requires data for learning and testing, data preprocessing, an intelligent system and interpretation of the output of the intelligent system. The output of the intelligent system is interpreted to determine if an attack is going on or not. The proposed system is shown in Figure 7.

1. Experimental Setup for data collection

It consists of attacker and the victim, and the zombie machines. All zombie machines put in one network and the master and victim in another. After zombies are ready to attack, the attacker sends an attack command. Based on the command, zombies start to generate attack network traffic. At victim's end data was collected for different scenarios.

2. Data Generation

The attack tool is used to generate attack traffic data. The purpose of data collection module is to listen to and record any network traffic arriving at victim machine. The attack tool has two parts, the communication part and generation part. The purpose of communication part is to wait and listen to commands sent by the master/attacker. When an attack command is received it initiates the traffic generation part and passes along the command received the generation module. The generation part accepts the command and prepares itself for the type of traffic orders for. Once attack traffic is ready, it sends out attack traffic to victim machine for defined amount of time. On the server side, it waits for clients to connect to it and waits for the attacker to key in commands. The attack tool had the capability to integrate itself with the system and automatically activate itself. Hence this capability had to be removed to protect the network. Once this process is finished, the clients will send in message notifying the successfulness of that command.

3. Preprocessing

In this module, the data collected is processed in such a way so that is more appropriate for detection process. The best features for identification of DDoS attacks are SYN and URG flags, the probability of distinct Source Ports in each timeframe, the number of packets that use certain port ranges, the TTL and the window size in each timeframe, bit rate and rate of change of bit rate.

The percentage of packets found,
%packets=(number of packets/sec)/(count of packets arrived/sec) *100

The probability of distinct source ports,
probDistinctSourcePorts=(numberof distinct source ports/sec)/(count of packets arrived/sec)

bit rate is given in kilobytes per second,
datarate=number of bytes received/time elapsed
rate of change of data rate= (current data rate-previous data rate)/time elapsed

These computed values are then input to the intelligent detector as training and testing data. Along with these values the desired outputs for each data type is also appended.

4. The Hybrid Intelligent Detection System (ANFIS)
The main purpose of this system is to learn and/or decide on the nature of network traffic. The main advantage of this system is that it does not need an expert to set the rules that are used by inference system. It generates the necessary rules from input-output training data set. The input of this module is the output from preprocessing module.

5. Output Interpreter
This module analyses the output by ANFIS and tries to present the meaning of the result to user. It checks if there is an attack, a possible attack or no attack. This can be identified by checking the output against some thresholds which are prepared for the classifications.

6. Implementation
Client connects to the server when it is activated / run. It then sends a message notifying the server of its readiness. When it receives a command from the server, it sends back an acknowledgement stating it is executing the command. It then creates a process, which is the attack tool, passing along the command received. After waiting for the attack tool to finish executing the command, it sends a message to the server regarding the successfulness of the command. The flow chart in Figure 7 shows the process.
The flow chart in master server in Figure 8 shows the process at server side. The master server listens to connection requests from client applications. It checks to see if it has enough resources to entertain the request. If there is available resource, it accepts the request and associates the client with a client interface socket and goes back listening for further connection requests. The server application goes through each client interface socket which currently in connection with a client and passes the command. The client interface sockets in turn send this command to the client applications.
The data collection module uses RAW sockets to listen to any type of network traffic arriving at the victim's computer. It then goes on processing the packets captured to identify their type. It also keeps count of any type packet that arrived at the victim. Another thing it does is keep track of distinct source ports. Only one occurrence of a distinct port is recorded. The data collected is summarized by the second and when the designated time of data collection is reached, these values are written to a file.

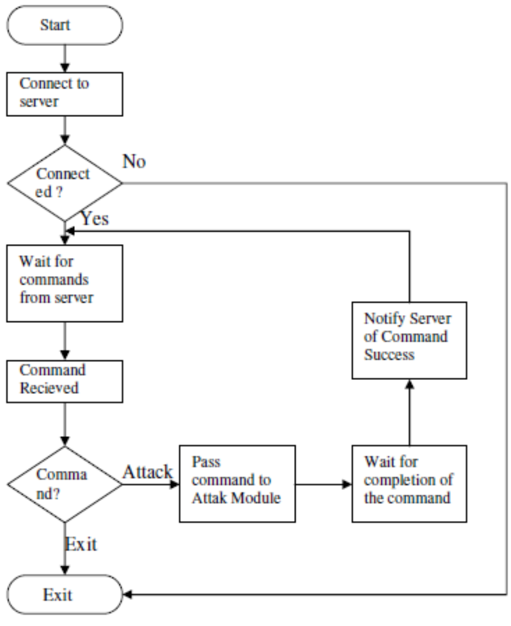

Start
Connect to server
Connect ed ?
No
Yes
Wait for commands from server
Notify Server of Command Success
Command Recieved
Comma nd?
Attack
Pass command to Attak Module
Wait for completion of the command
Exit
Exit

Figure 7 Flow chart of the client application

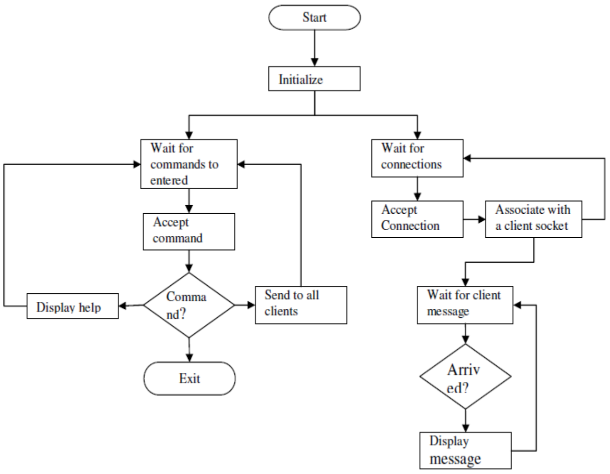


Figure 8 Flow chart for the Master server application

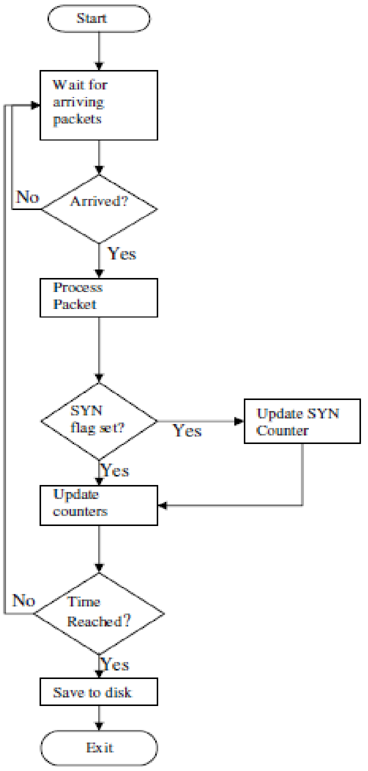


Figure 9 Flow chart for the data collection module

RESULTS

The percentage of TCP SYN packets and the probability of distinct source ports were found to decrease as compared to pure attack traffic. Also these values were much higher than that of the normal traffic data ascertaining the existence of flooding attack. The lowering of TCP SYN percentage and the probability of distinct ports arises from the fact that normal traffic and attack traffic exist together in this configuration. Using this type of data it was possible to train the hybrid intelligent system to classify the network traffic. During the training process the ANFIS generated rules for use in classification of network traffic. Table 2 shows the results for the different type of data packets errors built.

TABLE 2: DATA PACKET ERRORS

UDP	0.9688
ICMP	0.97312
SYN flood	0.97486
TCP	0.86732
Distributed	0.91329

CONCLUSION

The objective of this paper was to investigate the capability of ANFIS in the detection of DDoS attacks. ANFIS was selected because of its capability of approximate the uncertain data in TCP SYN flood attacks. ANFIS was able to differentiate between the attack and normal data.

ACKNOWLEDGMENT

I would like to thank the authors of the reference papers for their helpful and valuable discussions and experiments. Any opinions, findings, and conclusions or recommendations expressed in this paper are those of the authors.

REFERENCES

[1] Monowar H. Bhuyan, H. J. Kashyap, D. K. Bhattacharyya and J. K. Kalita, “Detecting Distributed Denial of Service Attacks:Methods, Tools and Future Directions”.

[2] Arbor Networks, Layered Intelligent DDoS Mitigation Systems.

[3] Glenn Carl, George Kesidis, Richard R. Books, Suresh Rai, “Denial of Service Attack-Detection Techniques”, IEEE Internet Computing, Jan/Feb 2006.

[4] Fitsum Assamne, “Distributed Denial of Service Attack Detection: A Hybrid Intelligent System Approach”, April,2008

[5] Amit Khajuria, Roshan Srivastava, “Knoledge Based system for detection and prevention of DDoS attacks using fuzzy logic”, International Journal of Enhanced Research in management and computer applications.