

Classification, Detection and Defense mechanisms for DDoS attacks

Prakash Gupta

Department of Computer Science, University of Southern California

941 W. 37th Place, Los Angeles, CA USA

Email: Prakash.Gupta@usc.edu

Abstract: In today's internet era, one of the major threats and hardest security problems to address are Denial of Service (DoS) attacks. In particular of prime concern are DDoS attacks which are capable of multiplying the effectiveness of the DoS significantly. A DDoS attack has such a severe impact that it can easily exhaust the computing and communication resources of its victim within very short time with no advance warning. The goal of the paper is to study various structural approaches to DDoS problem by classifying DDoS attacks and developing various defense mechanisms. We focus upon various Intrusion Detection techniques that are deployed and compare their effectiveness along different parameters for their effectiveness in detecting novel attacks.

Keywords: DoS attacks, DDoS attacks, intrusion detection, neural network, fuzzy logic.

1. Introduction

Due to rapid growth of computer networks and in particular internet, need for open distributed systems have become increasingly popular. With the fruits of enormous amount of shared data, high speed networking, reduced cost, it has also brought with itself the stability and security issues. With the leap increase in the usage of e-commerce infrastructure, the attackers have started devising and launching sophisticated attacks motivated by financial, political and even military objectives. While the security goals of confidentiality and integrity have been addressed by research community to a greater extent but the hardest goal i.e. availability of data and service still remains to be addressed.

One of the greatest threats that network security is facing under availability issue is Denial of Service (DoS) attacks. The DoS are of prime concern not only because of their ability to cause irreparable disruption of services but because of problem in addressing these attacks. In DoS attack, the attacker aims at rendering a network incapable of providing normal service by targeting either the network's bandwidth or its connectivity.

According to [4], the most common DoS attacks target the computer's network bandwidth or connectivity. Bandwidth attacks flood the network with such a high volume of traffic that all available network resources are consumed and legitimate user requests can not get through. Connectivity attacks flood a computer with such a high volume of connection requests, that all available operating system resources are consumed and the computer can no longer process legitimate user requests.

As per [11], DoS attacks can be classified into 5 categories based on the attacked protocol level as illustrated in Fig.1. DoS attacks based on *protocol features* take advantage of certain standard

protocol features. For example they exploit the fact that IP source addresses can be spoofed, while others involve attacking DNS cache on name servers.

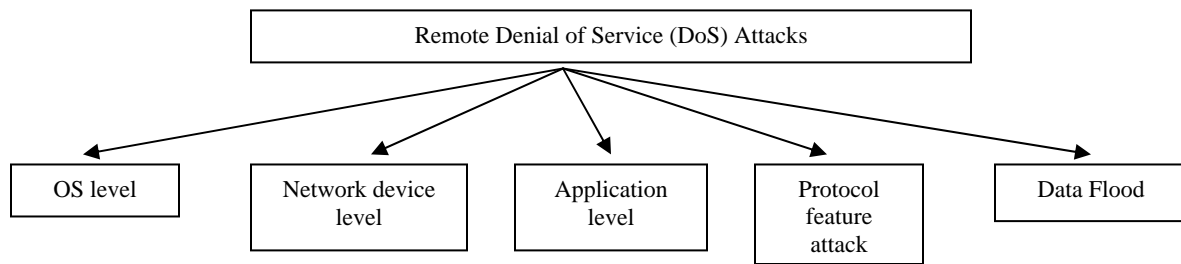


Fig. 1 Classification of remote DoS attacks.

While DDoS is a more sophisticated form of DoS attack that exploits the open resource access model of internet to target its victim. The traffic generated by DDoS attack is so aggregated that it is difficult to distinguish legitimate packets from attack packets. And unfortunately there are no apparent characteristics of DDoS streams that could be directly used for detection and filtering. Now days we are seeing the shift in attackers moving from password cracking attacks towards DDoS attacks. Although here exists no panacea for all flavors of DDoS, there are several countermeasures identified by research community that focus on either making the attack more difficult or making the attacker accountable.

The roadmap for the rest of the paper is as follows. Section 2 investigates the architecture, problem and various DDoS tools used. In section 3 we emphasizes on the classification of DDoS attacks and appropriate defenses that can be deployed to address them. In section 4, we do detailed study of different intrusion detection systems and compare their effectiveness in addressing DoS attacks. Due to recent increase in DDoS threats over web, the domain has huge potential for future work stated in section 5, where we conclude.

2. DDoS attacks

Distributed Denial of Service (DDoS) attack is a simple yet powerful technique to exhaust the computing and communication resources of its victim within very short time with no advance warning. DDoS attack adds the many-to-one dimension to the DoS problem by making the prevention and mitigation of such attacks more difficult and the impact proportionally severe. It exploits the inherent weakness of the internet system architecture, its open resource access model, which ironically, also happens to be its greatest advantage.

What makes DDoS attacks possible?

The opportunity for DDoS attack arises from current internet design of end-to-end paradigm i.e. the intermediate network provides the bare minimum, best effort packet forwarding service, leaving to sender and receiver the deployment of advanced protocols to achieve desired service guarantees such as quality of service, reliable and robust transport or security [2]. The end-to-end paradigm pushes complexity to end hosts, leaving the intermediate network simple and optimized for packet

forwarding. There is nothing that the host or intermediate network can do in stepping and stopping the intruder because internet was never designed for police traffic of packets. More analytically:

Internet security is highly interdependent: Regardless of how well the victim system may be, its susceptibility to DDoS attacks depends on the state of security in the rest of the global network.

Internet resources are limited: No internet host has unlimited resources that sooner or later can be consumed by a sufficient number of users.

Intelligence and resources are not collocated: Most of the intelligence needed for service guarantees is located in end hosts and also the high bandwidth pathways are designed in the intermediate network. As a result, attackers can exploit the abundant resources of unaware hosts in order to flood a victim with messages.

Accountability is not enforced: The presence of IP Spoofing gives attacker powerful mechanism to escape accountability for their actions and sometimes even the means to perpetrate attacks (reflector attacks such as the Smurf attack)

Control is distributed: Every network on internet is running their own local policies defined by its owners, which from security point of view has innumerable implications. There is no way to enforce global deployment of a particular security policy or mechanism. It is often impossible to investigate cross-network traffic behavior because of privacy concerns.

Many against a few: If the resources of attacker are greater than the resources of the victims then the success of the attack is almost definite.

According to [4], DDoS attack uses many computers to launch a coordinated DoS attack against one or more targets. Using client/server technology, the attacker is able to multiply the effectiveness of the DoS significantly by harnessing the resources of multiple unaware accomplice computers which serve as attack platforms.

Components of DDoS attack: DDoS attack is composed of 4 elements as shown in Fig.2:

- The real attacker
- Handlers or masters
- Daemon agents or zombie hosts
- A victim or target host

Phases carried out in DDoS attack:

- *Selection of agents:* Attacker chooses agents that have some vulnerability and abundant resources that will enable them to generate powerful attack streams.
- *Compromise:* Attacker exploits the security hole of agent machines to infect them with the attack code. He also tries to protect the code from discovery and deactivation. When participating in DDoS attack, each agent program utilizes only a small amount of resources (both in memory and bandwidth), so that the users of machines experience minimal change in performance.
- *Communication:* Attacker communicates with the one or more number of handlers to identify which agents are up and running, when to schedule attacks, or when to upgrade agents.

- *Attack*: This is the final phase when the actual attack takes place on victim machine. The parameter such as duration of the attack, type, length, TTL, port numbers etc can be adjusted.

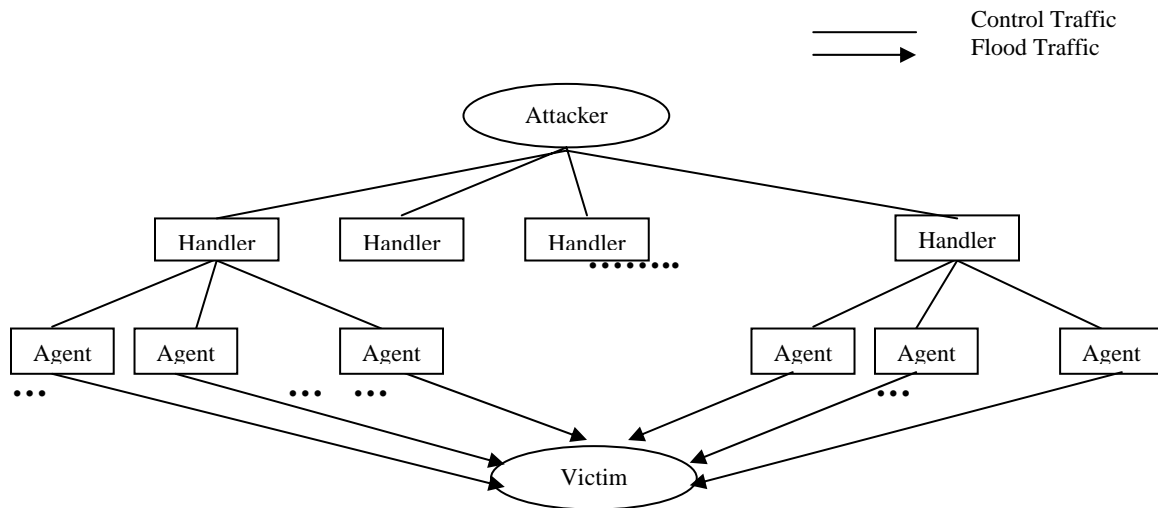


Fig. 2 [2] Architecture of DDoS attacks.

DDoS Tools:

Extremely sophisticated, “*user friendly*” and powerful DDoS toolkits are available to potential attackers increasing the danger of becoming a victim. DDoS attacking programs have very simple logic structures and small memory sizes making them relatively easy to implement and hide. The DDoS tools have almost similar architecture and have minor modifications among themselves. For presentation purposes they can be divided into agent-based and IRC-based DDoS tools [3].

Agent-based DDoS tools

Trinoo: was the first most popular tool used to deploy DDoS attacks. It depletes the bandwidth of the network by launching coordinated UDP flood attacks against one or more IP addresses. It exploits the buffer overrun bug to install trinoo agent.

Tribe Flood Network (TFN): tool provides the attacker with the ability to deploy both bandwidth and resource depletion attacks. It allows UDP, TCP SYN and ICMP flood as well as smurf attacks. Communication between the handler and the daemons is accomplished with ICMP ECHO REPLY packets, which are harder to detect than UDP packets and can even pass firewall systems. TFN has the ability to launch coordinated DoS attacks by generating multiple types of attacks, packets with spoofed IP addresses and also randomize the target ports.

Stacheldraht (German term for “*barbed wire*”): This tool focuses on eliminating weak points in TFN by combining the features of Trinoo (handler/agent architecture) with that of TFN. It has the ability to perform updates on agents automatically. It is capable of launching UDP, TCP SYN, ICMP echo request and ICMP directed broadcast flood attacks.

Mstream: It is a simple point-to-point TCP ACK flooding tool which uses spoofed TCP packets with ACK flag to attack the target. As a side-effect, it can overwhelm the tables used by fast routing

routines in some switches. This program has a feature not found in other DDoS tools. It can inform all connected users of access, successful or not, to the handlers by competing parties. The target gets hit by ACK packets and sends TCP RST to non-existent IP addresses. In turn, routers return “ICMP unreachable” causing more bandwidth starvation.

Shaft: It is a derivative of trinoo tool. It provides UDP, ICMP and TCP flooding attack options. A new feature in Shaft is the ability to switch the handler’s IP address and port in real time during the attack, making difficult for intrusion detection tools to detect an attack.

IRC-based DDoS tools: These tools are more sophisticated as they include more features than that found in agent-based attack tools.

Trinity v3: Besides the up to now well-known UDP, TCP, SYN, TCP ACK, TCP NULL packet floods introduces TCP fragment floods, TCP RST packet floods, TCP random flag packet floods and TCP established floods, while randomizing all 32 bits of the source IP address.

Knight: It is very lightweight and powerful tool that provides SYN attacks, UDP flood attacks and an urgent pointer flooder. It is designed to run on Windows operating system and has features such as an automatic updater via http or ftp, a check sum generator and more. It is typically installed using Trojan horse program called Back Orifice.

3. Classification and Defense Mechanisms for DDoS attacks

DDoS Classification

The variety of known attacks creates the impression that the problem space is very vast, hard to explore and address. [2] Provides a detailed classification of known DDoS attacks to facilitate a global view of the problem and solution space.

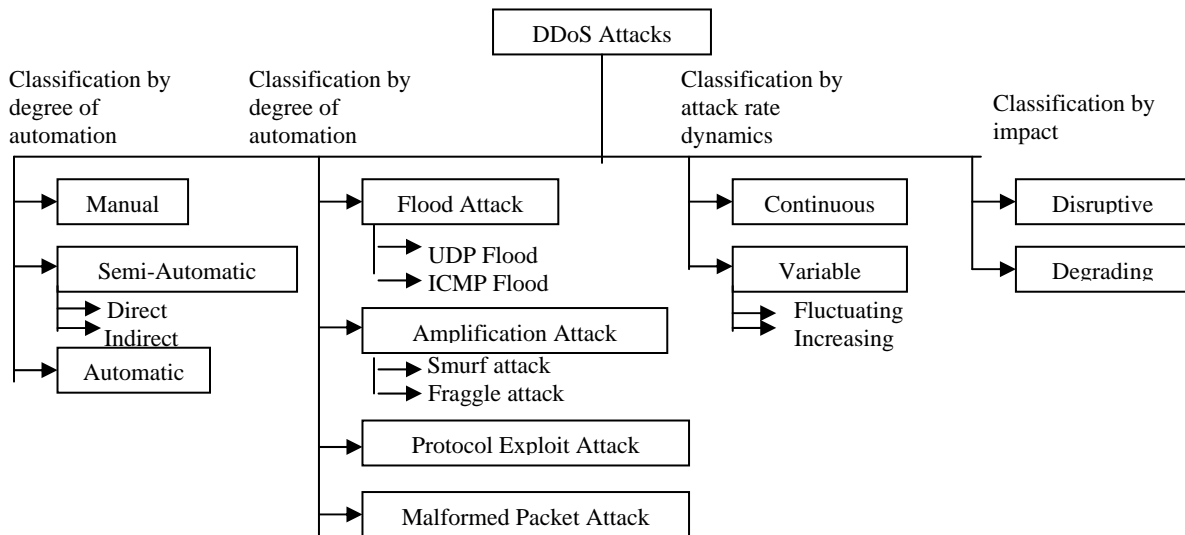


Fig. 3 [2] Classification of DDoS attacks.

It is very important to emphasize commonalities and features of attack strategies, so that researchers can adopt perfect design of countermeasures and also answer many important questions [3]:

- What are the different ways of perpetrating a DDoS attack? Why is DDoS a difficult problem to handle?
- What attacks have been handled effectively by existing defense systems? What attacks still remain unaddressed and why?

Classification by degree of automation:

1. **Manual:** Here the DDoS attack includes the scanning of remote machines for vulnerabilities, breaking into them and installing the attack code.
2. **Semi-automatic:** The attacker scans and compromises the handlers and agents by using automated scripts. It can further be divided into attacks with *direct* and *indirect* communication.

In *direct communication*, agent and handler needs to know each other's identity in order to communicate, that is IP address of handler machines in hard coded. The main drawback being that discovery of one compromised machine can expose the whole DDoS network. While *indirect communication* uses indirection in order to achieve a greater survivability of DDoS attacks. A representative example is IRC-based DDoS attack.

3. **Automatic:** The communication between attacker and agent machines is completely avoided All the features of the attack like type, duration and victim's address are programmed in the attack code, which there by has minimal exposure of attacker and the possibility of revealing his identity is small. The main drawback is that propagation mechanisms usually leave the backdoor to the compromised machine open, making possible future access and modification of the attack code.

Classification by degree of exploited vulnerability:

1. **Flood attack:** The agents send large volumes of IP traffic in order to congest the victim system's bandwidth. Some of the well-known flood attacks are *UDP* and *ICMP* flood attacks.

In *UDP flood* attacks, the UDP packets are sent to either random or specified ports of the victim system. When the victim realizes that there is no application waiting on the port, it generates an ICMP packet of "destination unreachable" to the forged source address. It results in the saturation of the network and depletion of available bandwidth for legitimate service requests to the victim system.

ICMP flood attacks exploit the Internet Control Message Protocol (ICMP), which enable users to send an echo packet to a remote host to check whether it's alive. Here the agents send large volume of ICMP_ECHO_REPLY packets ('ping') to the victim. These packets request reply from the victim, which results in the saturation of the victim's network bandwidth.

2. **Amplification attack:** It exploits the broadcast IP address feature found on most routers to amplify and reflect the attack and send messages to a broadcast IP address. This instructs the routers servicing the packets to send them to all IP addresses within the broadcast address range. This creates malicious traffic and thus reduces the victim system's bandwidth. Some well known amplification attacks are *Smurf* and *fraggle* attacks.

In *Smurf attack*, ICMP echo request traffic is sent with spoofed source address of the target victim to a number of IP broadcast addresses. Most hosts on an IP network will accept ICMP echo requests and reply to source address (Victim). In this attack not only the spoofed address target (the victim) is hurt but also the intermediate broadcast devices (amplifiers). The *fraggle attacks* are similar to Smurf except that they use UDP echo packets instead of ICMP echoes.

3. **Protocol exploit attack:** It exploits some implementation bug or specific feature of some protocol installed on the victim's machine. A good example of this attack is TCP SYN attack. It exploits the inherent weakness of 3-way handshake involved in the TCP handshake. An attacker initiates a SYN flooding attack by sending a large number of SYN packets and never acknowledges any of the replies, essentially leaving server waiting for non-existent ACK's. Other examples are PUSH + ACK, CGI request and authentication server attacks.
4. **Malformed packet attacks:** rely on incorrectly formed IP packets that are sent from agent to the victim's machine. It can be divided into types: IP address and IP packet options attack.

In *IP address attack*, the packet contains the same source and destination IP addresses which results in confusion of the operating system of victim and thus crash it. In *IP packet options attack*, a malformed packet may randomize the optional fields within an IP packet and set all quality of services bits to one, which would result in additional processing time of victim and could lead to crash of system.

Classification by degree of attack rate dynamics:

1. **Continuous rate attack:** is executed with full force and without a break or decrement of force. The impact of such attack is very quick.
2. **Variable rate attack:** avoid detection and immediate response by varying the attack rate. It can be further classified as increasing and fluctuating rate attacks. *Increasing rate attacks* gradually lead to exhaustion of victim's resources, thus delaying detection of attack. While *fluctuating rate attack* has a wavy rate that is defined by the victim's behavior and response to the attack, at times decreasing the rate in order to avoid detection.

Classification by degree of impact:

1. **Disruptive attacks:** leads to complete denial of victim's service to its clients.
2. **Degrading attacks:** consume some portion of a victim's resources which delays detection of attack and at the same time causes immense damage to the victim machine.

DDoS Defense Mechanisms

Due to distributed nature, no common characteristics and deployment of automated DDoS tools, the DDoS attacks are hardest security problems to solve. There are several factors that hinder the advance of DDoS defense research. They can be briefly summarized as[2]:

- Need for distributed response at many points on the Internet.
- Lack of detailed attack information
- Lack of defense system benchmarks
- Difficulty of large-scale testing
- Economic and social factors

We can divide DDoS defense mechanisms based on two different criteria: first classification categorizes DDoS according to *activity deployed*.

- Intrusion Prevention
- Intrusion Detection
- Intrusion Response
- Intrusion Tolerance and Mitigation

The second classification divides according to *location deployment* of attack.

- Victim Network
- Intermediate Network
- Source Network

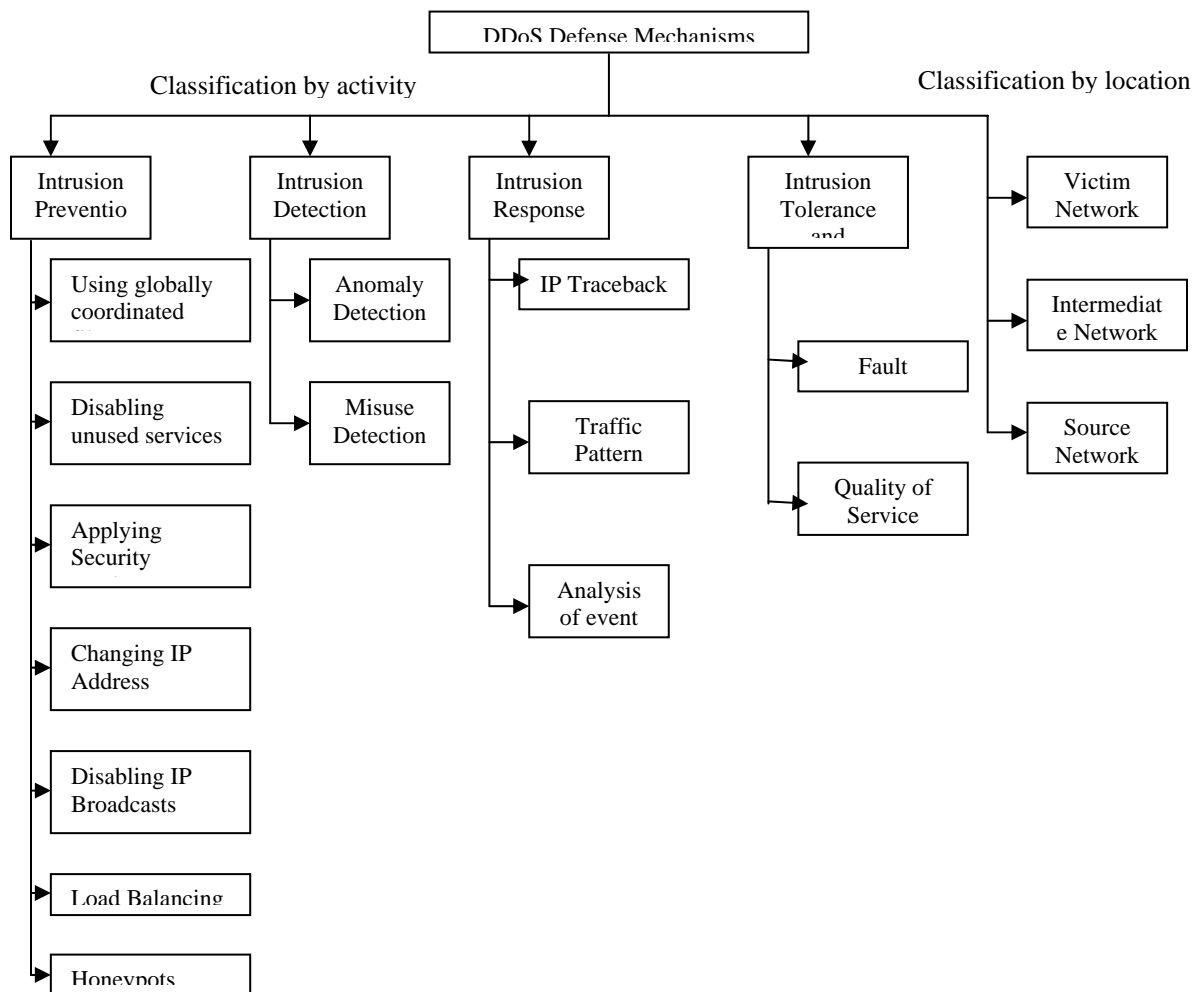


Fig. 4 [3]: Classification of Defense mechanisms for DDoS attacks.

4. Intrusion Detection techniques for DoS

The ultimate goal of any security tool is to protect the data in terms of preventing unauthorized usage (confidentiality), protecting any illegal modification (integrity) and making data reachable and readable by legitimate users (availability). Most of the computer hosts on network deploy security

tools that effectively address the confidentiality and integrity of data, but are unable to address the availability issue caused by attacks such as Denial of Service (DoS) attacks. In this section we will focus on the Intrusion detection techniques to counter Denial of Service attacks.

At an abstract level, an intrusion detection system extracts features, i.e. the individual pieces of evidence, from the system event-level or network packet-level audit data, and uses some modeling and analysis algorithms to reason about the available evidence. Intrusion Detection systems can broadly be classified into misuse detection and anomaly detection. *Misuse detection* uses a priori knowledge on intrusions and tries to detect attacks based on specific signatures or patterns of known attacks. Although they are very accurate in detecting known attacks, their basic drawback is that network attacks are under a continuous evolution and this leads to the need for an up-to-date knowledge base of all attacks. While *anomaly detection* techniques define what is normal and attempt to track deviations from the normal behavior that are considered to be intrusions. Thus they have the advantage of detecting unknown attacks. Now we will discuss in detail three different approaches to intrusion detection systems to counter DoS attacks.

1. Data mining approach

One of the most critical application areas for data mining is intrusion detection systems. A typical data mining approach in intrusion detection system focuses on: feature extraction and construction, customization of (general) algorithms according to semantic information, and optimization of execution efficiency of the output models. Here we will discuss the data mining framework proposed by [10] for intrusion detection models and observe their efficiency in handling DoS attacks.

Data mining techniques exploits certain characteristics of audit data in order to construct efficient detection models. *Processing*: In the first step it processes “raw” audit data to a suitable form like ASCII tabular data with attributes. The records are ordered by timestamps and are summarizes all data belonging to same connection. The key objective here is to *extract* and *construct* appropriate features so that effective detection models can be constructed. *Customizing*: Since the audit data contains rich network and systems semantics, the second step tries to customize the general algorithms to incorporate domain knowledge so that only the *relevant patterns* are computed. *Computing*: Since the audit data is high-speed and high-volume streaming data, it requires the run-time execution of ID systems to be very efficient. Otherwise, the long delay in data analysis simply presents a time window for attacks to succeed.

There are several types of algorithms [9] that are particularly useful for mining audit data:

Classification: technique maps a data item into one of several predefined categories. These algorithms normally output “*classifiers*,” for example, in the form of decision trees or rules. An ideal application in intrusion detection would be to gather sufficient “normal” and “abnormal” audit data for a user or a program, then apply a classification algorithm to learn a classifier that labels new unseen audit data as belonging to the normal class or the abnormal class.

Link analysis: technique determines relations between different fields in the database records. Correlations of system features in audit data, for example, the correlation between *command* and

argument in the shell command history data of a user, can serve as the basis for constructing normal usage profiles.

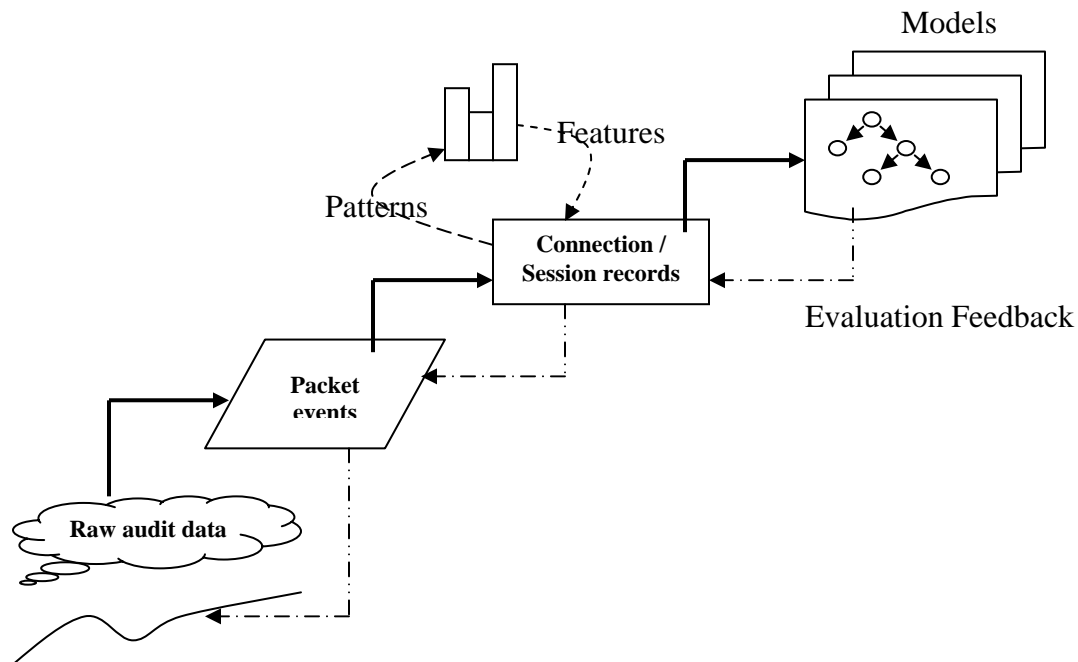


Fig. 5: The data mining process of building ID systems.

Sequence analysis: models sequential patterns. These algorithms can discover what time-based sequence of audit events frequently occur together. These frequent event patterns provide guidelines for incorporating temporal statistical measures into intrusion detection models.

We will now study the data mining approach that [10] proposed:

A framework: In the first step, they processed raw audit data and summarized them into network connection records containing number of basic features like: Timestamp, Duration, Source and Destination IP address and Port numbers, Protocol Type and an error condition flag.

Pattern Mining and Comparison: Secondly, they computed the association rules and frequent episodes to capture intra and inter-audit record patterns. The basic association rules and frequent episodes do not consider any domain knowledge. So while customizing these algorithms, they incorporated schema-level knowledge into interestingness measures. First, they incorporated partial “*order of importance*” among attributes i.e. while some attributes are essential while others provide auxiliary information. For example, a network connection can be uniquely identified by the combination of its start time, source host, source port, destination host and service. Another interesting schema-level information is that some attributes can be *references* of other attributes.

The patterns obtained from extended algorithms were then used for feature construction. The idea they exploited was first convert patterns into numbers in such a way that “*similar*” patterns are mapped to “*closer*” numbers. A comparison procedure then computes the “*intrusion score*” for each

pattern from the intrusion dataset, which is its lowest distance score against all patterns from the normal dataset, and outputs the user-specified top percentage patterns that have highest intrusion scores as the “intrusion only” patterns. As an example, let us consider the “SYN flood” attack where the attacker uses many spoofed source addresses to send a lot of S0 connections (only the SYN packet, the connection request, is sent) to a port (e.g. http) of the victim host in very short time span. The victim’s connection buffer is filled by mounting such a Denial of Service attack.

Feature Construction: Thirdly, each of the intrusion patterns is used as a guideline for adding additional features into the connection records to rebuild better classification models. They used the following automatic procedure for parsing the *frequent episode* and constructing features [10]:

- Assume F_0 (e.g. `dst_host`) is used as the reference feature and width of episode is w seconds.
- Add the following features that examine only the connections in the past w seconds that share the same value in F_0 as the current connection:
 - A feature that computes “the count of these connections”.
 - Let F_1 be service, `src_dst` or `dst_host` other than F_0 (i.e. F_1 is an essential feature). If same F_1 value (e.g. “http”) is in all the item sets of the episode, add a feature that computes “the percentage of connections that share the same F_1 value as the current connection”; otherwise, add a feature that computes “the percentage of different values of F_1 ”.
 - Let V_2 be a value (e.g. “S0”) of a feature F_2 (e.g., flag) other than F_0 and F_1 (i.e. V_2 is in all the item sets of the episode, add a feature that computes “percentage of connections that have the same value V_2 ”; otherwise, if F_2 is a numerical feature, add a feature that computes “the average of the F_2 values”).

The above procedure parses a frequent episode and uses three operators *count*, *percent* and *average* to construct statistical features. The essential features describe the *anatomy* of an intrusion, for example, “the same service (i.e. port) is targeted”. The actual values e.g. “http” is often not important because the same attack method can be applied to different targets, e.g. “ftp”. On the other hand, the actual non-essential feature values, e.g. flag = S0, often indicate the invariant of an intrusion because they summarize the connection behavior according to the network protocols.

Constructing Efficient Models: Finally, for providing run-time execution efficiency multiple models each with different computation cost and detection accuracy was used. They incorporated the idea of executing lighter weight detection model first and if the desired prediction accuracy is not attained, the more time-consuming modification will then be activated.

They partitioned features into three relative cost levels. *Level 1* features e.g. service, are computed using at most the first three packets (or events) of a connection (or host session) and these require simple recording. *Level 2* features are computed in the middle or near the end of a connection using information of the current connection only. These require simple book keeping. *Level 3* features are computed using information from all connections within a given time window of the current connection. They are often computed as some aggregates of the level 1 and 2 features. They then assigned qualitative values to these cost levels, based on run-time measurements with a prototype system using Network Flight Recorder simulator.

Because all level 3 features required iteration through the entire set of connections in a given time window, they can all be computed at the same time, in a single iteration. This addressed the computational cost when multiple level 3 features are computed for analysis of a given connection.

Experimental results

The detection model constructed under this approach had an acceptable false alarm rate (under 0.02%), but its detection rate was below 70% which is not acceptable in real life detection systems.

2. Anomaly detection using Emergent Self-Organizing Maps

In this section we discuss a network-based anomaly detection method to detect DoS attacks. The anomaly detection approach is based on a class of neural networks known as Kohonen's Self Organizing Maps (KSOMs). In order to achieve better results in detection of intrusions, the approach combines machine learning and information visualization techniques [1].

According to [9], "*Emergence is the ability of a system to produce a phenomenon on a new, higher level*" and in order to achieve emergence the existence and cooperation of a great number of elementary processes is necessary. One of the basic disadvantages of SOM maps that they noticed in KSOM was that they were limited to a few neurons. The small number of neurons of KSOM's did not permit them to show emergence.

However, Emergent Self-Organizing Maps can be expanded from some thousands to tens of thousands of neurons. The large number of neurons in ESOM is necessary in order to achieve emergence. This cooperation of such a big number of neurons helps in observing the systems in a higher level of overall structures, disregarding the elementary ones and allows considering structures that other would be hidden. Of many possible methods of visualization (such as distance-based (U-Matrix), density-based (P-Matrix), distance and density based (U*-Matrix)), ESOM used distance-based method.

Steps taken by [7] to perform analysis of network traffic:

a) ***Selection of features from KDD Dataset:*** The features of network traffic were selected in the form that is easily processed by ESOM and are representative of network activity in order to be able to distinguish normal and abnormal activity. The most important features for the detection of Denial of Service attacks from the 41 features of KDD data were:

- o Duration
- o Source bytes
- o Destination bytes
- o Count
- o Same service rate
- o Connections with SYN errors
- o Connections-Same service-SYN errors
- o Destination-Host-SYN error rate
- o Destination-Host-Same-Service error rate
- o Destination-Host-Same-Service error rate

b) ***Emergent KSOM procedure***

- i. The Emergent SOMs were trained with logs of network traffic
 - a. The random weights w_{ij} are initialized with small random values.
 - b. Use an input pattern x .
 - c. Calculate the Euclidean distance between data sample x and each neuron weight w_{ij} . The winner (Best Matching Unit) is chosen as $o(x)$:

$$o(x) = \arg \min_j \|x - w_{ij}\|, j = 1, 2, \dots, k$$

- d. In order to achieve the topological mapping, all the weights in the neighborhood are adjusted, depending on their distance from the winning neuron according to the following equation:

$$\forall j: w_{ij}(t) = w_{ij}(t-1) + a(t) n(t') * (x_i(t) - w_{ij}(t-1))$$

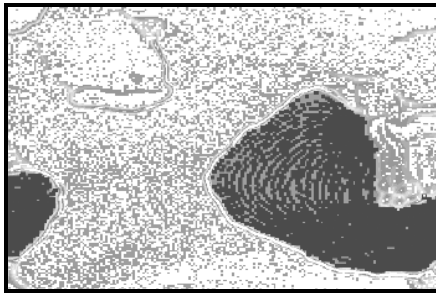
where a is the learning rate and n the neighborhood function and t' , the time that was spent in the current context.

- e. Steps b, c, d were repeated until convergence.

- ii. ESOM U-Matrices were used to perform DoS detection. The U-Matrix of the trained dataset is divided into valleys that represent borders between clusters.
- iii. Depending on the position of the best match of an input data point that characterizes a connection, this point may belong to a valley (cluster (normal or attack behavior)) or this point may not be classified if its best match belongs to a hill (boundary)

c) *Experimental Results*

The dataset used was of various sizes including both normal and DoS attack data. The number of input data was over 10,000 records. The ESOM of a trained dataset can be depicted in Fig.6. The training data has been divided into two classes: normal data class (dark color) and DoS data class (light color)



Input data set a set of small dots belongs to light color (DoS attack)

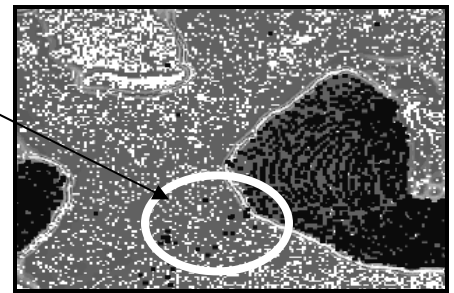


Fig. 6 [7] Emergent SOM U-Matrix of train dataset

Fig. 7 [7] Emergent SOM U-Matrix of test data

Fig. 7 depicts the test dataset for the corresponding trained dataset. The circle represents the input dataset that was tested and was found to belong to light color and hence DoS attack. The efficiency of this approach was measured using two parameters: *detection rate* and *false alarm rate*.

$$\text{Detection rate} = \frac{TP}{TP + FN}$$

$$\text{False alarm rate} = \frac{FP}{TN + FP}$$

Where TP is number of true positives (attack logs classified as attacks), TN is number of true negatives (normal logs classified as normal), FP is the number of false positives (normal logs classified as attacks) and FN is number of false negatives (attack logs classified as normal). The most effective approach should reduce as much as possible the False alarm rate and at the same time increase the detection rate. The evaluation experiments provided detection rate for DoS attacks that ranged between 98.3% and 99.81% and false alarm rate that ranged between 2.0% to 0.1%.

3. A hybrid Intelligent ID using neural network + fuzzy techniques.

In [8], a hybrid intelligent system was used to recognize novel attacks such as Distributed Denial of Service (DDoS) attacks and to learn the attack definitions and the description of attack properties. This approach applies neural network technique to learn the attack definitions and fuzzy inference approach to describe the relations of attack properties for recognition.

Architecture of hybrid intelligent ID

The architecture of hybrid intelligent ID system consists of three modules:

- i. A *data module* which collects the network data and computes the input features of them.
- ii. A *multi-layered feed-forward* neural network which is trained to explore the attack definitions from the input data. The output of this module is passed to the third module.
- iii. *Fuzzy inference module* based on empirical prior knowledge about the relations of attack properties is utilized to provide the reasoning results for recognition. It also provides the mechanism for explanation of the reasoning process.

Procedure involved

i. Collecting the training and test data

The packets both belonging to normal and attack behavior are sampled in user defined unit time. Four types of DDoS attacks were considered: ICMP smurf attack (BLOOP), ICMP flood attack (XICMP), UDP flood attack (PANTHER), UDP flood attack (OVERDROP) and TCP SYN flood attack (SYNK4).

In the experiments the SYNK4 attack data was not trained by the system, as these were regarded as novel attacks. The input features of the collected data were computed from statistical viewpoint. Seven different statistics definitions were utilized for the computation of input features: out_ip_count, port_source_count, port_destination_count, udp_count, icmp_count, tcp_count, inip_count. The input features were obtained by the *normalization* of these seven statistics definitions as:

$$X_{\text{normal}} = (X - M_X) \div \text{STD}_X$$

where X is one of the statistics definitions, M_X is the mean of input variable X and STD_X is the standard deviation of input variable X.

ii. Training feed-forward neural network

The neural network is composed of a 3-layered feed-forward network. There are 7 input neurons corresponding to 7 statistics definitions namely: out_ip_count, port_source_count, port_destination_count, udp_count, icmp_count, tcp_count, inip_count. 5 hidden neurons and 4

output neurons are variables U Value, I Value, T Value and IP Value which represent the four types of attacks.

The feed-forward neural network is trained by the *error back-propagation* algorithm. The activation function used is sigmoid function.

iii. *The reasoning process.*

The four outputs from neural network serve as the input to the fuzzy inference module. Fuzzy inference module utilizes the membership functions and fuzzy rules to get the RESULT.

Variables	Membership function	Categories
U Value	Tri_MF	1. Low 2. High
I Value	Tri_MF	1. Low 2. High
T Value	Tri_MF	1. Low 2. High
IP Value	Tri_MF	1. Low 2. High
RESULT	Result_mf	1. BLOOP 2. XICMP 3. PANTHER 4. OVERDRO 5. NORMAL 6. UNKNOWN

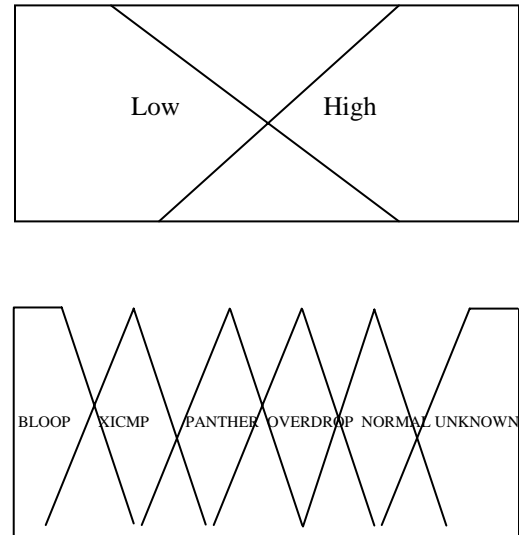


Fig.9 [8] Tri_MF and Result_mf membership function

Fig. 8 [8] Membership functions defined for four variables

The SYNK4 attack was not one of the output variables RESULT, since it was regarded as the novel attack. If the inference result for SYNK4 input data is UNKNOWN, the system will succeed to recognize these novel attacks and vice-versa. The inference results are made by the fuzzy reasoning engine using the inference rules about the relations of attack properties. The rules are constructed on the basis of *empirical prior knowledge* about attacks. The fuzzy inference rules were devised as follows:

R1: If I Value is very high AND IP Value is very low AND T Value is very low and U Value is very low, then RESULT is BLOOP.

R2: If I Value is very high AND IP Value is very high AND T Value is very low and U Value is very low, then RESULT is XICMP.

R3: If I Value is very high AND IP Value is very high AND T Value is very low and U Value is very high, then RESULT is PANTHER.

R4: If I Value is very low AND IP Value is very low AND T Value is very low and U Value is very high, then RESULT is OVERDRO.

R5: If I Value is very low AND IP Value is very high AND T Value is very high and U Value is very low, then RESULT is NORMAL.

R6: If no rule fired, then RESULT is UNKNOWN.

After recognizing the novel attacks, they *re-trained* the feed-forward neural network using error back-propagation algorithm. The inference rule

R+: If I Value is very low AND IP Value is very low and T value is very high AND U Value is very low, then RESULT is SYNK4, was added into the fuzzy inference engine. The SYNK4 was inserted category between NORMAL and UNKNOWN in the Result_mf membership function.

Experimental results

The detection model was tested with 51 test samples in which 3 sample were recognized as NORMAL while 48 samples were recognized as UNKNOWN. The error rate was 5.88% with most of the novel attacks detected successfully.

	Data mining	Emergent SOM	Hybrid approach
Types of attacks	Depends on audit data	Efficient in detecting Almost all attacks	Inference rules defines the Attacks that can be detected
Input set	Large amount of input set required	Very huge amount of Training set required	Not much input data required
Novel attacks	Cannot detect with preciseness	Very efficient	Very efficient
False Alarm	Low	Very Low	Low
Performance	High processing Speed: Efficient run-time computation models	High Computational Cost involved in training	Very fast, does not require any special tool
Accuracy	Very Low	Very High	High
Automation	Fully automated	Semi-automated	Manual intervention at Fuzzy interface

Comparison of different approaches on different parameters

5. Conclusion

The motivation for carrying out such an in-depth study of DDoS attacks was their ability to harness the resources of multiple unwitting accomplice computers, which serve as attack platforms. One of the benefits of classifying DDoS attacks and defense mechanisms is that effective communication and cooperation between researchers can be achieved which will help in identifying additional

weakness which might be exploited in future. There is also need for research community to develop common metrics and benchmarks for DDoS defense evaluation.

We also studied the different intrusion detection systems that can be deployed in the network to counter such novel attacks by learning their behavior over a period of time. The *data mining* approach constructed very accurate detection models based on audit data but failed to address novel attacks such as DoS. Merging audit data from different sites is still not possible due to legal constraints, so there is a need for *correlation algorithms* capable of merging alarms (i.e. detection outcomes) from different sources. The *emergent self organizing maps* were very powerful in producing efficient results with accuracy close to 99%. Its prime disadvantage of high computational cost was balanced by performing training process only once. The *hybrid intelligent system* incorporated the advantages of neural-network learning and fuzzy inference to address the problem of recognizing novel attacks accurately and efficiently.

The future work can be focused on developing hybrid intelligent systems which dynamically generates fuzzy inference rules and integrates neural network module and fuzzy inference modules. On detecting an alarm it should also be capable of generating dynamic policies which can be automatically enforced on the system to protect the legitimacy. DDoS attacks are not of concern only in wired networks but also in wireless infrastructures. Geng et al. [6] proposed a conceptual model for defending against DDoS attacks on the wireless networks, which incorporates both cooperative technological solutions and economic incentive mechanisms. Unlike DoS attacks which succeed by saturating the resources of a system, there is an urgent need to address *Denial of Information* (DoI [12]) attacks that target the inability of humans to handle information beyond a certain limit by flooding a particular service with massive syntactically correct requests.

References

- [1] Khaled Labib, V. Rao Vemuri, “*NSOM: A Real-time Network-Based Intrusion Detection system Using Self-Organizing Maps*”, 2002, Networks Security.
- [2] Jelena Mirkovic, Peter Reiher, “*A Taxonomy of DDoS attack and DDoS Defense Mechanisms*”, 2004, ACM SIGCOMM Computer Communications Review.
- [3] Christos Douligeris, Aikaterini Mitrokotsa, “*DDoS attacks and defense mechanisms: classification and state-of-the-art*”, 2003, Elsevier Computer Networks 44 643-666.
- [4] L.D. Stein, J.N. Stewart, The World Wide Web Security FAQ, version 1.7, February 23rd, 2003, < <http://www.w3.org/Security/faq/>>
- [5] D. Karig, R. Lee, “*Remote Denial of Service Attacks and countermeasures*”, Department of Electrical Engineering, Princeton University, Technical Report CE-L2001-002, October 2001.
- [6] X. Geng, Y. Huang, A.B. Whinston, “*Defending wireless infrastructure against the challenge of DDoS attacks*”, Mobile Networks and Applications 7 (3) (2002) 213–223.
- [7] Aikaterini Mitrokotsa et al, “*Detecting Denial of Service Attacks using Emergent Self-Organizing Maps*”, 0-7803-9314-7/05/2005 IEEE
- [8] Dwen-Ren Tsai, Wen-Pin Tai, and Chi-Fang Chang, “*A Hybrid Intelligent Intrusion Detection System to Recognize Novel Attacks*” 0-7803-7882-2/03/2003/IEEE
- [9] A. Ultsch, “*Data mining and knowledge discovery with emergent SOFMs for Multivariate Time Series*”, in Kohonen Maps (1999), pp. 33-46
- [10] W. Lee, W. Fan, “*Mining System Audit Data: Opportunities and Challenges*”, ACM SIGMOD, 2001, Vol.30, No.4.

- [11] W. Lee and S.J. Stolfo, “*A Framework for Constructing Features and Models for ID Systems*”, *ACM Transactions on Information and System Security*, 1094-9224/00/1100-0227
- [12] M. Ahamad, W. Lee, L Liu et al, “*Guarding the Next Internet Frontier: Countering Denial of Information Attacks*”, 2002, ACM New Security Paradigms Workshop, 1-58113-598
- [13] M. Roughan, S. Sen et al, “*Class-of-Service Mapping for QoS: A Statistical Signature-based approach to IP Traffic Classification*”, 2004 ACM IMC 1-58113-821-0