Protection Against Client Based HTTP Attacks on Web Proxy by TBAD and TSL Behavior

Jayasree P
Department of Computer Science and Engineering
Sri Sai College of Engineering and Technology, Badhani Pathankot Punjab
Email: jayasreesaji@gmail.com

Abstract *- Distributed Denial-of-Service (DDoS) attacks continue to be key threat to Internet applications. In such attacks, a set of attackers generates a huge amount of traffic, saturating the victim's network, and causing significant damage. In computer networks, a proxy server is a server that acts as an intermediary for requests from clients seeking resources from other servers. Overlay networks called proxy networks being to protect applications against such DDoS attacks. Today, most proxies are web proxies, facilitating access to content on the World Wide Web.Client can access web server through different proxies. Web servers have no technique for identifying malicious client. This paper proposed to resist the web-proxy based DDoS attacks. Here Hidden Semi-Markov Model (HsMM) parameterized using Gaussian Gamma features is to model web access behavior sequence and find which proxy cause attack using temporal and spatial access behavior. Thus by Gaussian Gamma features in HsMM reduces parametric computational complexity. A Soft control scheme for behavioral reshaping reshapes suspicious request sequence according to normal behavior. In this scheme both the long term and short term behavior of web proxies were analyzed. For detecting spatial and temporal attack behavior a threshold based algorithm called Threshold based attack detection – TBAD for detecting actual attacking client rather than an innocent proxy by adding custom headers in http protocol . A session hijacking method includes a source routing techniques also implemented to find the session hijacking attacks. Thus proceeding with these measures the QoS of legitimate users can be protected.*

Key words: *DDoS Attacks, HTTP Attacks, Temporal & Spatial Locality, Web Proxy*

1. INTRODUCTION

Distributed denial of service (DDoS) attacks, which are a major threat on the Internet, have recently become more sophisticated as a result of their ability to exploit application layer vulnerabilities. By using an HTTP request, an attacker has a legitimate path to your Web server and therefore can easily bypass firewalls and other security measures to initiate an attack. In computer networks, a proxy server is a server that acts as an intermediary for requests from clients seeking resources from other servers. Overlay networks called proxy networks being to protect applications against such DoS attacks. Today, most proxies are web proxies, facilitatingaccess to content on the World Wide Web.

Client can access web server through different proxies. Web servers have no technique for identifying malicious client. This paper proposed to resist the web-proxy based DDoS attacks. Here Hidden Semi-Markov Model (HsMM) parameterized using Gaussian Gamma features is to model web access behavior sequence and find which proxy cause attack using temporal and spatial access behavior. Thus by Gaussian Gamma features in HsMM reduces parametric computational

complexity. In existing system a proxy based approach is used to analyze the temporal and spatial behavior. It includes creation of web server, proxy server and then creates a request pattern including attack sequences.

In our proposed system a client based attack detection using the temporal and spatial behavior of request identified using a training algorithm called TBAD Training Algorithm. This algorithm provides the technique to find out the attacking client based on the request from the user. It don't consider about the client. A Soft control scheme for behavioral reshaping reshapes suspicious request sequence according to normal behavior. An http protocol change will make in the proposed system to find out the actual attacker. A source routing technique is employed to find out the various Session Hijacking Attacks. In summary this paper deals with the client based http attacks on web proxy.

2. HTTP ATTACKS

In the current scenario website and web applications rapidly growing. Complex business applications are now delivered over the web .Web based attacks are considered by security experts to be the greatest and oftentimes the least understood of all risks related to confidentiality, availability, and integrity. The purpose of a web based attack is significantly different then other attacks; in most traditional penetration testing exercises a network or host is the target of attack. Web based attacks focus on an application itself and functions on Application Layer of the OSI. Most of all attacks occur at the application layer. One approach for dealing with HTTP-based attacks is to identify malicious code in incoming HTTP requests and eliminates bad requests before they are processed. Different types of http (web) attacks:

Cross Site Scripting (XSS): Cross-Site Scripting attacks are a type of injection problem, in which malicious scripts are injected into the otherwise benign and trusted web sites. Cross-site scripting (XSS) attacks occur when an attacker uses a web application to send malicious code, generally in the form of a browser side script, to a different end user. Flaws that allow these attacks to succeed are quite widespread and occur anywhere a web application uses input from a user in the output it generates without validating or encoding it. An attacker can use XSS to send a malicious script to an unsuspecting user. The end user's browser has no way to know that the script should not be trusted, and will execute the script.

SQL Injection: A SQL injection attack consists of insertion or "injection" of a SQL query via the input data from the client to the application. A successful SQL injection exploit can read sensitive data from the database, modify database data (Insert/Update/Delete), execute administration operations on the database (such as shutdown the DBMS), recover the content of a given file present on the DBMS file system and in some cases issue commands to the operating system. SQL injection attacks are a type of injection attack, in which SQL commands are injected into data-plane input in order to effect the execution of predefined SQL commands.

Blind SQL Injection: When an attacker executes SQL Injection attacks, sometimes the server responds with error messages from the database server complaining that the SQL Query's syntax is incorrect. Blind SQL injection is identical to normal SQL Injection except that when an attacker attempts to exploit an application, rather than getting a useful error message, they get a generic page specified by the developer instead. This makes exploiting a potential SQL Injection attack more difficult but not impossible .

Server Side Includes (SSI): SSIs are directives present on Web applications used to

feed an HTML page with dynamic contents. They are similar to CGIs, except that SSIs are used to execute some actions before the current page is loaded or while the page is being visualized. In order to do so, the web server analyzes SSI before supplying the page to the user. The Server-Side Includes attack allows the exploitation of a web application by injecting scripts in HTML pages or executing arbitrary codes remotely. It can be exploited through manipulation of SSI in use in the application or force its use through user input fields.

Identity Spoofing (IP Address Spoofing): Most networks and operating systems use the IP address of a computer to identify a valid entity. In certain cases, it is possible for an IP address to be falsely assumed identity spoofing. An attacker might also use special programs to construct IP packets that appear to originate from valid addresses inside the corporate intranet. After gaining access to the network with a valid IP address, the attacker can modify, reroute, or delete your data

3. WEB PROXY

In computer networks, a proxy server is a server that may be a computer system or an application that acts as an intermediary for requests from clients seeking resources from other servers. A client connects to the proxy server, requesting some service, such as a file, connection, web page, or other resource available from a different server and the proxy server evaluates the request as a way to simplify and control its complexity. Today, most proxies are web proxies, facilitating access to content on the World Wide Web.

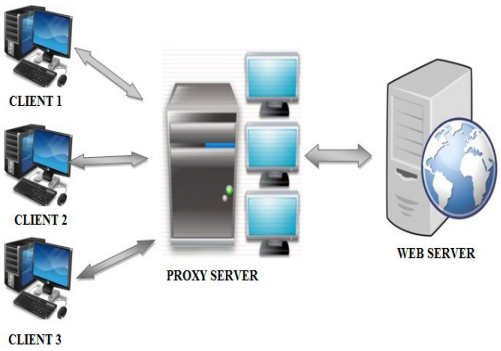

Fig 1.1 Proxy –Web Server Communication

A Web proxy may attacked in two scenarios: First attacker sends attack requests to a Web proxy and forces it to forward the attack requests to the origin server; Second Attacker disconnects connections between itself and the proxy. In Step 1, two methods can be used to penetrate through the Web proxies: requesting dynamic documents or setting “Cache-Control: no-cache” in the headers of HTTP requests. Repeating these steps, a single host can simultaneously trigger a lot of Web proxies to attack a Web server without the need of invading them.

The attraction of an attack lies in three aspects: It enables the attacking host to break through the client side restrictions by connecting different Web proxies via HTTP protocols; Resisting such an attack by the mid Web proxies is not a practical approach, due to lack of cooperation mechanisms between server and proxies, in particular those uncontrollable private proxies. Such an attack may confuse most of the existing detection systems designed for the traditional DDoS attacks due to two reasons: first, the origin server cannot directly observe and diagnose the terminal hosts shielded by the hierarchical proxy system; second, the attack traffic is mixed with the regular client-to-proxy traffic by each proxy that forwards the traffic.

4. TEMPORAL AND SPATIAL LOCALITY

Temporal and spatial locality analysis used to extract the proxy to server behavior. Temporal locality of reference has been widely applied in many fields, for example, program behavior reference pattern of Web access and Web proxy cache replacement strategy. Temporal locality

refers to the property that a referencing behavior in the recent past is a good predictor of the referencing behavior to be seen in the near future, whereas the resource popularity metric only represent the frequency of the requests without indicating the correlation between a reference to a document and the time since it was last accessed. Here, we resort to the concept of stack distance. The files are assumed to be placed on a stack such that, whenever a file f is requested, it is either pulled from its position in the stack and placed on the top, or simply added to the stack if the file is not yet in the stack. The stack distance for the request is then the distance of f from the top in the former case or undefined in the latter case.

Spatial locality refers to the property that objects neighboring an object frequently accessed in the past are likely to be accessed in the future. For example, when a home page is requested, all its embedded objects are likely to be accessed at the same time. Because spatial locality indicates correlation among a cluster of HTTP requests, capturing spatial locality can help mine the access behavior of Web proxies. Different from, here a new method issued to quantize the spatial locality. In the final aggregated proxy-to-server traffic, there is no obvious difference between the normal traffic and the attack traffic except their underlying purposes. Thus, the victim server is hard to accurately identify and filter the attack requests.

The Web proxy-based HTTP attack is more flexible and covert than most of existing DDoS attacks. The difficulty of detection lies in three aspects: First, Real attacking hosts are unobservable to the origin server since they are shielded by the hierarchical Web proxies; second, A Web proxy may be passively involved in an attack event and may unconsciously act as an attacker. Third , Observed from the victim server, both legal and illegal traffic comes from the same sources. Although most of the large-scale official proxies are usually configured to have high security, they cannot avoid being abused for the proxy base attacks Motivated by these issues, a novel resisting scheme is proposed to protect the origin server from Web proxy-based HTTP attacks in this work.

The proposed scheme is based on network behavior analysis. It maps a Web proxy's access behavior to a hidden semi-Markov model (HsMM) which is a typical double stochastic processes model. The output process of an HsMM profiles the observable varying process of a proxy-to-server traffic. The hidden semi-Markov chain of an HsMM describes the transformation of a proxy's internal behavior states, which can be considered as the intrinsic driving mechanism of a proxy to server traffic. Based on this behavior model, detecting the abnormality of a Web proxy can be achieved by measuring the deviation between an observed behavior and the Web proxy's historical behavior profile. Long-term and short term behavior assessment methods are proposed. Long-term behavior assessment issues warnings on a large scale, while short-term behavior assessment locates abnormal request sequences embedded in the proxy-to-server traffic. A new "soft-control" scheme reshapes the suspicious sequences according to the profile of a proxy's historical behavior. It converts a suspicious sequence into a relatively normal one by partly discarding its most likely malicious requests instead of denying the entire sequence. Thus, it can protect the HTTP requests of legitimate users.

5. RELATED WORK

Temporal locality and its impact on web proxy cache performance [1] deals with temporal locality characteristics present in document referencing behavior at Web proxies and the impact of this temporal locality on document caching. First, *drift* measures are developed to characterize how the popularity profile of "hot" documents. Second, a measure of short-term temporal locality is developed that characterizes the relationship between recent-

past and near-future document references. This paper present impact of temporal locality of request pattern in cache performance. But proposed paper deals with both temporal and locality. Here behavior analysis used for improving cache performance, and used at proxy side. But in my work it is used for anomaly detection and used at server side.

In [2] A Lightweight Mechanism to Mitigate Application Layer DDoS Attacks attack refers to the attempt to prevent a server from offering services to its legitimate users, typically by sending requests to exhaust the server's resources, e.g. bandwidth or processing power. This system calculates trust behavior of users instead of request. Bases on these trust value, server manage next request from same user .Same detection will be on the proposed system but it based on request not the user.

In[3] Anomaly-based network intrusion detection: Techniques, Systems and Challenges deals with the intrusion detection behavior**.** In this scenario, anomaly-based network intrusion detection techniques are a valuable technology to protect target systems and networks against malicious activities. This deals with well-known anomaly based intrusion detection techniques. Intrusion Detection Systems (IDS) are security tools that, like other measures such as antivirus software, firewalls and access control schemes, are intended to strengthen the security of information and communication systems. Our proposed system also deals with detection of anomalies by using Hidden Markov Model. This system considers only proxy behavior. But our proposed system considers the web browsing behavior of client also.

Sequence-order-independent network profiling for detecting application layer DDoS attacks[4] deals on Distributed denial of service (DDoS) attacks, which are a major threat on the Internet, have recently become more sophisticated as a result of their ability to exploit application layer. Comparing with proposed system this system detects DDoS attack. With the profiling of web browsing behavior, the sequence order of web page requests can be used for detecting the application-layer DDoS (App-DDoS) attacks. This system considers web browser behavior the proposed system deals with web proxy. This system consider sequential behavior, our paper consider both temporal and spatial behavior.

In [5] Traceback of DDoS attacks using entropy variations deals with the source of Distributed Denial-of-Service (DDoS) attacks in the Internet are a big challenge. Comparing with proposed system this system present a detection mechanism for DDOS attacks using trace back methods. This system deals with a generalized detection not consider special behavior of proxy. But the proposed system considers the web proxy behavior.

The use of entropy based detection and chain model present in this paper can be used for Hidden Markov Model in my proposed work. Discriminating DDoS attacks from flash crowds using flow correlation coefficient [6] presents a flow similarity based approach is taken to discriminate DDoS attacks from flash crowds. Only difference from proposed method that it use flow from proxy. This system uses the HsMM technique. In the proposed system same HsMM technique with Gaussian Gamma parameters is employed. In [7] Low-rate DDoS attacks detection and traceback by using new information metrics deals with DDoS attack in the distributed, cooperative environment. The anomaly-based detection metric typically models the normal network (traffic) behavior and deploys it to compare differences with incoming network behavior. But proposed system considers the web proxy behavior.

In A Large-Scale Hidden Semi-Markov Model for Anomaly Detection on User Browsing Behaviors [8] deals with Distributed denial of service .Comparing with proposed system this system provides Hidden Semi-

Markov Model for Anomaly Detection on User Browsing Behaviors. But proposed method implements same work But anomalies in Web proxy not in individual browser.

Measuring the Normality of Web Proxies Behavior Based on Locality Principles deals on web proxy locality behavior [9]**.**Web Proxy and cache play important roles in the modern Internet. Comparing with proposed system this system proposed a method for finding malicious web request by using locality behavior of Proxy deals on locality. But not considering spatial behavior. In Monitoring the Application-Layer DDoS Attacks for Popular Websites [10] deals with continuous critical threat to the Internet. Derived from the low layers, new application-layer based DDoS attacks utilizing legitimate HTTP requests to overwhelm victim resources are more undetectable.. This system introduces a scheme to capture the spatial-temporal patterns of a normal flash crowd event and to implement the App-DDoS attacks detection. This system also a provide Hidden semi Marko Model for detecting Application Layer DDOS attack at Popular website. Same HsMM model used in the proposed system. But in proposed system it is used to implement in web Proxy to analyze the temporal and spatial behaviors.

6. THE PROPOSED SCHEME

6.1 Existing System -Disadvantages

Temporal and spatial locality analysis used to extract the proxy to server behavior. Temporal locality of reference has been widely applied in many fields, for example, program behavior reference pattern of Web access and Web proxy cache replacement strategy. Temporal locality refers to the property that a referencing behavior in the recent past is a good predictor of the referencing behavior to be seen in the near future, whereas the resource popularity metric only represents the frequency of the requests without indicating the correlation between a reference to a document and the time since it was last accessed. Here, we resort to the concept of threshold inter distance using linked list. The files are assumed to be placed on a linked list such that, whenever a file f is requested, it is either pulled from its position and placed on the particular list, or simply added to the list if the file is not yet in the linked list. Spatial locality refers to the property that objects neighboring an object frequently accessed in the past are likely to be accessed in the future. For example, when a home page is requested, all its embedded objects are likely to be accessed at the same time. Because spatial locality indicates correlation among a cluster of HTTP requests, capturing spatial locality can help mine the access behavior of Web proxies. Different from, here a new method issued to quantize the spatial locality Here this system uses hidden semi-Markov model, model without state information. That is requests are grouped based on proxy server ID. And this model has no idea about original request source. Because proxy server hide this information from web server due to the HTTP protocol limitation. By using this model, we detect both temporal and spatial behavior in this system. And this model also reshape incoming request to remove attacking request instead of blocking proxy server. Requests from attackers and non attackers through proxy server are reached at web server side in a mixed manner. Web server has no facility to split this requests. From this request, web server detects spatial and temporal behavior using this markov model. But this will increase false positive and false negative ratio when number of non attacking client increases.

6.2 Proposed system

The architecture includes three phases: data extraction, model training, and detection and control. Different client’s requests their files to web server through proxy. The detection system extracts a proxy’s TSLs from

its reference string and generates a TSL string. TSL analysis first learned the attacks.

After training the attacks TSL algorithm find out the temporal and spatial behavior of the request. Thus by identifying that it be denial of request. Suppose it be denial of request the it give that request to the

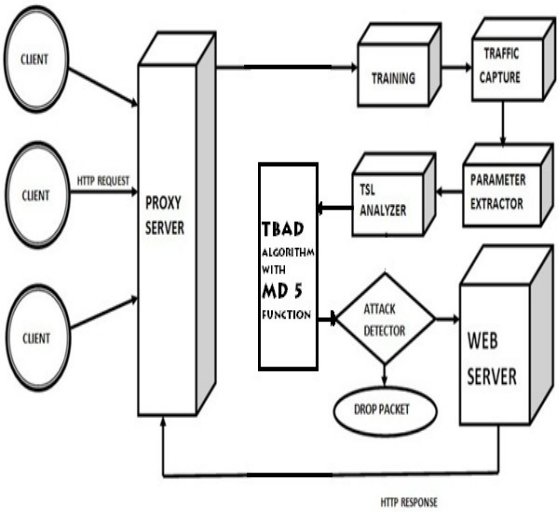

Fig 1.2 System Architecture

abnormal queue and block it. The main reason of flood attacks is the vulnerability in the protocol. For example, a UDP Flood or SYN Flood attack uses the nature of protocol's design to saturate the network traffic. In a SYN Flood attack, attacker uses the TCP 3 way handshake's first initiation step to spoof IP addresses and to drain server side system/network resources. When the subject comes to the UDP Flood attack, attacker uses the stateless design of the UDP protocol to spoof IP addresses and to drain server side system/network resources. For this reason, to accomplish an effective security solution, every mitigation method for the flood attacks must be implemented in a consideration and perspective of system/protocol design. Since HTTP protocol serves at the application (7th) layer of the OSI Model, it is possible to detect and analyze packet payloads only by application layer security devices like IPS or WAF (Web Application Firewall). For other security devices which do not serve at the application (7th) layer, there are no inspection and analyzing chance on the HTTP flood attacks. The only detection way for these devices is TCP connection counts made for the HTTP responses. As a result of detection, HTTP Flood attack attempts can be prevented by and blocked on different layers of OSI model other than application (7th) layer.

There are many situations in the real world scenarios that the HTTP flood attacks are not mitigated properly. Some of them might be related with security configuration weakness of the security device and others might be depending on an absence of a security device. Situations like these might be handled with the other security enhancements at the different level of the information technology architecture. This is where the web application level comes in. Unlike network-layer protection products, an application layer solution works within the application that it is protecting. Web application is a bunch of technologies which serves for the web service. It is important to clarify whether the attack is a HTTP flood attack or not. To consider an attack attempt as a HTTP flood attack, a TCP packet which carries a HTTP request payload should be interpreted by the web service. Attack surface for HTTP flood attacks always begins with the web service and its backend infrastructure. A HTTP flood attack attempt, which cannot make it to the web service, is just a TCP DoS attack that saturates the network traffic. To create a resistance at the web application level against the HTTP flood attack formalized into three steps: First, detect IP addresses of the abnormally excessive requests according to a previously defined rule. Second,To reduce attack surface, return these requests with a low resource used response (like a blank page or else), Third,Block detected IP addresses by using other components at the other mitigation levels (WAF, web server/service, etc.).

While reducing attack surface by sending low resource used responses, this implementation will also save resources of the backend infrastructures like SQL Servers, distributed services or e-mail/ media/application servers, etc. This is extremely important and critical for the network architectures which share the backend infrastructure members with other infrastructures like intranet or distributed web application servers. Saving the resources for the backend infrastructure will prominently reduce the amplitude of the HTTP flood attacks

6.3 The Rule Creation Concept

The critical point for the web application level HTTP flood attack mitigation is the false positives. In order to avoid false positives, the detection rules must be well defined and be tested with the real world traffic usage scenarios. Also a good understanding for the rule creation concept is highly suggested. But here we implement an enhanced HTTP protocol in this proxy server. So proxy server doesn't hide application id from web server. So web server got client identity of each request. So client can group requests based on this application ID.

7. IMPLEMENTATION

7.1 Algorithm

Here We implement a new algorithm (Threshold based attack detection – TBAD) for detecting attacks modules of the proposed TBAD, viz., Traffic capture, Parameter extractor, and the Analyzer module. First the packets are captured at kernel level by the traffic capture module. The output of the traffic capture module, the outbound TCP packets alone, are filtered and sent to the parameter extractor module which extracts the features such as remote IP address and the arrival time of packets. Then the packets are subjected to the Analyzer which decides whether to drop or allow the packets into the network based on the threshold set The analyzer module contains an IP frequency list to store the number of occurrences of individual IP address over a period of time. It checks the frequency of each IP address in IP frequency list and further decides whether to block or allow the packets. The flow chart for the basic functioning of TBHF is given in Fig. 1.3. Threshold value is set to restrict the number of HTTP requests to a particular IP address for a given period of time. Based on the parameters extracted from the packets ΔT values are calculated, which gives the time interval between current and previous instance of a packet for a particular IP address. Based on the threshold set if the value of the IPfrequency list exceeds the threshold value, then the packets are dropped else they are allowed into the network. Using HTTP GET different IP addresses are extracted from the filtered packets. The presence of the extracted IP address is checked in the IP address list and accordingly the IP frequency list is updated. If the IP address is not found in the IP address list, then the new IP address and its corresponding arrival time are stored in the IP address list. Then in Step 3, the difference in time between two packets of same IP address (ΔT) is calculated. If the time interval is less than 1 second, then the IP frequency list value for the corresponding IP address is incremented by 1. Otherwise the IP address is added to the IP frequency list. The time interval to refresh list is set to 1 second and also the IP frequency list values are reset to null after the elapse of every second. The threshold value (N) is set to 20 based on the experiments conducted in our lab, viz., only 20 HTTP GET requests are allowed to a particular IP in one second. The HTTP GET packets are allowed to an IP address till the corresponding IP frequency list value reaches N. If the IP frequency list value for an IP address exceeds N, then the packets are dropped.

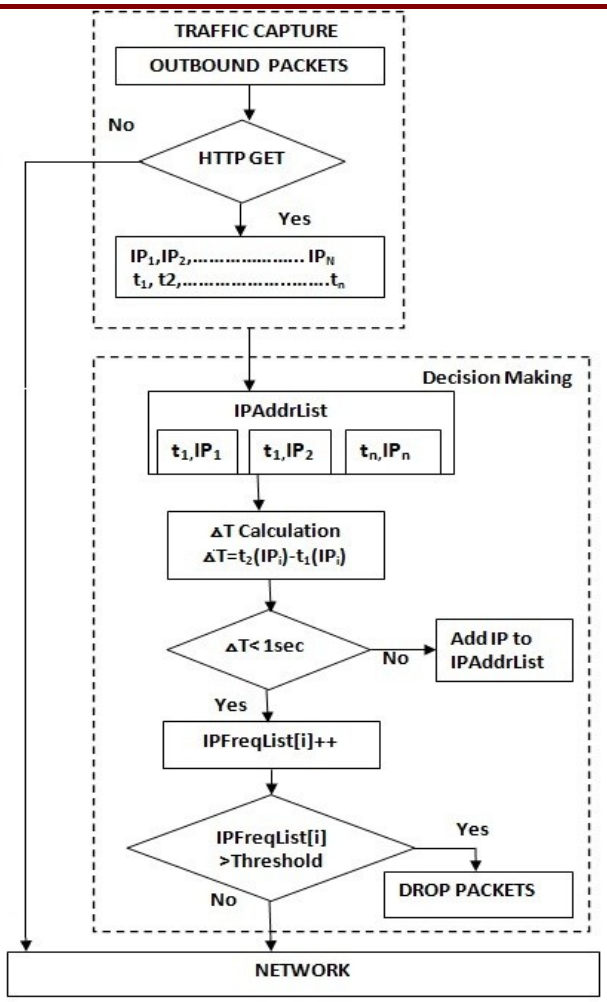

Fig1.3 TBAD Algorithm Flow Chart

7.3TBAD Algorithm

Input: *Network Traffic*

Step 1:

IF (Outbound packets)
THEN
IF (Packet == HTTP GET)
THEN

Step 2:

//ExtractParameters
// IP1, IP2, …,IPi - remote IP address
// t1, t2, …, ti - Arrival time of packets
//IPAddrList – List of ip address
IPAddrList[IPi][0] = IPi;
IPAddrList[IPi][1]=ti;

Step 3:

// ΔT - Difference in time between two instance of same IP address
// N - Threshold value
//IPFreqList - IP Frequency List
ΔT = t2(IPAddrList[IPi][1]) - t1 (IPAddrList[IPi][1]);
IF (ΔT < 1 second)
THEN
IPFreqList[i]++;
IF (IPFreqList [i] < N)
THEN
Allow packet to Network;
ELSE
Drop packet;
END IF
END IF
ELSE
Allow packet to Network;
END IF
END IF

8 EXPERIMENT AND ANALYSIS

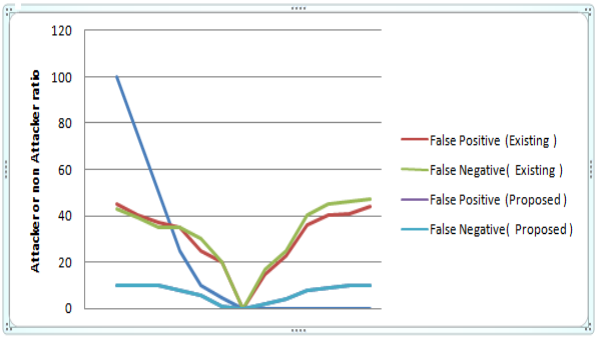

Fig 1.4 Performance graph

In base paper, both false positive and false negative ratio increases when difference between number of attacking and non attacking packets increases. But from figure 1.4, it is clear that it have no effect on proposed system.

9 .CONCLUSION

Application layer attack has become a major threat to the internet in today's world. The focus of this project was to come out with an effective solution for the detection and prevention of clients from inadvertently taking part in such attacks. Accordingly, a Threshold

Based Attack Detection (TBAD) was proposed and implemented in Linux OS using J2SDK. Experiments were conducted by generating HTTP GET attacks and using TBAD for its mitigation. It was evident that the TBAD suppressed the flooding packets and thus prevented the client system from taking part in such an attack. The ongoing work is to implement TBAD in order to find different type DDoS attacks

REFERENCES

[1] A. Mahanti, D. Eager, and C. Williamson, "Temporal Locality and Its Impact on Web Proxy Cache Performance,"(2000) Performance Evaluation, vol. 42, nos. 2/3, pp. 187-203,2000.Computing, vol. 5245, pp. 61-73, 2008.

[2] J. Yu, C. Fang, L. Lu, et al., "A lightweight mechanism to mitigate application layer DDoS attacks," (2009)Scalable Information Systems, vol. 18, pp. 175–191, 2009.

[3] P. Garcia-Teodoro, J. Diaz-Verdejo, G. Macia-Fernandez, and E. Vazquez, "Anomaly-Based Network Intrusion Detection: Techniques, Systems and Challenges,"(2009) Computers and Security, vol. 28, nos. 1/2, pp. 18-28,2009.

[4] S. Lee, G. Kim, and S. Kim, "Sequence-Order-Independent Network Profiling for Detecting Application Layer DDos Attacks,"(2011) EURASIP J. Wireless Comm. and Networking, vol.2011, no. 1, p. 50 2011.

[5] S. Yu, W. Zhou, R. Doss, and W. Jia, "Traceback of DDos Attacks Using Entropy Variations,"(2011) IEEE Trans. Parallel and Distributed Systems, vol. 22, no. 3, pp. 412-425, Mar. 2011.

[6] S. Yu, W. Zhou, W. Jia, S. Guo, Y. Xiang, and F. Tang,"Discriminating DDos Attacks from Flash Crowds Using Flow Correlation Coefficient,"(2012) IEEE Trans. Parallel and Distributed Systems, vol. 23, no. 6, pp. 1073-1080 july 2012

[7] Y. Xiang, K. Li, and W. Zhou, "Low-Rate DDos Attacks Detection and Traceback by Using New Information Metrics," (2011) IEEE Trans. Information Forensics and Security, vol. 6, no. 2, pp. 426-437, June 2011.

[8] Y. Xie and S. Yu, "A Large-Scale Hidden Semi-Markov Model for Anomaly Detection on User Browsing Behaviors," (2009)IEEE/ACM Trans. Networking, vol. 17, no. 1, pp. 54-65, Feb. 2009.

[9] Y. Xie and S. Yu, "Measuring the Normality of Web Proxies Behavior Based on Locality Principles," (2008) Network and Parallel

[10] Y. Xie and S.-Z. Yu, "Monitoring the Application-Layer DDoS Attacks for Popular Websites,"(2009) IEEE/ACM Trans. Networking, vol. 17, no. 1, pp. 15-25, Feb. 2009.