Alleviation of Application Layer DDoS Attacks Using Data Specification Module

R.Saravanan 1† and Vigneswari.K2††,

University of Pondicherry, Sri Manakula Vinayagar Engineering College, Madagadipet, Pondicherry, India

Summary
Distribute denial of service (DDoS) attacks can inflict chaos on any susceptible web site. The goal of these attacks is to consume the network bandwidth and reject services to legitimate users of the targeted systems. Hence the client loyalty and buoyancy can be eroded due to annoyance of slow site performance .The attacks at the layer-7 are more harder to alleviate since they deny the service without causing the consumption of available network bandwidth when compared to network layer DDoS attacks. Several mechanisms has been introduce to alleviate application layer DDoS attacks in which the attackers are identified and blocked after reaching the server. Our proposed system has incorporated a module called data specification module in which the attackers are chunked in the client side itself. This can be done by computing the trust of a client based on the threshold value by considering the parameters such as OS name, port number, IP address and Mac address. Thus only the legitimate users can be allowed to send requests and access the service. Thus an attacker can be eliminated in the client side thereby reducing the bandwidth overhead in the server and only the legitimate user is given priority to access the services.

Key words:
Distributed Denial of Service (DDoS) attacks, Reacting module, Trust, Trust Management Helmet

1. Introduction

Distribute denial of service attacks are the attacks in internet in which multiple computers launch a coordinated denial of service attack against one or more targets. The most common DoS attacks will target the computers network bandwidth. The group of systems attacks a single target, thereby causing denial of service for users of the targeted system. The flood of incoming requests to the targeted system essentially forces it to shut down, thus denying service to the legitimate users. Bandwidth attacks flood the network with a high volume of traffic such that all available network resources are consumed and legitimate user requests cannot 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. The two main classes of DDoS attacks are resource flooding and bandwidth flooding. In resource flooding the attacker consumes victim computer resource (memory, CPU, hard disk) to make it unavailable for legitimate users. In bandwidth flooding the victim network is flood by unwanted traffic to prevent the legitimate traffic from reaching the victim network.

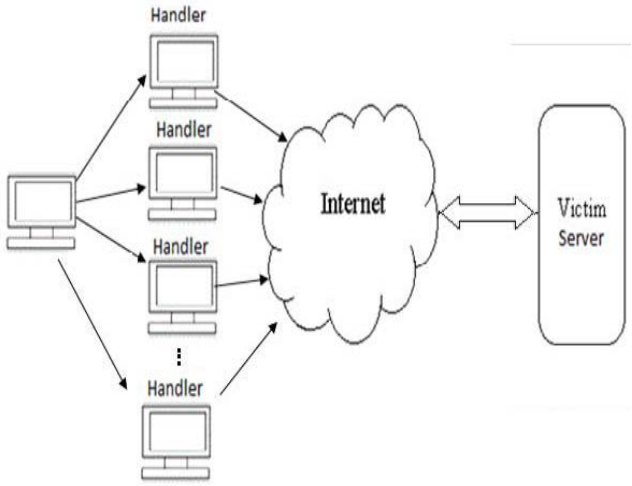


Fig1. DDoS attacks in internet

2. App- Layer DDoS Attacks

The new variant of DDoS attack is the application layer DDoS attack. Application layer DDoS attack is a DDoS attack that sends out requests following the communication protocol and thus these requests are indistinguishable from legitimate requests in the network layer. Application layer DDoS attacks employ legitimate HTTP requests to flood out victim's resources. Attackers attacking victim web servers by HTTP GET requests (HTTP flooding) and pulling large image files from victim server in large numbers. Sometimes attackers can run large number of queries through victim's search engine or database query and bring the server down the first characteristic of App-DDoS attacks is that the application-layer requests originating from the compromised hosts are indistinguishable from those generated by legitimate users. Application layer DDoS attacks include session flooding attack, request flooding and asymmetric attack. Session flooding attack Session flooding attack sends session connection requests at higher rates than that of legitimate

Manuscript received April 5, 2013
Manuscript revised April 20, 2013

users. Request flooding attack sends sessions that contain more requests than normal sessions.
Asymmetric attack sends sessions with higher workload requests. Most of the well known sites are affected by these kinds of attacks. Commercial sites are more vulnerable during the business time as there will be many genuine users accessing it, and attacker needs only a little effort to launch DDoS attack. It is difficult to prevent such attacks from happening and the attackers may continue their damage using new and innovative approaches. These attacks not only send network packets, but they actually complete TCP connections from the attacker to the victim service. Once the TCP connection is made , the attacking computers make repeated requests to the application in an attempt to exhaust the resources of the application , rendering it to unable to respond to all of its requests. These intelligent attacks are harder to defend against because they create denial of service conditions without causing the consumption of available network bandwidth or overloading outers, firewalls and switches.eg.HTTP request.
Application layer DDoS attacks causes unavailability of resources, revenue loss. The request originating from the compromised hosts are indistinguishable from those generated by legitimate users. Usually app DDoS attacks utilize the weakness enabled by the standard practice of opening services such as http through most firewalls to launch the attack. Many protocols and applications, both legitimate and attacker can use these openings to tunnel through firewalls by connecting over a standard port address.

3. RELATED WORK

S. Ranjan et al. proposed a counter-mechanism by building legitimate user model for each service and detecting suspicious requests based on the contents of the requests. To protect servers from application layer DDoS attacks, they proposed a counter-mechanism that consist of a suspicion assignment mechanism and DDoS resilient scheduler DDoS shield. The suspicion mechanism assigns continuous value as opposed to a binary measure to each client session, and scheduler utilizes these values to determine if and when to schedule a session's requests.
M. Srivatsa et el. performed admission control to limit the number of concurrent clients served by the online service. Admission control is based on port hiding that renders the online service invisible to unauthenticated clients by hiding the port number on which the service accepts incoming requests. The mechanism needs a challenge server which can be the new target of DDoS attack.
J. Yu, Z. Li, H. Chen, and X. Chen proposed a mechanism named DOW (Defence an Offence Wall), which defends against layer-7 attacks using combination of detection technology and currency technology. An anomaly detection method based on K-means clustering is introduced to detect and filter request flooding attacks and asymmetric attacks. But this mechanism requires large amount of training data.
Yi Xie and Shun-Zheng Yu introduced a scheme to capture the spatial-temporal patterns of a normal flash crowd event and to implement the App-DDoS attacks detection. Since the traffic characteristics of low layers are not enough to distinguish the App-DDoS attacks from the normal flash crowd event, the objective of their work is to find an effective method to identify whether the surge in traffic is caused by App-DDoS attackers or by normal Web surfers. Web user behaviour is mainly influenced by the structure of Website (e.g., the Web documents and hyperlink) and the way users access web pages. In this paper, the monitoring scheme considers the App-DDoS attack as anomaly browsing behaviour.
Our literature survey has noted that many mechanisms are developed to service legitimate users only. Abnormalities are identified and denied. But large amount of training data is required. Sometimes mitigation mechanism can itself becomes target of DDoS attack. The need is felt to design and develop a new lightweight mechanism that can alleviate both session flooding and requests flooding Application layer DDoS attacks with small amount of training data. It will service all users if and only if resource is available and use bandwidth effectively. It will identify the abnormalities and serve them with different priorities.

4. PROPOSED WORK

To intuitively describe about our approach, we construct a framework to filter the various users according to their trust value. In our model, we focus to chunk the attackers in the client side itself before their request reaches the server. We extract a user based on the Trust value which is composed by Data Cleansing Technique. The mechanism used for eliminating the attackers after reaching the server side requires more bandwidth overhead which can reduce the service to slow down. Hence to overcome this problem we proposed a framework in the client side which consists of reacting module and data specification intrusion extraction module where the clients are segregated based on their trust value. The trust of each client can be evaluated based on the threshold value considering the parameters such as Os name, ip address, port number and Mac address. Threshold Value is the number of requests that a server can handle without straining its resources. It is defined as a predetermined percentage of the maximum number of requests that a server can handle. The data specification modules segregate the clients into positive, negative, untrusted and trusted and are queued in the respective unit. The negative

clients whose trust values are too low beyond the threshold value are treated as attacker and are rejected. The rest trusted clients those queued in the units are sent to the next level and they are again validated by using TMH mitigation mechanism in the server side for further verification.

For each session connection request TMH checks whether the client is blacklisted; if not, it computes the trust to the client and schedule the connection request for the server using trust--based scheduling. Fig 2 shows the architecture of the proposed system. When the clients send the request to the browser it reaches the reacting module which decides whether the client can be sent to the next level or filtered.

4.1 Reacting Module

Legacy data comes from virtually everywhere within the information system. The legacy data are the information stored in an obsolete format in computer systems which are difficult to access or process. The reacting module in the client machine Fig 2 decides whether the request has to be sent to next level or to be filtered. This filtration process is done with the help of the Data Cleansing technique.

This module will verify the legacy data which are provided by the users are correct and forwards the requests to the next level. During this process, the data is checked for accuracy and consistency.

The filtration process involves the following.

- Checks for inaccurate record or data
- Checks for typos or spelling errors
- Checks for obsolete records or data
- Checks for incomplete records or data

4.2 Data Intrusion Extraction Module

Data Intrusion Extraction (DIE) is a software application that monitors the system activities for malicious activities or policy violations and produces reports to a Management Station. Data Intrusion Extraction system primarily focused on identifying possible incidents, logging information about them, and reporting attempts. Typically this system record information related to observed events, notify security administrators of important observed events, and produce reports.

Data Intrusion Extraction Module (DIE) consists of four blocks namely Negative Data Cleansing, Positive Data Cleansing, Unidentified Untrusted Data and Unidentified Trusted Data. These four blocks are used to detect the client behaviour in various means. DIE will monitor the requests which are travelled from Reacting Module and evaluate the Trust percentage of the client. The trust of the client is built up through his visiting history, IP, Current OS, Port number etc. and used as the criteria in evaluating the likelihood of the client being Legitimate or not.

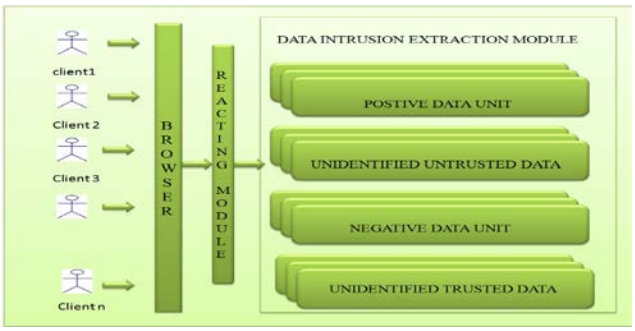
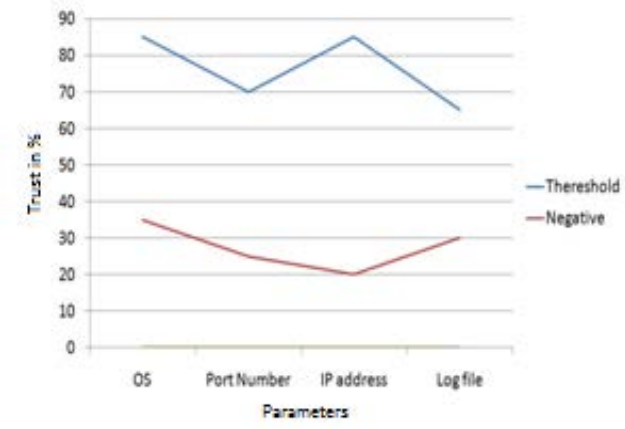


Fig.2 Proposed system with Data Intrusion Extraction Module

4.3 Negative Data Cleansing Unit

Negative Data Cleansing Unit stores the requests whose trust value is too low when compared to the Threshold value as shown in Graph 1. The threshold value is the number of requests that a server can handle without straining its resources. It is defined as a predetermined percentage of the maximum number of requests that a server can handle.

The system will treat these requests as attackers and reject their request. The queued request in the Negative Data Cleansing unit cannot be moved further to access the service which it request. The Negative Data Cleansing unit contains also contains Unidentified Untrusted data unit.

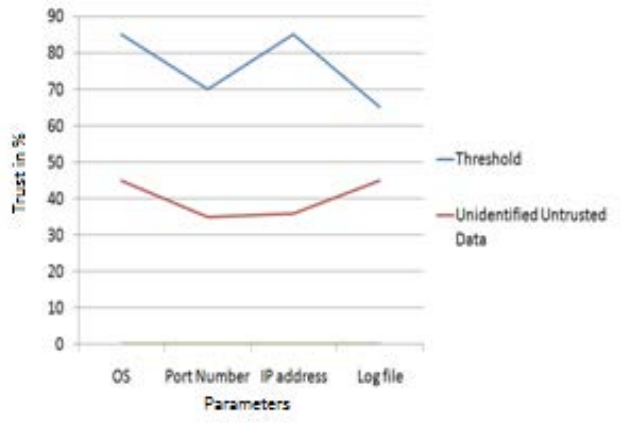


Graph 1: Negative Data Cleansing Unit

4.4 Unidentified Untrusted Data Unit

Unidentified Untrusted Data Unit is used to queue the client request which cannot be identified by the Reacting module whether they are legitimate requests or attacks. The client may ask for the proper service but the trust value does not meet with the threshold value. Those client requests are queued in this unit. The queued request is further moved to the next level. Thus the queued requests

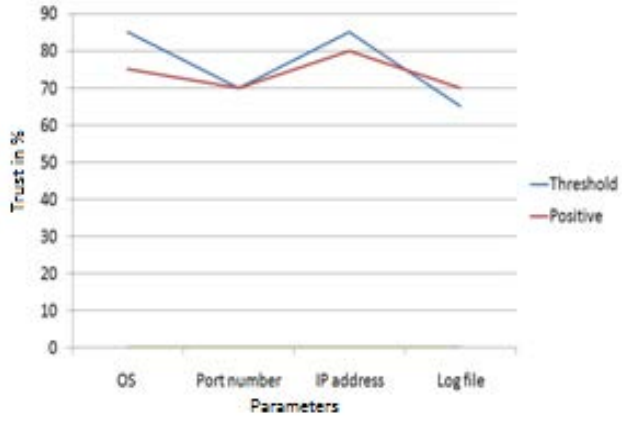
in this unit can able to access the service which it requested.



Graph 2 : Unidentified Untrusted Data Unit

4.5 Positive Data Cleansing Unit

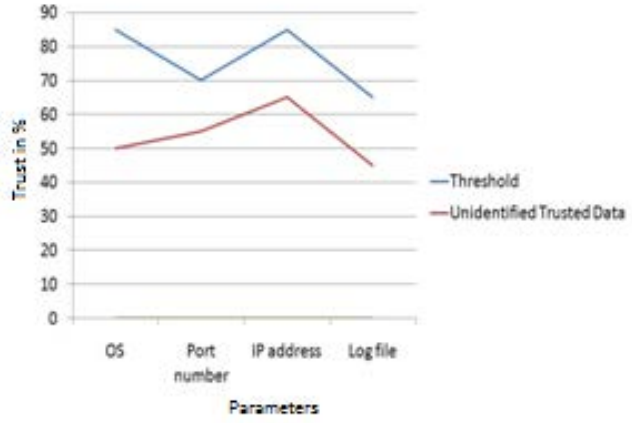
The Trust value of the client is greater than the threshold then that client requests are queued into the Positive Data Cleansing unit. The Positive Data Cleansing unit also contains the unidentified trusted data that is the user cannot be identified by the Reacting module whether the client is legitimate user or not. Since the user action is trusted, so that types of requests are queued into this unit.



Graph 3: Positive Data Unit

4.6 Unidentified Trusted Data Unit

The client requests which are queued in the Positive Data Cleansing unit, Unidentified trusted data and the unidentified untrusted data are then transferred to the next level i.e. to TMH for further verification. The TMH in the server side will again calculate the trust value of the client requests for deciding about the legitimate user or not. By evaluating the user request in the client side using the Data Intrusion Extraction Module the bandwidth overhead is reduced.



Graph 4: Unidentified Trusted Data Unit

5. Related Work

Recently, the more sophisticated application layer DDoS attack [1] is threatening the security of the Internet content providers, especially web servers. One critical application layer DDoS attack is the index reflection attack [2]. In this attack, attackers declare to be the victim and pretend to share lots of resources in peer-to-peer (P2P) network, so as to fool a large number of peers into requesting download of resources from the victim. Since application layer DDoS attacks are non-intrusive and protocol- compliant, attackers are indistinguishable based on packets or protocols and thus these attacks cannot be defended using network layer solutions. Walfish et al. [3] proposed a speak-up method that encourages clients to send more session connection requests. This method is based on the assumption that attackers are already using most of their upload bandwidth so that they cannot react to the encouragement.

Ranjan et al. [2] proposed a counter-mechanism by building legitimate user model for each service and detecting suspicious requests based on the content of the requests. Yu et al [4] proposed a Trust Management Helmet (TMH) as a partial solution to this problem, which is a lightweight mitigation mechanism that used trust to differentiate legitimate users form attackers. Its key insight is that a server should give priority to protecting the connectivity of good users during application layer DDoS attacks, instead of identifying all the attack requests. Yu j., li z., Chen h., Chen x et al [6] proposed a mechanism named as DOW (Defence and offense wall), which

defends against layer-7 attacks using combination of detection technology and currency technology.

M. Walfish et al. [3] proposed a speak-up method, which encourages clients to send more session connection requests. This method is based on the assumption that attackers are already using most of their upload bandwidth so that they cannot react to the encouragement.

S. Ranjan et al. [1] proposed a counter-mechanism by building legitimate user model for each service and detecting suspicious requests based on the content of the requests. S. Khattab et al. [9] proposed living baiting for applications that can be decomposed into several virtual services. It leverages group-testing theory to detect attackers with small state overhead. J. Yu et al. [3] introduced a detection and offense mechanism to protect legitimate sessions, but it is too resource consuming to be implemented. Xie y, Yu s. et al [7], proposed to describe the dynamics of Access Matrix and to detect the attacks. The entropy of document popularity fitting to the model is used to detect the potential application-layer DDoS attacks. Numerical results based on real Web traffic data are presented to demonstrate the effectiveness of the proposed method. Yu j., Fang c., Lu l., Li z. et al [8] propose Trust Management Helmet (TMH) as a partial solution to this problem, which is a lightweight mitigation mechanism that uses trust to differentiate legitimate users and attackers.

Likewise, the scheme proposed in [10] rate limits incoming traffic while timing out flows and forcing them to retransmit several times in order to be successful. This scheme is based on the assumption that attackers would run out of bandwidth earlier than legitimate users, an assumption that is not necessarily valid for the attacks that we consider in which a moderate request rate can generate an overwhelming server workload. Such techniques are all geared towards countering high bandwidth flows reminiscent of today's DDoS attacks. In contrast, by rate limiting the work a server cluster performs, we can prevent attacks on both network bandwidth as well as those that are aimed at other types of system resources, such as CPU or storage.

Distinguishing a DDoS attack from a flash crowd has also proven difficult. Two properties to make the distinction are identified in [11]: (1) a DoS event is due to an increase in the request rates for a small group of clients while ash crowds are due to increase in the number of clients; and (2) DoS clients originate from new client clusters1 as compared to ash crowd clients which originate from clusters that had been seen before the ash event.

These characteristics may not help distinguish the attacks discussed in this paper since (1) it is difficult to associate the amount of resources consumed to a client machine and (2) botnets consisting of geographically wide-spread machines are increasingly likely to belong to known client clusters.

S. Ranjan, R. Swaminathan, M. Uysal, and E. Knightly et al [1] propose we consider sophisticated attacks that are protocol-compliant, non-intrusive, and utilize legitimate application-layer requests to overwhelm system resources. We characterize application layer resource attacks as either request flooding, asymmetric, or repeated one-shot, on the basis of the application workload parameters that they exploit. S. Khattab, S. Gobriel, R. Melhem, and D. Mosse et al [9] propose live baiting, a novel approach for detecting the identities of DoS attackers. Live baiting leverages group-testing theory, which aims at discovering defective members in a population using the minimum number of "tests".

Yu j., Fang c., Lu l., Li z. et al [4] propose trust management helmet (TMH) as a partial solution to this problem, which is a lightweight mitigation mechanism that uses trust to differentiate legitimate users from attackers. Its key insight is that a server should give priority to protecting the connectivity of good users during application layer DDoS attacks, instead of identifying all the attack requests.

The trust to clients is evaluated based on their visiting history and used to schedule the service to their requests. The authors introduce license, for user identification (even beyond NATs) and storing the trust information at clients. The license is cryptographically secured against forgery or replay attacks. In this paper, we apply trust management to defend against application layer DDoS attacks.

6. Conclusion

This paper presents a framework for detecting direct DDoS attacks. The system consists of a client detector and a server detector. The TMH in the server side is used detect for further verification that can both passively and actively detect DDoS attacks. In order to monitor the behaviour of the client machine various queuing techniques are followed. Thus this helps us to reduce the bandwidth as well as the workload of the server.

Then, a mechanism named as Data Intrusion Extraction modules which defends against the attacks of the various users. The users are also categorized into four types and the access is given based on that types. Only the legitimate users are allowed to access the service. Using this technique the network overhead and the bandwidth of the server will be reduced. Our further work is to implement our mechanism and deploy it in real network.

References

[1] Ranjan S., Swaminathan R., Ysal M., Knightly E.: 'DDoS-Resilient Scheduling To Counter Application Layer Attacks Under Imperfect Detection'. Proc. Infocom'0, 2006

[2] Liang J., Naoumov N.M Ross K.W.: 'The Index Poisoning Attack In P2p File Sharing Systems'. Proc. Infocom'06, 2006
[3] Walfish M., Vutukuru M., Balakrishnan H., Karger D., Shenker S.: ' DDoS Defence by Offense'. Proc. Sigcomm'06, 2006
[4] Yu J., Fang C., Lu L., Li Z.:'Mitigating Application Layer Distributed Denial Of Service Attacks Via Effective Trust Management'. ProcIssn 1751-8628, 2009
[5] Prabha S., Anitha R.: 'Mitigation Of Application Traffic DDoS Attacks With Trust And AM Based HMM Models'. Proc Sep 2010
[6] Yu J., Li Z., Chen H., Chen X.: 'A Detection and Offense Mechanism to Defend Against Application Layer DDoS Attacks'. Proc. ICNS'07, 2007
[7] Xie Y, Yu S.: 'Monitoring the Application-Layer DDoS Attacks For Popular Websites', Ieee/AcmTrans Netw.' 2009, 17,(1), Pp. 15-25
[8] Yu J., Fang C., Lu L., Li Z.: ' A Lightweight Mechanism To Mitigate Application Layer DDOS Attacks'.Proc. Infoscale'09, 2009
[9] S. Khattab, S. Gobriel, R. Melhem, And D. Mosse.' Live Baiting For Service-Level Dos Attackers'Proc Infocom'08, 2008-12
[10] M. Wal sh, H. Balakrishnan, D. Karger, and S. Shenker. DoS: Fighting Fire with Fire. In 4th ACM Workshop on Hot Topics in Networks (HotNets), College Park, MD, November 2005.
[11] J. Jung, B. Krishnamurthy, and M. Rabinovich. Flash crowds and denial of service attacks: Characterization and implications for CDNs and web sites. In Proceedings of the International World Wide Web Conference, pages 252–262. IEEE, May 2002.