A Novel Approach of Detection and Mitigation of DDOS Attack

Khundrakpam Johnson Singh, and Tanmay De

***Abstract*--**We are in the era of internet and depend on it for every necessary requirement. It is the tendency of the some human to have destructive approach rather than having constructive approach. Among the abuse and misuse of internet, the distributed denial of service attack (DDOS) is the most hectic one. People have carried out various method of mitigation using the CAPTCHA (Completely Automated Public Turing Test to tell Computer and Human Apart) technique, but frequent use of CAPTCHA test for every access may turn the legitimate client away from the server. So our proposed mechanism tries to impose CAPTCHA test only after finding the suspected clients instead of applying to all the clients. The suspected clients have to go through a CAPTCHA test in order to get the access to the server. In our experiment we considered the slowloris attack which is one of the application layer DDOS attack.

***Keywords*----**Botnets, captcha, denial of service, filtering, intrusion, mitigation.

I. Introduction

EVERYDAY our websites is visited by customers, business partners and competitors. When our website suddenly becomes offline or unavailable, then our business is affected every hour or day. When our network suddenly came to hold, it can be the symptoms of DDOS attack. DDOS attack generally generates with the internet hackers planting a virus in a huge number of computers without the consent of the customers. These infected computers can be controlled remotely to form an attack network. This network can be distributed around the world to focus on a target victim. The goal being to overwhelm a specific network location in so doing deny access to all the legitimate traffic. Today's perimeter solutions are not giving full assurance when these types of attacks occur.

Each electronic system capable of running an IP might be infected with a bot by executing malicious software. This mostly happen by luring the victim to download and run the Trojan horse program from any malicious websites in the form of email attachment or any download [1].

Khundrakpam Johnson Singh is pursuing Ph.D from National Institute of Technology, Durgapur-713209 West Bengal, India (e-mail: johnkh34@gmail.com).

Tanmay De is now with the Department of Computer Science and Engineering, National Institute of Technology, Durgapur-713209 West Bengal, India.

A group of vulnerable system under the control of a bot herder is known as a botnet [2]. Bots are different from viruses, they don't show any sign of infection or their stay in the system, thereby keeping the user unaware of these malicious downloaded files and passively take part in the greatest assemble of the bots to form the army of botnets. These infected clients are secretly taking part in the DDoS attack without the consent of the user.

Bots are controlled by the bot-master or the bot-herder through the command and control server. Any updates or instruction to the bots came from this bot-herder. Through these main system or human user they can add new features and program code to make up their defects and breach the defense algorithm. Due to this frequent updating, bots are hard to detect.

R&D of Kaspersky Labs and Symantec [3] have found that botnets are the biggest threat to Internet security. Their researches have proved that among the numerous attacks on the server, DDOS attack is quite difficult to handle. The open feature of current IP structure enables anyone to send anything at anytime without prior permission [4]. There is no restriction on the device to become the client of the bot. Any devices capable of hosting an IP address have the chances of becoming the bot clients. These devices are distributed globally and on receiving the instruction from bot master, they are ready to bring down the server. The common motives of creating such malicious software are mainly for earning money [5]. Other motives include competition among business organizations, enmity, revenge or sometime fun. The more the attackers infect users and spread the number of victims the more they earn money. The attackers adopt sophisticated malware infection way in order to lure the legitimate clients to become a victim of this malicious software. The most commonly used malware infection is through the malicious website, this technique is term as drive by download attack [6].

The remainder of the paper is structured as follows: section II provides the related works, section III provides the proposed method along with the proposed algorithm used. The experimental results are shown in section IV. Finally, we conclude this paper in section V.

II. Related Works

Cookie based accounting model [7] records each and every client request in the cookie and the hash value of the cookie in the server database to detect the client's misbehavior. These misbehavior ranges from modifying the cookie information or retransmitting the prior request cookie along with the current request. This method requires accounting the cookie history and is not effective when DDoS attack is considered. This method has to deal with the cookies from multiple clients.

Detection technique based on traffic analysis [8] allows us to identify botnet activity in real time by considering the properties of the traffic flows in small time windows using machine learning. They analyze different traffic behavior by capturing the traffic. But when we considered the DDoS attack the proposed mechanism will be unable to analyze each traffic pattern. Training and recognition of each individual traffic flow will consume time and recourses.

Re-Traffic pricing strategy [9] aims to defend against DDoS by agreeing to the bandwidth uploaded after encouragement to the server as the constrained resource. Re-traffic architecture aims to allocate service resources in rough proportion to the users' re-traffic. The method is motivated by observing and considering the activities carried out by spam transmission in the application layer DDoS attack. It is observed that bad clients exhaust most of their available bandwidth generating spurious requests for service. Where as legitimate clients used only a small portion of their available bandwidth in accessing the server.

Edge-based Capabilities (EC) [10] provides the basic mechanisms for a combined application and network defense against DDoS attack. EC provides abstractions of protocols that are cryptographically tag as legitimate traffic and control the behavior and resource consumption of non-tagged and wrongly tagged traffic. Senders in EC can generate tags when they obtain a permission to send from the intended receiver. At the network edge, a network element termed gate enforces that only tagged traffic is forwarded directly, whereas untagged or wrongly tagged traffic is treated as potentially malicious traffic. The gate therefore creates a differentiation that prioritizes legitimate traffic.

Denial of Service (DoS) [11] attack is an attack that interrupts the network by disallowing the legitimate users from utilizing the network resources. Any layer of the Open Systems Interconnection (OSI) layers can be targeted by the DoS attack. Denial-of-service attacks can essentially disable our computer or our network. Depending on the nature of the enterprise, this can effectively disable the organization. There are even asymmetric attacks [11] that are carried out with limited resources against a large, sophisticated system.

A Distributed Denial of Service (DDoS) [12] attack is a coordinated attempt to used up all the services and the resources of a victim system or a group of systems. It is launched indirectly from a large number of compromised machines on the Internet as shown in Fig. 1. A bot herder controls remotely the botnet through internet relay chat (IRC) using a command and control (C&C) server which can be located at different places. These bots entered into the infected system in the form of virus or worms through the transmission channel. Once entered into the system they log on to a particular C&C server. These bots continuously got the information and updates from the bot herder through the C&C server. Fig. 1 shows a brief idea of how bot clients are generated by hacker, who is also known in this domain as botmaster.

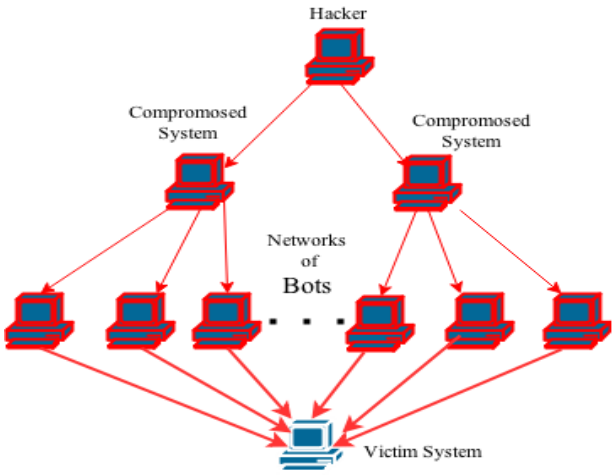

Fig.1 A typical DDoS attack on the target victim

Firewalls and IPS (Intrusion Prevention System) have limited scope regarding mitigation and protection against DDoS attacks. The main objective of these devices is to prevent one entity at a time and not on large volume of legitimate traffic. Firewall and IPS devices act as a stateful inline solutions [13], they are vulnerable to DDoS attacks there by making themselves the target. Even during moderate DDoS attacks firewall and IPS devices will be the bottleneck and can be first point of failure as shown in Fig. 2. Slowloris attack opens connections to the target web servers and keeps the connections open with partial HTTP requests. These characteristics of the slowloris attack can overwhelm the intermediate IPS device and even hinder access for the legitimate requests. We can set up or add policy in the firewall to block unwanted packets or IP addresses but this is not possible in case of DDoS attack when millions of connections are trying to access the server. Later on we could not able to set policy to filter each and every malicious packet.

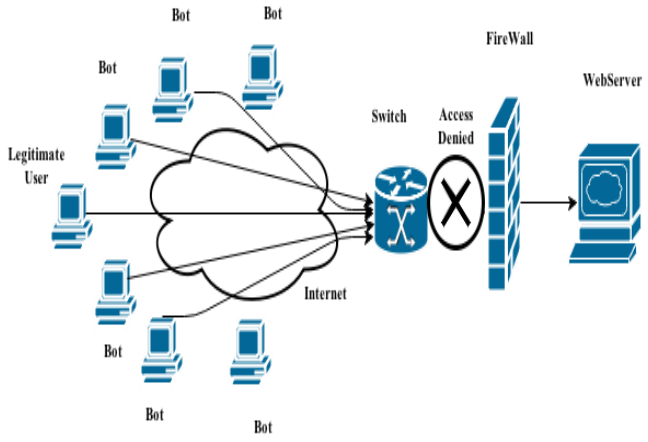

Fig. 2 Firewall becomes the bottleneck in case of DDoS attack.

III. Proposed Method

We are considering the application layer attack in the experiment which is carried out in the Linux environment. This type of attack is difficult to handle as they cannot be detected easily. The application layer DDoS attack uses legitimate http request and is difficult for the firewall and IDS system to detect. As we are considering only the application layer DDoS attack, mechanism to detect and mitigate network layer DDoS attack are not considered in the paper. Compared to application layer DDoS attack detection and its defense mechanism the network layer DDoS attack defense mechanism are easier. Our proposed mechanism is broadly divided into three folds, first we filter out the blacklisted IP address and those IP address from restricted IP ranges. In our experiment we take up the already available range of blacklisted IPs. These IP address are blacklisted as they either cause IP spoofing or may have potentials of carrying out unwanted activities. All these blacklisted IP addresses are updated in the bad-host file. This simple filter cannot be able to block IP address that has the capability of DDoSing. A dataset of blacklisted IP has to be maintained, every new entry of IP address has to be checked against the dataset. These procedures are not effective when we have new entry having the capability of DDoS attack. The algorithm for blocking the known blacklisted IP addresses is given below.

BEGIN

1: Set up a default policy for accepting.
2: Monitor the traffic flow through the network.
3: Consider all the ports that are monitored over the network perimeter.
4: Construct an IP tables with the ports.
5: Initialize the IP table with the default policy to accept.
6: Enable traffic to flow between the networks.
7: Check the bad-hosts file to know the blacklisted IP addresses.
8: Add the blacklisted IP addresses to IP tables with action as block
9: Bypass the IP addresses that are not in the bad-hosts file
10: Flush all the rules and add the rules again with the newly added IP addresses.
11. Add dropped IP addresses of the successive method of filtration to the bad-hosts file.
12. Repeat step 10 for every new entry in the bad-host file.

END

To cope up the limitations in the previous filtering mechanism we monitor the bypassed IP addresses http GET request rate. Every IP address http GET request is compared with the normal request rate. Any suspicious IP address having maximum variation in the http GET request is marked and sent for the next detection mechanism. Simple blacklisting of IP addresses are unable to filter out the malicious clients. So we apply one more detection mechanism soon after the first filter. The algorithm to count the http GET request and detect the suspected IP addresses is given below:

BEGIN
FOR each bypassed IP addresses
Monitor the IP addresses
Compute the Http count Count1 of each IP addresses
Compute the normal Http count Count2
IF (Count1 ≤ Count2)
The request is legitimate hence grant access
ELSE
Proceed for the next filter test
END IF
END FOR
END

The previous filtering mechanism provides us the IP addresses that are suspected to be the malicious IP addresses but does not guarantee fully. Even after the previous detection mechanism the suspicious IP addresses are not updated in the bad-host file, these IP addresses undergo the next filter test known as CAPTCHA test [14]. The CAPTCHA test provides a mechanism that can easily differentiate the legitimate users from the automated bots. By taking the disability of the bot to pass the CAPTCHA test, we could easily find out the clients running the automated program which enable them to send http GET request. Such clients are warned, their IP address are blacklisted and updated in the blacklisted IP table. The CAPTCHA test is divided into two modules: generation module and verification module. The objective of the generation module is to send a CAPTCHA text to the suspected addresses by modifying the web pages. The objective of the verification module is to capture the reply to

the CAPTCHA test and verify whether the matching CAPTCHA text is obtained. The algorithm for the CAPTCHA test is given below:

```
BEGIN
  For each suspected IP address from previous filter
  Generate text CAPTCHA to the particular IP address by
  modifying the request web page.
   Compute a time counter Tcount limit
   Tcount = Tcount - 1
   IF (Tcount ≠ 0)
       IF (Correct Reply from IP address)
          The request is legitimate and hence forward for
          normal access
        ELSE
          Drop the IP and update in the blacklist bad-hosts
          file
        END IF
    ELSE
       Update IP address in the blacklist bad-host file
   END IF
       END FOR
END
```

The CAPTCHA test rejects any mismatched in the text send and text entered by the clients. We also set up a time limit so that any delay to the reply of the CAPTCHA test is not entertained and are blocked. We also set up a policy so that multiple attempts that exceeds three times are blocked. These blocked IP address are then updated in the blacklisted IP table

IV. Experimental Results

The experiment is carried out in real time on Linux environment. We create an attack scenario by creating a webserver and numerous clients. The webserver has the capability of granting access to the normal as well as malicious clients. All the clients have a unique IP address and all are configured to access the webserver. Considering a minimum of fifteen clients and a web server, we install the slowloris attack package for apache version in one of the clients. We run the proposed rate limiting algorithm, the experiment shows that we could easily monitor the traffic flow towards the web server and could count the number of http GET request from each IP address. We also compute the normal http GET request rate and find out the average rate. We then compare the average normal rate to that of the suspected rate. Such IP addresses are marked and proceed for the net CAPTCHA test. The statistics of various IP address is given in Table 1. In order to have easy understanding and better visibility we named the fifteen IP addresses that we considered as A,B,C,D,E,F,G,H,I,J,K,L,M,N,O respectively. This is to have better virtualization of the IP addresses.

TABLE I
STATISTICS OF VARIOUS IP ADDRESS

Sl.No	IP Address	http count	packets per millisecond
1	A	262	0.017778
2	B	4	0.000271
3	C	60	0.004071
4	D	92	0.006243
5	E	49	0.003325
6	F	25	0.001696
7	G	25	0.001696
8	H	54	0.003664
9	I	54	0.003664
10	J	4	0.000271
11	K	3	0.000204
12	L	13	0.001189
13	M	28	0.001785
14	N	39	0.002117
15	O	5	0.000321

All the IP address towards the web server are monitored and we extract the packet per second rate for each IP address as shown in Fig.3. The IP having the maximum number of packet per second count is suspected of having an automated system to generate numerous packets. Normal legitimate clients are unable to generate the http GET request to exceed a maximum value. These always require an automated system to generate unnecessary http GET request.

Similarly we also extract the http GET request count from individual IP address. We monitor the IP address having the maximum number of http GET request count as shown in Fig. 4; such an IP is listed under suspected mode.

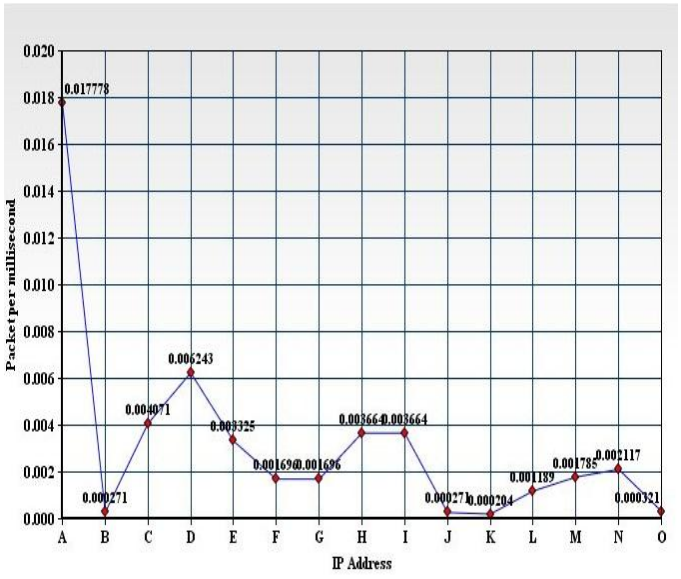

Fig. 3 Packet per millisecond count for individual IP address

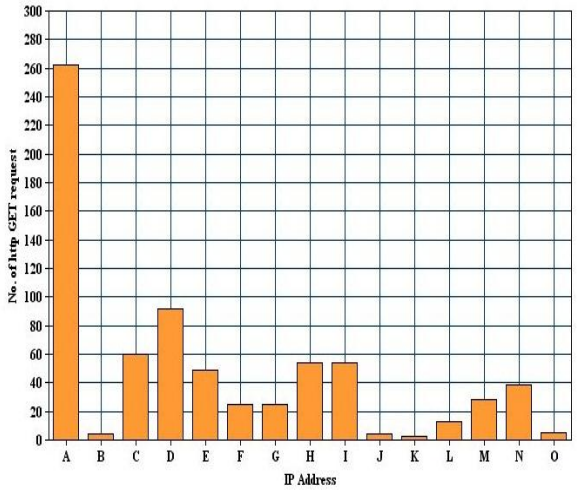

Fig. 4 Number of http GET request count for each IP address

From Fig. 3 and Fig. 4 we view that the IP address A is having the maximum number of packet rate and http GET request count. This IP address is kept under suspected mode and further proceeds to the final stage of CAPTCHA test to differentiate between legitimate client and bots. Fig. 5 shows a simple CAPTCHA test, the inability of the bots to solve such problem become the solution to the differentiation of the normal clients from bots.

Fig. 5 A simple text based CAPTCHA test

V. Conclusion

The distributed denial of service attack cannot be mitigated with a single defense line. It requires something distributed in nature. Our mitigation solution is a three-step process. Firstly the process of blacklisting IP from malicious source or known malicious IP address, secondly the process of monitoring http GET request rate and marking the IP with the highest count of http GET request. Lastly the counter check of the suspected IP with CAPTCHA test to differentiate between legitimate clients and bots. Our experimental results shows that the IP address which have the maximum number of http count seems to be running some automated program to send http GET request. These IP address is tested against CAPTCHA test and failed to authenticate. Further work is needed to monitor on the parameter like the content of http request and also improve the process of blacklisting malicious source. We consider only the text based CAPTCHA technique to differentiate the normal and malicious clients. It can be extended even to motion based technique and selection of image to carry out the differentiating mechanism. We consider only the slowloris attack; this can be extended even to other types of application layer DDoS attack.

Acknowledgment

The author would like to thank the anonymous referees, reviewers and editors for their valuable comments and their feedbacks for better improvement of the paper.

References

[1] Amirmohammad Sadeghian and Mazdak Zamani, "Detecting and Preventing DDoS Attacks in Botnets by the Help of Self Triggered Black Holes," 2014 Asia-Pacific Conference on Computer Aided System Engineering (APCASE). http://dx.doi.org/10.1109/APCASE.2014.6924468

[2] Sergio S.C. Silva, Rodrigo M.P. Silna, Raqel C.G. Pinto, Ronaldo M. Salles, "Botnet: A Survey," Computer Networks, Volume 57, Issue 2, 4 February 2013, pp. 178-403. http://dx.doi.org/10.1016/j.comnet.2012.07.021

[3] Kim-Kwang Raymond Choo, "The cyber threat landscape: Challenges and future research directions," Computer & Security, Volume 30, Issue 8, November 2011, pp. 719-731. http://dx.doi.org/10.1016/j.cose.2011.08.004

[4] Karrer R.P, Deutsche Telekom Labs, "EC: an edge-based architecture against DDoS attacks and malware spread," Proceedings of 20th International Conference on Advanced Information Networking and Application (AINA'06).

[5] J.Franklin, V. Paxson, A. Perrig and S. Savage, "An inquiry into the nature and causes of the wealth of internet miscreants," in Proceedings of the 14th ACM conference on Computer and communications security (CCS'07) October 2007, pp. 375-388.

[6] Yoshiro Fukushima, Yoshiaki Hori, Kouichi Sakurai, "Proactive Blacklisting for Malicious Web Sites by Reputation Evaluation Based on Domain and IP Address Registration," International Joint Conference of IEEE TrustCom-11/IEEE ICESS-11/FCST-11, pp. 352- 361.

[7] S. Venkatesan,M.S. Saleem Basha, C.Chellappan, Anurika Vaisha P. Dhavachelvan, "Analysis of accounting models for the detection of duplicate requests in web services," Journal of King Saud University - Computer and Information Sciences Volume 25, Issue 1, January 2013, pp. 7–24

[8] David Zhao, Issa Traore, Bassam Sayed, Wei Lu, Sherif Saad, Ali Ghorbani, Dan Garant, "Botnet detection based on traffic behavior analysis and flow intervals," 27th IFIP International Information Security Conference,Computers & Security Volume 39, Part A, November 2013, pp. 2–16 http://dx.doi.org/10.1016/j.cose.2013.04.007

[9] Yue-Yun Shen,Feng-Qin Fan, Wen-Xiu Xie , Lu-feng Mo, "Re-Traffic Pricing for fighting against DDoS," ISECS International Colloquium on Computing, Communication, Control, and Management (CCCM '08), Volume 2, 2008.

[10] Karrer, R.P, Kuehn, Huehn, "Joint application and network defense against DDoS flooding attacks in the future Internet," Second International Conference on Future Generation Communication and Networking, 2008. FGCN '08, Volume 1. http://dx.doi.org/10.1109/FGCN.2008.168

[11] N.Hougue, Monowar H.Bhuyan, R.C. Baishya, D.K Bhattacharyya, J.K Kalita, "Network attacks: Taxonomy, tools and systems," Journal of Network and Computer Applications, Volume 40, April 2014, pp. 307-324.

[12] Hamza Rahmani, Nabil Sahli, Farouk Kamoun, "DDoS flooding attack detection scheme based on F-divergence," Computer Communication, Volume 35, Issue 11, 15 June, 2012, pp. 1380-1391. http://dx.doi.org/10.1016/j.comcom.2012.04.002

[13]Errin W.Fulp, "Chapter 6- Firewalls," Managing Information Security (Second Edition), 2014, pp. 143- 175. http://dx.doi.org/10.1016/B978-0-12-416688-2.00006-4

[14]Ying-Lien Lee and Chih-Hsiang Hsu, "Usability study of text-based CAPTCHAs," Displays 32 (2011) pp. 81–86. http://dx.doi.org/10.1016/j.displa.2010.12.004

Khundrakpam Johnson Singh receives his bachelor of engineering degree in Computer Science and Engineering from KBNCE, Gulbarga, Karnataka, in 2010. He completed his master of technology degree in Computer Science and Engineering from Dayananda Sagar College of Engineering, Bangalore, Karnataka, in 2012. He is currently pursuing Ph.D degree in Computer Science and Engineering at National Institute of Technology, Durgapur, West Bengal. He is interested in carrying out the research work in server security. Other area of interest includes various network issue and security, pattern recognition. Currently he is working in National Institute of Technology, Manipur as Assistant Professor in Computer Science and Engineering department.

Tanmay De received his BTech in Computer Science and Engineering from University of Calcutta, India, in 1996, MTech in Computer Science and Engineering from the Jadavpur University, India, in 1998, and PhD degree in Department of Computer Science and Engineering, Indian Institute of Technology (IIT), Kharagpur, India, in 2010. Since 1998, he has been a faculty member of National Institute of Technology (NIT), Durgapur, India. Tanmay De is presently an Associate Professor in the Department of Computer Science and Engineering at National Institute of Technology (NIT), Durgapur, India. His research interests include optical WDM networks, mobile ad hoc networks, and delay tolerant networks. He has published several research papers in various international journals and conference proceedings. He is a member of the IEEE, USA.