JACET Journal of Advances in Computer Engineering and Technology, 1(3) 2015

HF-Blocker: Detection of Distributed Denial of Service Attacks Based On Botnets

Bita Amirshahi[1], Ali Ahangari[2]

Received (2015-08-21)
Accepted (2015-10-30)

Abstract **- Today, botnets have become a serious threat to enterprise networks. By creation of network of bots, they launch several attacks, distributed denial of service attacks (DDoS) on networks is a sample of such attacks. Such attacks with the occupation of system resources, have proven to be an effective method of denying network services. Botnets that launch HTTP packet flood attacks against Web servers are one of the newest and most troublesome threats in networks. In this paper, we present a system called HF-Blocker that detects and prevents the HTTP flood attacks. The proposed system, by checking at the HTTP request in three stages, a Java-based test, check cookies and then check the user agent, detects legitimate source of communication from malicios source, such as botnets. If it is proved the source of connection to be bot, HF-Blocker blocks the request and denies it to access to resources of the web server and thereby prevent a denial of service attack. Performance analysis showed that HF-Blocker, detects and prevents the HTTP-based attacks of botnets with high probability.**

Keywords **- botnet, web servers, DDoS attacks, HTTP, HTTP Flood**

1- Department of Computer Engineering, Payam Noor University, Tehran, Iran (amirshahi@tpnu.ac.ir)
2- Department of Computer Engineering, Payam Noor University, Tehran, Iran (ahangari66@gmail.com)

I. INTRODUCTION

Massive growth in the use of Internet technologies in various aspects of life, forms many people habits. But these technologies abuse by some hackers and cyber crinimals to carry out crimes such as spam and distributed denial of service attacks [1]. Botnet, as a tool to launch distributed attacks, means a collection of infected computers that is controlled and driven by an cyber crinimal [2]. Set of the computers that are infected by malicious bot code, named botnets and any infected computers, named zombies. The zombies are controlled by an attacker remotely [3]. Botnets with large-scale are used to carry out activities such as a large spam, install spyware, virus and distributed denial of service attacks [4]. Distributed denial of service attacks based on botnets are launched by using a network of controlled computers. DDoS attacks often take advantage of the weakness of the network layer. Flooding the network with packets ICMP, SYN and UDP is one example of denial of service attacks at the network layer. Such attacks limit victim's bandwidth and system resources and thereby prevent legitimate and normal requests. By definition, in a distributed denial of service attack, attackers attempt to prevent users from accessing their interested services [5]. Flooding web server with http packets is one example of this type of attack that are one of the latest attacks and threats on networks [10]. The recent denial of service attacks are made up with using botnets of tens of thousands of victims. To circumvent detection solutions of distributed denial of service attacks based on botnest, attackers mimic the Web browsing bihavior of a large number of clients and as a result new attacks are made up.

In this way, botnets target high level expensive resources such as CPU, memory and database. The resulting attacks are hard to defend against using standard techniques as the malicious requests differ from the legitimate ones in intent but not in content.

According to the definitions and facts mentioned and in order to defend against the HTTP flood attacks of botnets, this article suggests a new way. The proposed method called HF-Blocker utilizes difference between malware and malicious traffic from legitimate and normal one and provides three consecutive authentication and without user interaction. The main idea of the proposal, is detection of the source of the connection with checking three major differences between the browser and malicious code.

II. RELATED WORK

In 2005, Matthias Jacob et al provided [7] the design and implement of a kernel extension called Kill-Bots to protect Web servers against DDoS attacks that masquerade as flash crowds. In other words, Kill-bots distinguishes human users from zombie machines by presenting a graphical test. Instead of authenticate clients, Kill-Bots, based on whether or not they solve the graphics tests, utilizes the test to identify IP address of the attacker machine. This feature allows Kill-Bots block malicious requests and still provide access to real users. As second feature, Kill-Bots sends the test to client and then checks its response to client without allowing for access sockets, TCB and worker processes. This feature protect process of authentication against DDoS attacks.

In 2009, Manimaran et al [8] in a project named JUST-Google address the problem of botnet based DDoS attacks with using Google's position as the first choice of many Internet users. In this study, a defense service against DDoS attacks by Google is provided to users as follows:

When a search request which ultimately leads to in question website, is received should not show the URL of the site in result page (the first step). Instead, a different URL is provided to the search request (the second step). With the URL provided, Just-Google considers the user as a node that controls by Google and then show a Web page to the user (the third stage). This page is a graphical test aimed at resolving by the user (the forth IV). The page enables the node of controlled by Google to distinguishes between bot code and real users. Once the correct response is received from user (step five), the node inform the client IP address to compromised web site(step six). Then the IP address is inserted to whiteliste. Finally, The node shows the actual URL to user.

Another way to deal with DDoS attacks based on botnets called Phalanx [9]. In Phalanx project, a packet-based capability is used to identify packets are allowed to pass through loop filtering. One of the weaknesses of Phalanx, additional delays due to the three-step routing packets through the mail boxes. Thumbnail project similar to the node or nodes that real users will steer clear where users must provide their own authentication. This authentication is based on a graphical test. Dyksn Klein and colleagues demonstrated that the proposal Flnks any additional delay to the actual traffic flow after Authentication clients not enter because after authentication, routing traffic on the three-step authentication of the real does not apply.

The design uses a three-step authentication request to separate the healthy from malicious request is generated by botnets used.

The proposal is similar to the Kill-Bots except where for separating normal HTTP request and malicious HTTP request is not used graphical test and user is not involved in the authentication process. HF-Blocker uses a three-step authentication for separating the normal requests from malicious requests that are generated by botnets.

III. THE PROPOSED SCHEME: HF-BLOCKER

Cyber attackers are now able to mimic the behavior of real and legal users in the widely distributed are that The resulting attacks are hard to defend against using standard techniques as the malicious requests differ from the legitimate ones. In this section, we present a system that will able to protect web servers against botnet-based denial of service attacks by seperating malicious HTTP requests from the normal ones. The proposed scheme is based on the fact that malicious codes

that send HTTP traffic to a web server, don't have some parameters and can't they will not able to perform some interactions with user. For this purpose, the system of detection and prevention is designed . In Figure 1, the overall operation of HF-Blocker, can be seen.

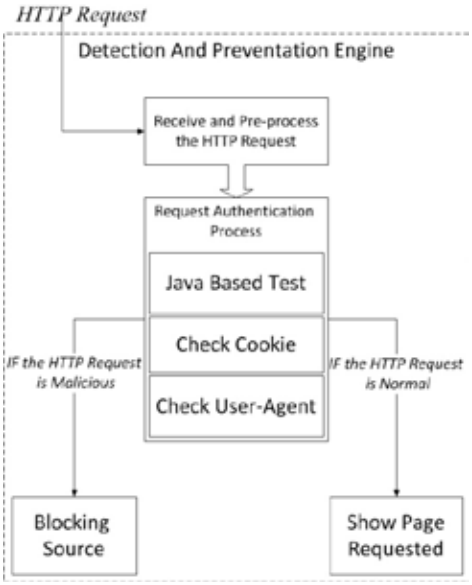

Fig. 1 HF-Blocker overall operation

A. DETAILS

This section outlines main components of the proposed scheme. First, request pre-processing component, then assessment component and Finally, connection tracking component will be discussed.

The request pre-processing component: As can be seen in Figure 2, all outgoing HTTP requests from the client, will be received by the pre-processing component. After receiving the requests, source IP address are registered by this component and then will be compared against both trust and blocked IP addresses list. In addition to a list of addresses IP, the component will record the result of assessments done in the assessment phase. Being trust IP address depends on the positive result of all the assessments that performed on the requestes. In other word, The request pre-processing component will check registered results in the IP Addresses list and just if that all of the assessment results is positive, then will send the received requests from that IP to the connection tracking component and otherwise, if at least one of the assessment result is negative, the request is blocked. If the source IP address is already in none of the list of IP addresses authorized and unauthorized, means a new request that has so far not been investigated. In this case, the IP address is registered in the database and a record that contains source IP address, Java-State, Cookie-State, UserAgent-State, ConnectionCounter and Random-Value is made. Java-State, Cookie-State and UserAgent-State fields contains the assessment results done by assessment component. Both Random-Value and ConnectionCounter fields are also used by the connection tracking component used that will be discussed in descriptions of these components. After recording the IP address of source and making a record with these fields, a HTTP cookie is provided to the resquest by HF-Blocker.

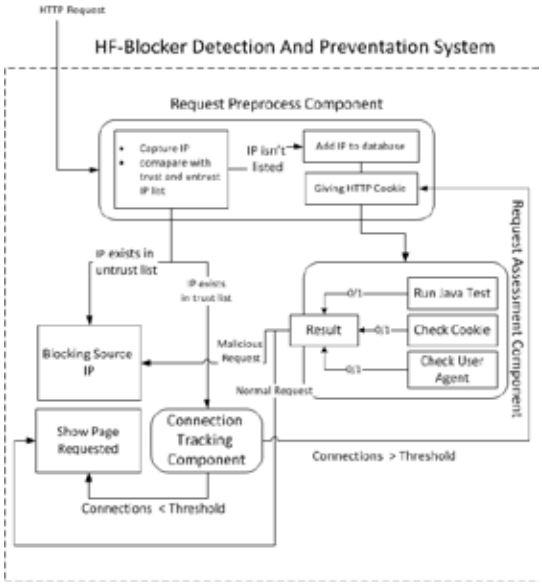

Fig. 2 HF-Blocker internal components operation

The assessment component: the requests of a legitimate user that are trying to view a Web site, send and receive through the web browser. In fact the browser plays role of interpreter for the code sent to client. One of the major differences between the browser and the code is Java language understanding [11,13]. The browser is able to detect and analyze Java code sent by server, while the code is not able to understand Java. The suggested system uses the difference between the browser and the code and after that, records IP and deliver cookies to the browser, and finally, as the first of the assessment component, two random number, α and β, sends in the form of Java code. If the sum of these two numbers is as expected, means that the source is a legitimate user or a browser and otherwise

means to be malicious source such as a bot code. As a result, HF-Blocker will determine whether the correct answer . If the answer is as expected, then positive value will be recorded in the Java-State field and otherwise negative result will be recorded in the field of Java-State. The first assessment of our proposal is same as the Kill-Bots project with the difference that Kill-Bots, in order to authenticate a user, uses a graphical test called CAPTCHA. Instead of using of the time-consuming and boring test, we use the simple and easy test for sending to client system.

The great advantage of this Java-based test is that doesn't require for user interaction to resolve. The Java-based test in this project can be solved by the browser and sent to the server back. After examining the reply sent by the browser, server begin the second phase. In this phase, as mentioned before, with arrival of HTTP requests to server, a HTTP cookie will be sent to client and then if it successfully passes assessments, in subsequent connections, client won't need to be assessment and will be able to consume the server resources. In the similar operation and after solving of the graphical test, killbots also provide HTTP cookies to the client. Before explaining the difference between the idea of offering in HF-Blocker and Kill-Bots, it need to be followed one of the differences between browser and code. Browsers establishing a session with Web server, will be able to understand cookies and set them. A cookie is a message that the server sends to the browser [12]. The browser stores the message in a text file and After that, every time that a user visits the server or opens a web page on which the server is placed, it will return the message to the server. The main purpose of cookies is to identify users and serve them based on their user settings on websites, such as Yahoo, that has done on previous connections. In HF-Blocker, offered cookies is made with two purposes. For the first goal, delivering cookies can help to speed up the future clients connections and to needless client for unnecessary assessments.

But the main difference between offering cookies in this project and Kill-Bots can be understanding cookies by the browser's. As the second purpose, in the proposed method, we exploit the brower's feature and evaluate the cookies in the second assessment phase. In the first phase, as soon as receipt of a new request, a HTTP cookie will be sent to client and then, in the second phase, is checked whether the request sent by the client has a cookie or not. In other words, check whether the client have HF-Blocker provided cookies or not. If the answer is positive, it means that the source of communication is a browser, and if the answer is negative, possibility to being the bot malicious code is enhanced.

As a result of the evaluation, positive or negative value, in the field Cookie-State will be recorded. In the last stage of assessment in the assessment component, the user agent of the request is examined. The user agent is a string which provide to web server many information such as operating system, browser name and version and other information provided in this field [13]. For example, some websites present user information on the part of the site such as address IP, the type of browser and so on or as another example, in some cases, if the user connects to the site with a older of the browser version, such as Internet explorer, to connect will be notice that in order to being better display and upload, please use another browser. The information appears due to analysing the user agent. In some cases, malicious codes avoid to provide the user agent to the server and send an empty string or invalid value as the user agent string. The final step is to the client's user agent. If the client has no valid user agent, a positive value and otherwise a negative value will be assigned to the UserAgent-State field. In this component, a function called AuthenticationResult is considered that is responsible for evaluating the results or checking values in the three fields Java-State, Cookie-State and UserAgent-State. If all the values of these fields is positive, the page requested is shown to the user and otherwise the source IP address is blocked.

Connection Tracking component: In order to bypass the detection and prevention tool HF-Blocker, a Botmaster can first masquerade itself as a trust client and after that being in the list of trust IP addresses, manage to run its interested bots. The purpose of designing connection tracking component is to cover this weakness. This component deals with requests that have passed all assessments successfully. By using the component, requests that have successfully passed all assessments, is tracked and re-evaluated after establishing several connections. For this

purpose, the connection tracking component uses two parameters for each IP address, the number of requests and threshold. For each IP address, the threshold parameter is randomly generated, between two numbers, and for each request, the number of requests parameter is increased. Every time that the component receives a request, the number of requests parameter corresponding the source IP is compared with the threshold. If the number of HTTP sent requests is less than the threshold, the request is redirected to the requested page and otherwise, the request is came pre-processing component back to be evaluated again after receiving cookies.

The use of a random number to determine the threshold of each connection prevents to being prediction the number by botmaster. If the botmaster could predict the threshold, can set the number of requests sent by botnets less than that and perform DDoS attacks. Therefore, the threshold is randomly generated and not recognizable. The randomly generated threshold value between two number is default and therefore it can be adjusted by Web server as needed. Algorithm of HF-Blocker has shown in Table I.

TABLE I
DATA COLLECTED IN THE ASSESSMENT PROCESS OF HF-BLOCKER

Number of Zombies	38	75	75	79
Number of HTTP Requests Sent by each Bot	20	100	500	1000
Botnet Acivity Period	00:09:07	00:08:40	00:15:22	00:25:49
Totall of Malicious HTTP Request	414	4455	16228	35352
Number of Malicious HTTP Request Detection	411	4436	16193	35319
Average of HTTP Packets Size	350.254	348.965	341.087	343.091
Totall HTTP Packet Size	290010	3099855	33519293	72539092
Delay Between the First and Last HTTP Packet	547.445 sec	338.939 sec	806.864 sec	1329.547 sec
Destination Port Used by Zombies	80	80	80	80
Source Port Used by Zombies	Random Range	Random Range	Random Range	Random Range
Correct Detection Rate	% 99.27	% 99.57	% 99.78	% 99.90

IV. EVALUATION

HF-Blocker detection and preventation system has been evaluated in the Internet and by using botnet real traffic.

A. Lab environment

Web Server: our lab Web server has 3.19GHZ Pentium processor and 2GB of RAM. The operating system of Web server is Linux, CentOS6.4 and its web server is the Apache version 2.2.8.

Requests sent to Web server: In order to evaluate HF-Blocker performance against DDoS attacks, an actual botnet with ability to carry out the HTTP flood attacks is used. Bot number of bots of the botnet that is distributed in different areas of the Internet is more than 78 bots that each one have the ability to send up to 10,000 requests to the server. The botnet has also ability to send up to 10 concurrent requests.

Setting up the lab environment: HF-Blocker detection and preventation system, as shown in Figure 3, is implemented on the web server with bandwidth of 100 Mbps. Also, in this experiment, Wireshark packet sniffer and analyser [15] is used to collect data. The experiment data using HTTP requests of the real botnet real in different intervals in Table 2, is collected.

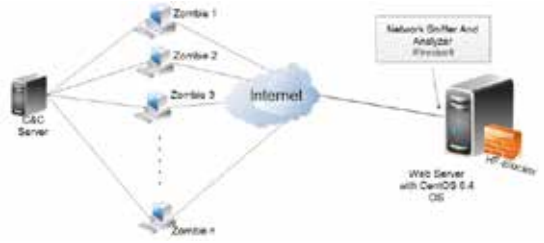

Fig. 3 Lab environment

B. Performance HF-Blocker

HF-Blocker detection and prevention system has been evaluated in two parts. In the first part, only the detection ability of HF-Blocker and in the second part, both HF-Blocker features include detection and prevention, has been evaluated.

Performance of the detection of HF-Blocker: In this section, only the rate of HF-Blocker detection system has been tested and is not involved in its prevention. In this evaluation, the variable TN, as the total number of malicious HTTP requests, variable TB , as number of the malicious request detected by HF-Blocker and variable DH, as HF-Blocker detection rate is assumed.

TABLE II
HF-BLOCKER OPERATION ALGORITHM

HF-Blocker Algorithm

```
/*HF-Blocker Operation*/
Capture(Src-IP)
IF Exist SRC-IP in Block-List
        Block(IP)
        Else IF Exist SRC-IP in Trust-List
                Connection_Tracking(Connection)
Else
        Register(IP)
        Give(HF-Blocker-Cookie)
        Generate(Threshold)
        Authentication(Connection)
Authentication(Connection):
        Step 1: Send JavaTest  /* Send To Clinet*/
                IF Result of JavaTest == True
                        Java_State = True
                Else  Java_State = False
        Step 2:  Check  HF-Blocker-Cookie  /*For  Recieved
Connection*/
                IF Cookie == HF-Blocker-Cookie
                        Cookie_State = True
                Else        Cookie_State = Flase
        Step  3:  Check  User_Agent  /*For  Recieved
Connection*/
                IF User_Agent <> " "
                        UserAgent_State = True
                Else UserAgent_State = False
        IF  Java_State==True  &&  Cookie_State==True  &&
UserAgent_State==True
                Insert IP into Trust-Liste
                Redirect Connection To RequestedPage.php
        Else
                Insert IP into Block-List
                Block(Connection)
Connection_Tracking(Conncetion):
        Number of Connection + 1
        IF Number of Connection > Threshold
                Authentication(Connection)
        Else        Redirect Connection To RequestedPage.php
```

With the above assumptions and with calculatuing rate of detection based on DH = (TB / TN) * 100, as shown in Table 2, the rate of detection of HF-Blocker is more than 99.26%, which represents a very high accuracy and detection for HF-Blocker system. Based on this evaluation and the values obtained of DH, it can be concluded that with increasing TN, DH rate is also increased. The evaluation showed that with growing up the number of malicious HTTP requests, HF-Blocker accuracy is also increased.

Complete HF-Blocker Evaluation: In the second part of the evaluation, HF-Blocker is fully evaluated.

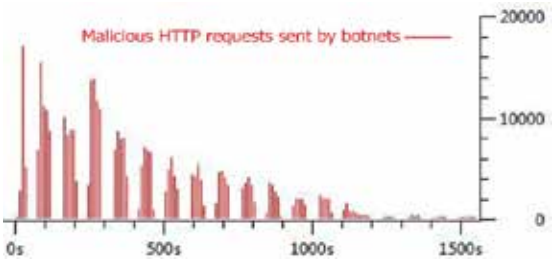

Fig. 4 Web Server without using HF-Blocker

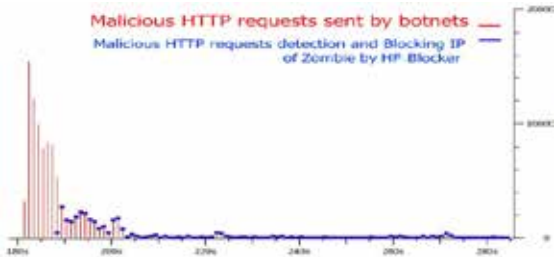

Fig. 5 Web Server with using HF-Blocker

In addition to evaluating the performance of the detection, the prevention performance of the system is assessed. According to Figure 4, botnet attacks started at t = 10 and ended at t = 1500. During this period and peak of attacks, β value increased to 10000, 14000, 16000 and also. In evaluating the Web server using HF-Blocker, based on Figure 5, botnet attacks at t = 182 started and ended at t = 285. At t = 188, HF-Blocker system, after evaluating requests, starts for blocking the IP address of zombies of the botnets. At the beginning of the process of prevention by HF-Blocker, the value of β is reduced from 6000 to 3000 requests per second. At t = 203, evaluating requests has been completed and the IP address of most of the zombies has been blocked. Since HF-Blocker system via blocking the IP address of zombies doesn't allow them to access the TCB, Worker Process and other resources of the Web server, from t = 203, β value is strongly reduced and in the most of points also reaches 0.

V. CONCLUSIONS

Distributed denial of service attacks based on botnets are dangerous and harmful threats that can leave heavy damages to organizations, companies, the public sector and military. Botnets using thousands and even millions of machines able to mimic the behavior of legitimate users and in this way they can evade the security systems on the network and carry out attacks they want. This article showed that HF-Blocker detection and prevention system is an efficient way to defense against the threats. HF-Blocker, unlike similar projects, instead of using graphics tests, evaluates HTTP requests sent to the Web server without the user interaction and thus seperates HTTP requests sent by. Evaluation of HF-Blocker system resulted high efficiency of the system against attacks and HTTP floods with very low incorrect detection rate.

REFERENCES

[1] H. Binsalleeh, On the Analysis of the Zeus Botnet Crimeware Toolkit, National Cyber Forensics and Training Alliance Canada, 2010.

[2] H. R. Zeidanloo, A. A. Manaf, "Botnet command and control mechanisms," In the proc. of Second International Conference on Computer and Electrical Engineering, (ICCEE '09), pp. 564-568, 2009.

[3] Paul Barford, Vinod Yegneswaran, An Inside Look at Botnets, Computer Sciences DepartmentUniversity of Wisconsin, Madison, 2006.

[4] C. Douligeris and D. N. Serpanos, "Network security: current status and future directions," Wiley-IEEE Press, 2007.

[5] B. B. Gupta, M. Misra, R. C. Joshi, ―FVBA: A Combined Statistical Approach for Low Rate Degrading and High Bandwidth Disruptive DDoS Attacks Detection in ISP Domain, In the proceedings of 16th IEEE International Conference on Networks (ICON-2008), DOI: 10.1109/ ICON.2008.4772654, New Delhi, India, 2008.

[6] D. Dagon, G. Gu, C. P. Lee, and W. Lee, "A Taxonomy of Botnet Structures," In the Proc. of ACSAC 2007, Miami, FL, USA.

[7] S. Kandula, D. Katabi, M. Jacob, and A. Burger, "Botz-4Sale: Surviving DDos Attacks that Mimic Flash Crowds," inProc. USENIX NSDI 2005. Boston, MA, May 2005.

[8] Al-Duwairi B, Manimaran G (2009) JUST-Google: A search engine-based defense against botnet-based DDoS attacks. IEEE International Conference on Communications (ICC '09).

[9] C. Dixon, T. Anderson and A. Krishnamurthy, "Phalanx: Withstanding Multimillion-Node Botnets,"In Proc. Of NDSI 2008.

[10] D. McPherson, ―Worldwide Infrastructure Security Report," Arbor Networks, January 19th, 2010, avilable at: http://ipv6.org.sa/sites/default/files/World_ Infrastructure_Security_Report_2011.pdf.

[11] Eli Tilevich, Yannis Smaragdakis, "Appletizing: Running Legacy Java Code Remotely From a Web Browser", International Conference on Software Maintenance and Evolution (ICSME), 2005.

[12] Ari Juels, Markus Jakobsson, "Cache Cookies for Browser Authentication", Security and Privacy, 2006 IEEE Symposium on , May 2006.

[13] Stefan Frei, Thomas Duebendorfer, "Understanding the Web browser threat: Examination of vulnerable online Web browser populations and the "insecurity iceberg", 2008.

[14] Ahmad Mudhar, "Evaluation of the CSF Firewall", 2013.

[15] Laura Chappell, "Wireshark Network Analysis, The Official Wireshark Certified Network Analyst Study Guide", 2009