iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 2 **Issue:** XI **Month of publication:** November 2014

DOI:

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)

Mitigation of HTTP-GET flood Attack

Hally Khatri[1], Akanksha Gupta[2], Dheeraj Pal[3]
Department of Computer Science,Amity University
Gwalior, Madhya Pradesh

Abstract- An HTTP GET flood is a volumetric attack that does not use malformed packets, spoofing or reflection techniques. HTTP flood attacks may be one of the most advanced non-vulnerability threats facing web servers today. It is very hard for network security devices to distinguish between legitimate HTTP traffic and malicious HTTP traffic, and if not handled correctly, it could cause a high number of false-positive detections. Rate-based detection engines are also not successful at detecting HTTP flood attacks, as the traffic volume of HTTP floods may be under detection thresholds. Because of this, it is necessary to use several parameters detection including rate-based and rate-invariant. Denial of Service attacks to web services is called HTTP-GET flood attack and threats of them increase day by day. In this type of attacks, malicious clients send a large number of HTTP-GET requests to the target web server automatically. Since these HTTP-GET requests have legitimate formats and are sent via normal TCP connections, an intrusion detection system (IDS) cannot detect them. In this paper, we propose mitigation techniques of HTTP-GET flood based on log file

Keywords— SSL, Intrusion Detection, HTTP-GET, Intrusion Detection Systems

I. INTRODUCTION

web-browsing is based on GET requests generated by sender. For two-node communication,sender repeats sending GET requests to receiver; who in return replies with requested data. In this type of attack, attackers abuse a HTTP-GET request protocol and they send a large number of malicious request packets to a target server [1]-[2].The difficulty in detection of the attack stems from legitimate nature of attacking hosts. Rather than sending ill-formatted network packets, attackers make their zombies comply with computer network regulations and request objects as they appear at pages, pretending to be legitimate users. In addition, low rate arrival of zombies and their request frequency, make them look as system-friendly connections for rate-based Intrusion Detection Systems (IDS).Scenario for running the HTTP-GET attack is alike with standard techniques used by attackers performing Distributed Denial of Service (DDoS) attack[9]. The layer 7 attacks require more understanding about the website and how it operates. The attacker has to do some homework and create a specially crafted attack to achieve their goal. Because of this, they require less bandwidth to take the site down, and are harder to detect and block.The way of HTTP-GET flood attack is classified as follows[8].

Computer Virus : Client machines that are infected with computer virus launch the distributed HTTP-GET flood attack. In this type of attack, packet parameters such as a target URL,an access interval, and an attack term are previously decided by a computer virus code.

Bot : The Bot is a malicious program that launches HTTPGET flood attacks as with a computer virus. But unlike in the case of computer virus, attacker can modify packet parameters as needed, thus more dynamic attack is possible.

DoS tool : The tool for DoS attack is taken to the public on the Internet. In this attack, the attack parameters are decided depending on the attacker.

An IDS is an effective tool to detect network attacks and malicious activities. The IDS detects attacks by monitoring packets by using attack signatures [3]. However, because the HTTP-GET flood attack use a normal HTTP protocol, it
is difficult to detect the HTTP-GET flood attack by using IDS. A bandwidth controller is used against the HTTP-GET flood attack. The bandwidth control intercepts the traffic that exceeds the tolerable quantity of the server to protect quality of web services [4]-[5]. However, this method limits not only the attacks but also the normal traffic. There is an anomaly detection module against HTTP-GET flood attack [6]. This is a quite simple threshold-based technique that detects the clients who request same page more than a few times in a second. However, this method can be easily evaded by changing the packet parameters, and it also includes high miss-detection. In this paper, we try to detect the HTTP-GET flood attack based on the analysis of page access behavior. We propose two HTTP-GET flood attack detection algorithms. One is analysis of browsing order of pages, and the other is analysis of correlation with browsing time to page information size. We implement an attack detection system on the test bed server and

evaluate the attack detection rates, i.e., false positive and false negative. The results show that we can detect the HTTPGET flood attack quickly with a few access logs, and it also can detect a dynamic HTTP-GET flood attacks.

II. PROPOSED ATTACK MITIGATION METHODS

In this section, we describe proposed HTTP-GET flood attacks detection methods.
To create a resistance at the web application level against the HTTP flood attacks, the basic idea might be summarized into 3 steps:

- Detect Internet Protocol(IP) addresses of the abnormally excessive requests according to a previously defined rule,
- To reduce attack surface, return these requests with a low resource used response (like a blank page,Page not found),
- block detected IP addresses by using other components at the other mitigation levels (WAF, web server/service, etc.).

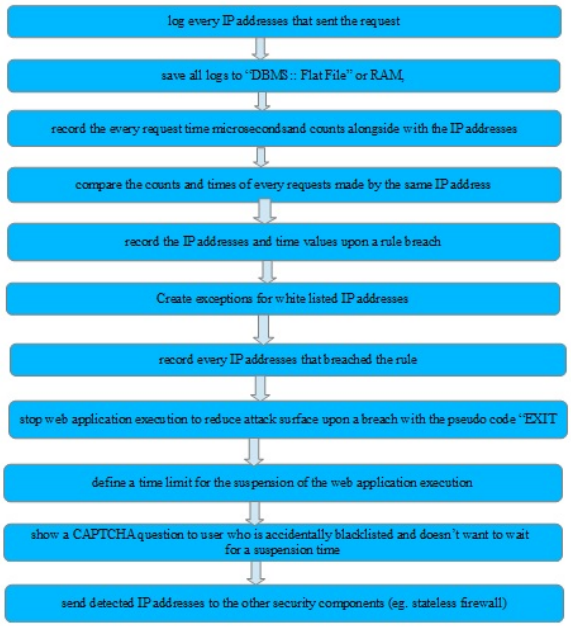


Figure 1.Flow Chart Flood Detection

While reducing attack surface by sending low resource used responses, this implementation will also save resources of the back end 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 back-end infrastructure members with other infrastructures like intranet or distributed web application servers. Saving the resources for the back-end infrastructure will prominently reduce the amplitude of the HTTP flood attacks.

It is a good security practice to bring the HTTP

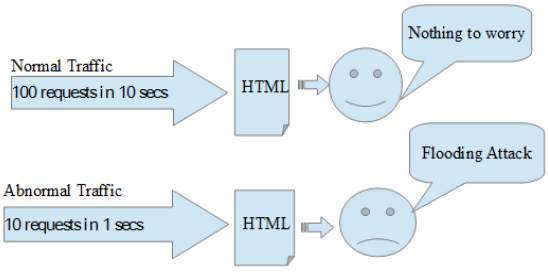


Figure 2.

flood attack awareness for the web application and implement additional precautions to every mitigation level including the web application level.

A. The Rule Creation Idea

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.

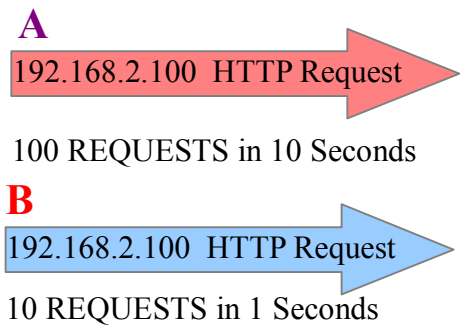


Figure 2- The Rule Creation Idea

In Figure 3, there are two HTTP request scenarios at the same mathematical ratio (10 requests/second). Nevertheless, they are not in the same characteristics. A detection rule written for the request “A” is always more tolerant (10 times) than the detection rule written for the request “B”. For example, a rule written for the request “A” can let 100 requests in a second. However, a rule written for the request “B” would not let it. Defining the normal usage traffic scenarios is really important at this phase of defining the detection rules. Weak rules can cause false positives.

The steps below are steps to design a web application level HTTP anti flood system to detect flooder IP addresses and reduce the attack surface against the HTTP flood attacks at the web application level:

- call a global script file/class/library before the every code related to the web application,
- log every IP addresses that sent the request,
- save all logs to “DBMS:: Flat File” or RAM,
- record the every request time microseconds and counts alongside with the IP addresses,
- compare the counts and times of every requests made by the same IP address,
- record the IP addresses and time values upon a rule breach,
- create exceptions for white listed IP addresses,
- record every IP addresses that breached the rule,
- stop web application execution to reduce attack surface upon a breach with the pseudo code “EXIT”,
- define a time limit for the suspension of the web application execution,
- show a CAPTCHA question to user who is accidentally blacklisted and doesn’t want to wait for a suspension time,
- send detected IP addresses to the other security components (eg. stateless firewall),
- notify the administrator via an e-mail.

B. A HTTP Flood Attack Scenario with Traffic Baselines and Rules

In order to accomplish a healthy rule base against HTTP flood attacks, the first step should be the defining the normal traffic.

The normal network traffic values on the web application are given below:
- maximum request count from a single IP address: 10 requests/second,
- the time between the closest two requests from a single IP address: 0.1 seconds.

These are the baseline traffic values for creating a rule against the HTTP flood attacks. This is the minimal information to create a healthy rule.
A basic abnormal traffic rule based on these baseline values could be sampled as "10 requests in 0.1 seconds". According to the normal traffic baseline values, 1 request in 40.000 microseconds from a single IP address will not be considered as a HTTP flood attack. The abnormal traffic rule above allows 1 request in 10.000 microseconds at 10 times from a single IP address. According to the rule creation concept, this rule also has a request count pointed out by "10 times" description. The request count can give an opportunity to mitigate false positives.
Besides HTTP flood attacks, this web application level implementation can provide an opportunity to slow down the Brute Force Attacks and Web Vulnerability Scanners. When the detected IP addresses shared with other security components, this also would provide an opportunity to block attackers' access to the web application.

III. CONCLUSION

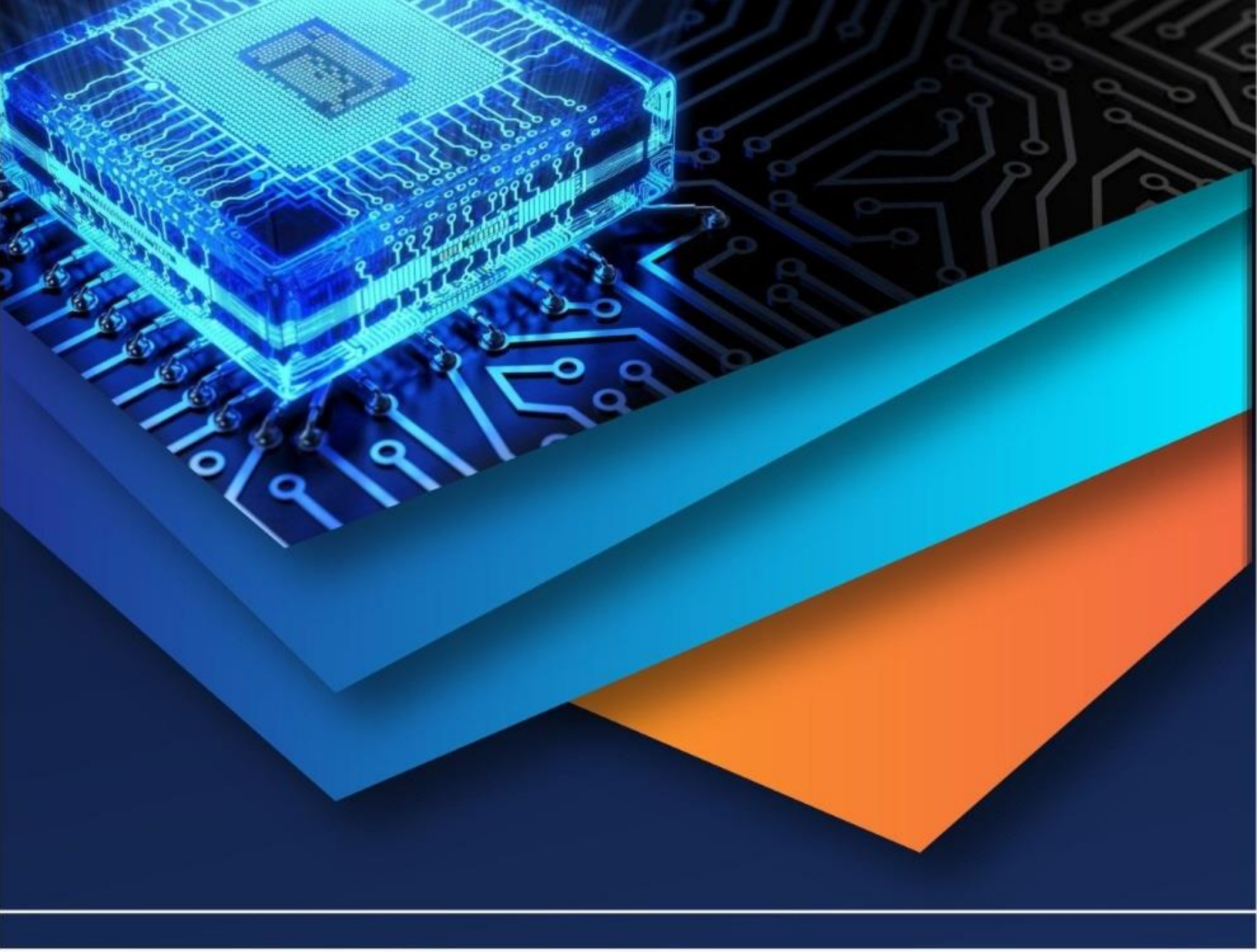
In this paper we have proposed an Mitigation methods of HTTP GET Flood Attack. The results of our approach are promising. Our Methods are based on traffic baseline(defining the normal traffic) and rules which detect the flood attacks and block detected IP addresses .We implement the attack detection system and evaluate a false positive and a false negative by using the access log of the Internet web server.When the detected IP addresses shared with other security components, this also would provide an opportunity to block attackers' access to the web application.In the future,we are aiming at developing a system that provides satisfactory detection rate, without defining the traffic baseline rule.

IV. ACKNOWLEDGEMENT

The authors would like to thank the anonymous referees for their helpful comments and suggestions to improve this work.

REFERENCES

[1] S.Byers, A. D. Robin and D. Korman, "Defending Against an Internet-Based Attack on Physical World", *ACM Transactions on Internet Technorogy,* vol.4 No.3, August 2004, Page 239-254.
[2] Juan M. Estevez-Tapiador, Pedro Garcia- Teodoro and Jesus E. Diaz- Verdejo, "Detection of Web-based attacks through Markovian protocol parsing", *Computers and Communications, 2005. ISCC 2005. Proceedings.10th IEEE Symposium,* 27-30 June.2005, Page 457-462
[3] Chie Ishida, Yutaka Arakawa, Iwao Sasase and Keisuke Takemori,"Forecast techniques for Predicting Increase or Decrease of Attacks Using Bayesian Inference", *IEEE Pacific Rim Conference on Communications,Computers and Signal Processing,* Victria, Canada, August 2005, Page 1088-1192.
[4]http://www.topology.org/src/bwshare/README.html
5]http://www.sakura.ad.jp/tanaka/apache/module/mod access limit.tar.gz
[6]http://www.zdziarski.com/projects/mod evasive/
[7] Wei Zhou Lu, and Shun Zheng Yu, "A HTTP Flooding Detection Method Based on Browser Behavior", *IEEE Computational Intelligence and Security, 2006 International Conference on,* Volume 2, Nov.2006, Page 1151-1154.
[8] Takeshi Yatagai, Takamasa Isohara, and Iwao Sasase "Detection of HTTP-GET flood Attack Based on Analysis of Page Access Behavior" 1-4244-1190-4/07/$25.00 ©2007 IEEE.
[9] Pawel Chwalinski, Roman Belavkin and Xiaochun Cheng "Detection of Application Layer DDoS Attacks with Clustering and Bayes Factors" 978-1-4799-0652-9/13 $31.00 © 2013 IEEE DOI 10.1109/SMC.2013.34



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY