Web Application Level Approach against the HTTP Flood Attacks IOSEC HTTP Anti Flood/DoS Security Gateway Module

While HTTP Flood and DoS attacks are spreading nowadays, there is a new attack surface reduction approach against these attacks called “Web Application Level Approach against the HTTP Flood Attacks”. If it is used properly with the conventional mitigation methods, the web application level approach against the HTTP flood attacks can save the day.

What you should know:

Basic understanding of DoS attacks

Knowledge of HTTP protocol

What you will learn:

Denial of Service mitigation method against HTTP Flood attacks at the web application level

About the author:

The author has over 12 years of field experience in the Information Security. He works as a consultant at PwC which is one of the big four.

In order to accomplish a denial of service state on systems, flood attacks aim to push limits of system usage to the out of boundaries determined by the normal usage scenarios. There may be a flood attack between the considered normal network traffic and the considered abnormal network traffic. The flood attack name can be determined by the specific protocol that attack is made on. For example, a flood attack on the DNS protocol is called as DNS Flood Attack while a flood attack on the HTTP protocol is called as HTTP Flood Attack. Since every protocol has its own technical architecture and vulnerabilities, flood attacks can differ on the attacking techniques from protocol to protocol.

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 (7^{th}) 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 (7^{th}) 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 (7^{th}) 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. Before starting a discussion on the web application level approach to the HTTP flood attacks, 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.

Mitigation Levels against the HTTP Flood Attacks

There are 5 main mitigation levels against the HTTP flood attacks:

- first level is the Cloud Services Level (ISP, Cloudflare, etc.),
- second level is the Network level (web application firewall),
- third level is the Web Server Level (host IPS),
- fourth level is the Web Service Level (mod_evasive, Dynamic IP Restrictions),
- and the fifth level is the Web Application Level (IOSEC HTTP Anti Flood Security Gateway Module).

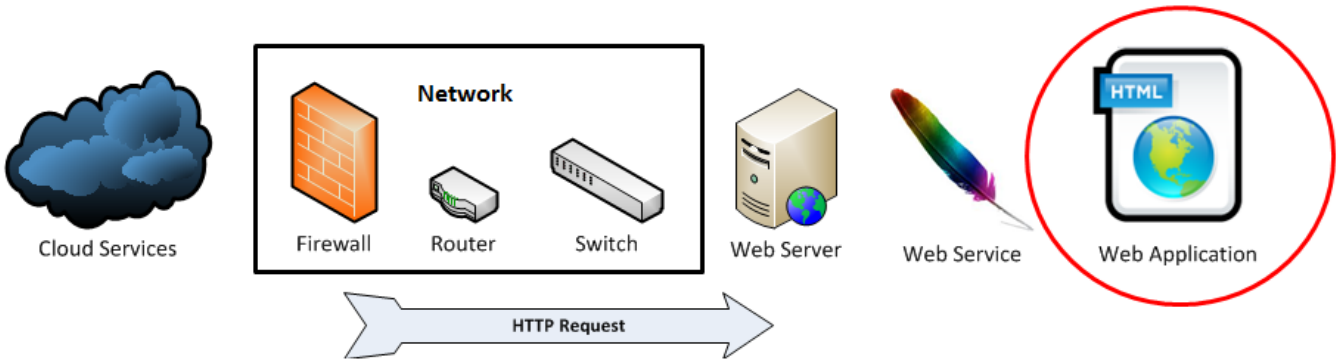

Figure 1 - Mitigation Levels of HTTP Flood Attacks

Every mitigation level has its own specific technical design to mitigate the HTTP flood attacks. The First and Second levels are the most important levels for handling the attack attempts. The Second level includes every network related component that comes before the web server. Even a WAF (web application firewall) which operates at the application (7^{th}) layer could be an example for this level. If it is possible, preventing the HTTP flood attacks at these levels is the best way. However, it is a good security practice to implement additional precautions to every level including the web application level.

Today's enterprise information technology infrastructures that are commonly used in corporations seem well prepared for attacks like SYN, UDP, DNS, ICMP, HTTP flood attacks. Of course, there are a lot of exceptions yet when it is only compared to the resistance of the HTTP flood attacks, this assumption drastically changes its way.

Unlike SYN flood scenarios, according to the 3-way handshake of the TCP protocol's design, HTTP requests cannot be spoofed. Therefore, it is easy to detect real flooder IP addresses. This is the basic idea for creating a web application level mitigation method against the HTTP flood attacks

The Basic Concept Idea of the Mitigation

To create a resistance at the web application level against the HTTP flood attacks, the basic idea might be summarized into 3 steps:

- detect 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 or else),
- block detected IP addresses by using other components at the other mitigation levels (WAF, web server/service, etc.).

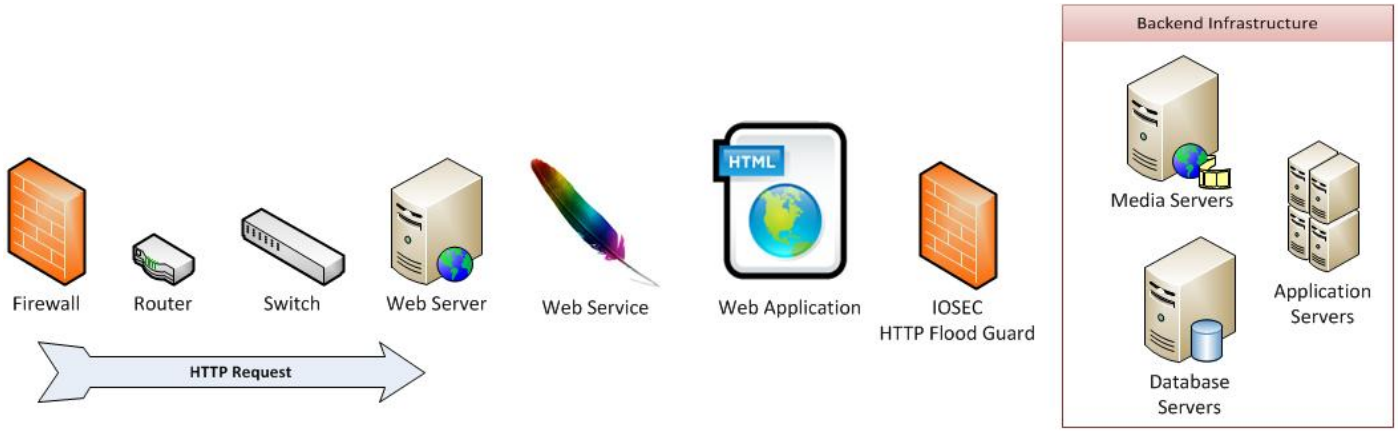

Figure 2 - Saving the Resources for the Backend Infrastructure

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.

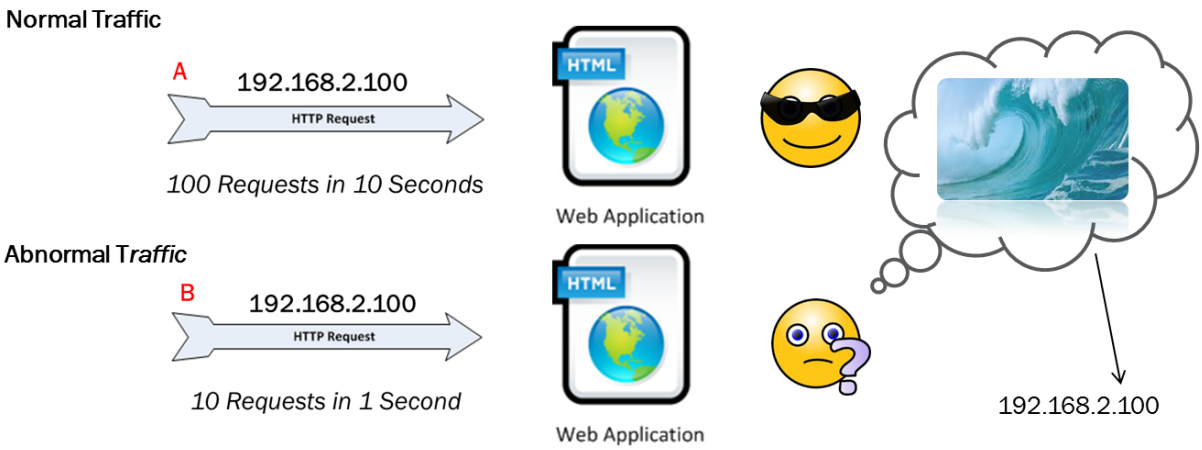

Figure 3 - HTTP Flood Attack Awareness for the Web Application

It is a good security practice to bring the HTTP flood attack awareness for the web application and implement additional precautions to every mitigation level including the web application level.

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.

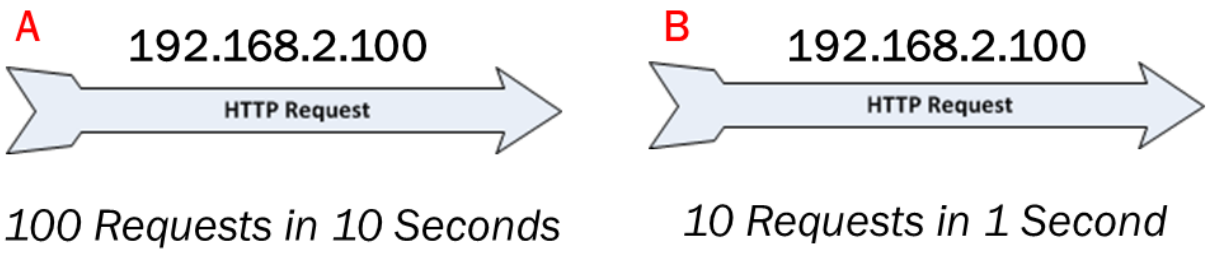

Figure 4 - the Rule Creation Concept

In Figure 4, 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 might be an example 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 psuedo code “EXIT”,
- define a time limit for the suspension of the web application execution,
- show a CAPTCHA question to user who is accidently 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.

Basic design for a HTTP flood guard might include these steps. Further information on project and proof of concept content may found at *http://www.iosec.org* and *http://sourceforge.net/projects/iosec* Internet addresses.

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: 5 requests/second,
- the time between the closest two requests from a single IP address: 0.2 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 tolerance factor pointed out by “10 times” description.

The tolerance factor (the request count) can give an opportunity to mitigate false positives. Many web development technologies like Ajax queries or HTML inner frames (iframe) may cause false positives without this tolerance value.

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.

IOSEC HTTP Anti Flood/DoS Security Gateway Module
|http://sourceforge.net/projects/iosec

Gökhan Muharremoğlu
Information Security Specialist
gokhan.muharremoglu@iosec.org

gokhan.muharremoglu@tr.pwc.com

http://www.linkedin.com/in/gokhanmuharremoglu