IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

SLOW-RATE DOS ATTACK DETECTION IN HTTP/2 PROTOCOL OVER TLS ENABLED TCP FOR MULTIPLE CLIENTS

[1]Anusha A Murthy, [2]Dr. K.N. Rama Mohan Babu, [3]Prathima Mabel J

[1]M.Tech-Student, [2]Professor & Head of the Department, [3]Assistant Professor

[1]Deptarment of Information Science & Engineering, Dayananda Sagar College of Engineering, Karnataka, Bangalore, India

Abstract: With the exponential growth of internet usage, based on user's demand there is a need for faster processing of requests with available bandwidth. The HTTP/1.1 protocol is a widely adapted communication protocol in the internet platform. Its frame structure makes it compact with website configuration. Considering the popularity of the HTTP/1.1, the upgraded version HTTP/2 protocol was found to have improved utilization of TCP with basic functionality of HTTP. The security for TCP is enhanced in the network by adding a TLS credential. By the concept of multiplexing, the HTTP/2 protocol can withstand payload overload. Hence HTTP/2 protocol provides a way of solving the traffic overload in the Internet Protocol layer caused due to flooding of packets. Here, the experiment is conducted on HTTP/2 server parameters such as connection time to the webpage from the server, the response-reply gap between server and clients, latency for a response from the server, inter-packet arrival time at the server and webpage load time. A slow-rate DoS attack called Slowloris was launched in the network supporting HTTP/2 protocol. With the analyzed server parameters, a threshold value is set for detecting and terminating attacker from the network. The results showed that with the launched attack over a network, HTTP/2 protocol is proved to be faster compared to the lower versions of HTTP.

Index Terms **- HTTP/2, Multiplexing, TLS-TCP, Slowloris, Slow-rate DoS attack, HTTP/2 benchmark tool analysis.**

I. INTRODUCTION

A protocol is a set of rules or regulations that govern the data communication between web browsers and servers. The most widely and popularly adopted application protocol over the Internet is Hypertext Transfer Protocol (HTTP). The evolution of the HTTP protocol has the following versions:

a) HTTP/0.9:termed as the One-Line protocol
b) HTTP/1.0 :termed as Informational RFC protocol
c) HTTP/1.1:termed as Internet Standard protocol
d) HTTP/2: termed as Transport-level performance enhanced protocol

The HTTP/1.1 protocol performs optimizations such as keep-alive connections, encoding of chunked data, byte-range requests, and request pipelining. A limited connection between client and server makes a series of requests to wait for the completion of one particular request. This condition is termed head-of-line blocking in HTTP/1.1 and it is a major drawback. When multiple clients access the World Wide Web they often suffer from less internet speed due to overloading on the server [20]. This issue found in HTTP/1.1 was solved by HTTP/2 protocol by the process of parallel response-reply mechanism.

The HTTP/2 protocol has originated from the SPDY protocol developed by Google. According to the recent survey conducted in the year 2017 it was found that around 20% to 40% of websites such as Chrome, Firefox, and Edge use this protocol as the communication platform [2]. HTTP/2 helps in providing higher communication speed over client and server connection through multiple requests/responses by multiplexing feature as shown in Figure 1. This is achieved by breaking HTTP/2 messages into independent units called header, data, and control frame and finally reassembles them into a single message on a single TCP connection [13]. For all performance enhancements of HTTP/2, the binary framing layer explains the method of HTTP messages encapsulation and transfer between the client and server as shown in. By using streams, messages, and frames in a binary framing mechanism the data can be interchanged between client and server.

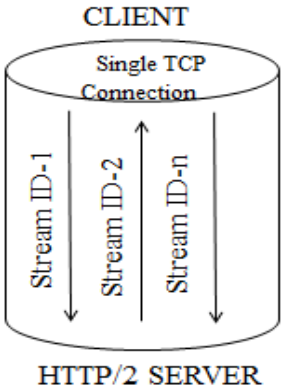

Figure 1: Multiplexing feature in HTTP/2 server through multiple streams

1.1 Functional feature of HTTP/2 protocol

1.1.1 Response and request multiplexing: When clients want to make a large number of requests, HTTP/2 enables the multiplexing feature by splitting a single HTTP message into independent frames.

1.1.2 Prioritization of steam ID: HTTP/2 splits the HTTP message into independent frames of different streams. These frames are prioritized based on stream ID.

1.1.3 Data flow control: Flow control is a directional mechanism to prevent receiving a huge set of data from the sender without prior knowledge of network capacity.

1.1.4 HTTP-server push: The ability of the HTTP/2 protocol to allow a single server to transfer multiple responses for a single client requests.

1.2 HTTP/2 protocol vulnerableness

1.2.1 Stream Multiplexing in HTTP/2: Here a TCP connection partitions are logical and can be manipulated leading to DoS.

1.2.2 Slow Attacks: Here it opens multiple connections at the victim server leading to a DoS attack.

1.2.3 Priority and Dependency: The stream prioritization consumes a lot of memory thus making server resources not available.

1.3 Common attacks over HTTP/2 protocol

1.3.1 Cross-protocol attacks: When an attacker end initiates communication in one type of protocol while the path of connection between client and server uses another type of protocol it triggers the cross-protocol attack.

1.3.2 Intermediary Encapsulation attacks: The attacker allows the HTTP/2 header to fill invalid names in the header field. This can exploit the feature of HTTP/2 to decode header values.

1.3.3 Denial of Service attacks: With the use of header compression and flow control feature there is a huge dependency on the available resources for processing a large set of stored data.

II. Related Works

M. Belshe R. Peon introduced the concept of HTTP/2 protocol in his work which explains the concept of multiplexing that utilizes TCP connection efficiently [1]. The investigation on HTTP/2 specifies the features of the protocol such as response multiplexing, server push, and header compression that makes it difficult for eavesdropping [9]. It was found that DDoS attack over HTTP/1.1 causes a similar threat in HTTP/2 by creating a lot of payloads [15]. The vulnerabilities of protocol resulted in excessive use of memory, reduced web throughput, and packet drop when a slow-rate DoS attack was launched [4]. According to Nikhil Tripathi, the HTTP/2 protocol is more vulnerable concerning slow rate DoS attack [6]. This new version of HTTP requires explicitly TLS support for better performance to secure data [10]. The investigation conducted by Imperva Defense center proved that server administration cannot directly switch to the new version of the protocol without an additional layer of security [11]. According to an investigation conducted by Cisco, working on HTTP/2 protocol over QUIC will help in faster browsing. The added feature of QUIC will support handshakes in connection [12]. The impact of HTTP/2 on web services is that the default encryption feature of HTTP/2 resulted in traffic hiding that affected many services such as web caching, traffic classification [17]. According to Mukhtar Abdillahi, HTTP/2 helps to save energy consumption for mobile devices by improving the performance of browsing [18]. The introduced HTTP/2 studies proved that the protocol can be adapted to cellular networks but also have a negative impact on packet loss in mobile technology [3]. A forensic

report on security breaches over web surfing proved that HTTP/2 is more useful in the cellular field [19].

III. System Design

The architecture of the system comprises virtual machines installed on a host machine. It consists of multiple clients and a server connected through a TCP connection via a Wi-Fi network having a 72Mbps link speed. The server VM is installed with Apache2 server with HTTP/2 enabled. The clients and server VMs are connected through a TCP connection having an SSL certificate, thus ensuring TLS support over TCP. Among the multiple clients, one is made as a malicious client. This malicious client and genuine clients process their requests to the server over TCP. On analyzing server parameters connection time to the webpage from the server, the response-reply gap between server and clients, latency for a response from the server, inter-packet arrival time at the server and webpage load time, a threshold value is set to detect malicious activity. When an experiment is conducted at random and the result is found greater than the defined threshold, that particular IP address of the attacker client is isolated from the communication network as shown in Figure 2.

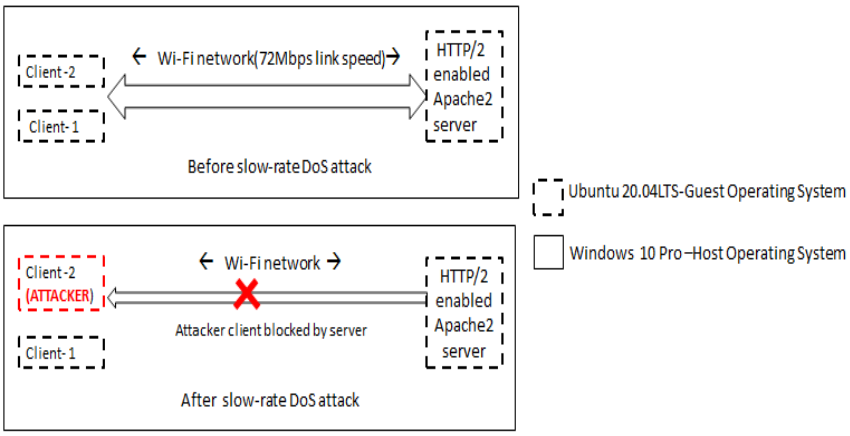

Figure 2: Architectural Overview of the designed prototype

3.1 Experimental set-up

3.1.1 Virtual Machine resource: A resource unit that uses software instead of a physical system to deploy applications is termed a Virtual Machine (VM). One or more VMs are allowed to run on a host machine. Each virtual machine includes a log file, Non-Volatile Random Access Memory, virtual disk, and runs on their own operating systems.

3.1.2 VMware Workstation: VMware Workstation is a hypervisor that can run either on Windows or Linux host operating systems. It enables users to install VMs on the host OS and allows them to work simultaneously with the host machine. The installation of any VMs requires an optical disc image file called ISO image that is mounted as virtual hard disk drives.

3.1.3 Network Connectivity: Wireless Fidelity (Wi-Fi) helps to transfer data or connect to the internet using radio bands for an electronic device. It is an elementary baseline for wireless local area networks. This network is connected to the host OS and the same network acts as a LAN and establishes a bridged network between VMs.

3.1.4 TCP/IP protocol: Transmission Control Protocol (TCP) is a connection-oriented protocol that allows the establishment of prior network connections before communication is initiated for the exchange of data between endpoints as shown in Figure 3.

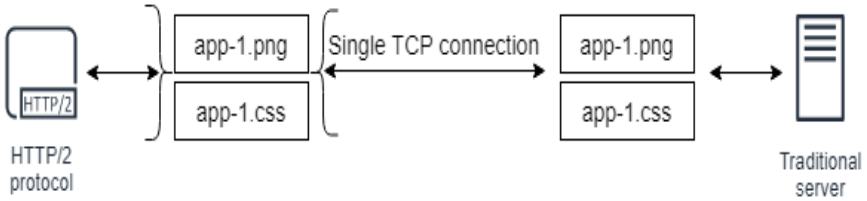

Figure 3: TCP connections in HTTP/2 protocol

3.2 Virtual Machine Configuration

3.2.1 Ubuntu 20.04LTS version: Ubuntu is free open-source Linux distribution software. The latest version is 20.10. LTS is used for server security, maintenance provides guarantees and updates in a long run of usage of its version.

3.2.2 Apache2 Web server: The most commonly used web server on the Linux platform is Apache2 Web Server. Web servers are used to serve web requests of clients using browsers such as Firefox, Chromium, or Internet Explorer. The protocol used to process web-request is HyperText Transfer Protocol (HTTP).

3.2.3 SSL support for Apache2 webserver: A security protocol that establishes encrypted links between a web server and a client over the internet is called Secure Socket Layer (SSL). It uses a Certificate Authority to identify both ends of communications sent over HTTP connection thus making a HTTPS secured connection between endpoints. This SSL support on Apache2 helps in establishing TLS over TCP link as shown in Figure 4.

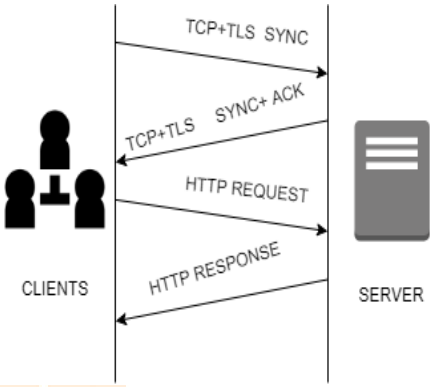

Figure 4: Representation of HTTP/2 protocol over TLS enabled TCP connection

3.3 HTTP/2 server parameter analysis benchmark tools

3.3.1 h2load benchmark tool: It is benchmarking tool that is used for HTTP/2 supported web calls. It also helps with verifying SSL/TLS certification. It uses a command to analyze a particular URL by using Equation (1)

$$h2load\ [flags]\ [URL] \tag{1}$$

where,
=> h2load: h2load command
=> flags: flags for the particular task assigned
-n, number of requests to perform for the benchmarking session across all clients.
-c, number of concurrent clients and denotes the number of multiple requests to perform at a time
=>URL: path URL required to test

3.3.2 Apache benchmark tool: Apache Bench (ab) is web-load testing which is a command-line computer program for HTTP or HTTPS webserver. This tool helps to know the number of requests processed per second by the webserver. It uses a command to analyze a particular URL by using Equation (2)

$$ab\ [flags.....]URL \tag{2}$$

where,
=> ab: Apache Bench command
=> flags: flags for the particular task assigned
-n, is the number of requests to perform for the benchmarking session
-c, is the number of multiple requests to be performed at a given time
=>URL: path URL required to test.

3.3.3 Fetching server parameters through benchmark tool

a) h2load benchmark is used to obtain parameters -response and reply gap between endpoints and inter-packet arrival time at the server

b) Apache benchmark is used to obtain parameters- server connection time to a webpage, requests processing time by the server, page load time, latency in getting a response from the server is calculated mathematically by using the formula referred to as Equation. (3),

$$Latency\ for\ response = (2 * requests\ processing\ time\ by\ server)/\ server\ connection\ time\ to\ webpage \tag{3}$$

3.4 Slowloris - Slow-rate DoS attack tool

Slowloris is a type of denial-of-service attack that allows an attacker to exhaust resources of the targeted server by opening many simultaneous HTTP live connections between the attacker and victim as shown in Figure 5. It is an application layer attack and falls under the category of low and slow rate Denial of Service (DoS). This type of attack uses a minimum amount of bandwidth and tries to consume all server resources and provides a slow response to requests. The targeted server will receive many threads to handle concurrent connections. Each thread attempts to stay active while waiting for the request to complete its cycle. When the server's maximum connection limit is exceeded it results in resource-hungry for the server and halts the data transfer between endpoints. The attack launched from the client-side sends small HTTP requests continuously to keep the server thread alive.

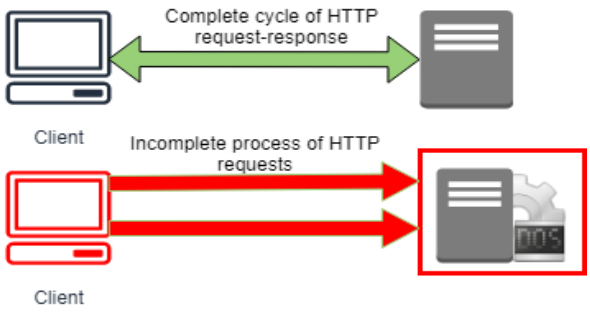

Figure 5: Slow-rate DoS attack on web-server

3.5 Configurations of experimental components

a) Windows 10 Pro 64-bit operating system is used as host OS

b) Ubuntu-64bit operating system is used as guest OS(s)

c) ISO image used is ubuntu-20.04.3.0-desktop-amd64.iso to install guest OS

d) Wi-Fi protocol with network band 2.4GHz and link speed 72Mbps is used for network connectivity

e) VMware Workstation Player 16.1.0 is used as VMware workstation

IV. ALGORITHM OF THE PROTOTYPE

The workflow of the conducted experiment to analyze the performance of the Apache2 HTTP/2 server by using server parameters using is explained as shown in Figure 6. Here, a server is connected to multiple clients in a network. Client-2 is intended to be an attacker that launches a Slowloris attack. The server script at the server end is executed, and parameters are obtained from the benchmark tool. With the obtained test runs, a threshold is set for parameters such as connect time to the webpage from the server and response latency from the server. If the parameter value is found to be greater than the set thresholds, it is identified as malicious activity in the network. After the client attacker source is identified the server blocks the route of that client thus breaking the connection between server and attacker.

Algorithm : Detection & Termination of DoS attack on HTTP/2 server

Step 1: Start
Step 2: Connect server to multiple clients
Step 3: Launch Slowloris attack from client-2
Step 4: Fetch server parameters from benchmark tool
Step 5: Set-up threshold for parameter connect time to webpage (or) response latency from server
Step 7: Check the obtained value is greater than set threshold for these parameters
Step 8: Valid ? Detect the attacker client IP
Step 9: Block the IP from server
Step 10: End

Figure 6 : Algorithm of the prototype

V. EXPERIMENTAL RESULTS

5.1 Execution of server parameters for HTTP/2 protocol

A series of test runs are conducted to set a threshold for attack detection. The results of the test run on selected server parameters are indicated graphically as shown in Figure 8(a)(b)(c)(d)(e)(f). The values of parameters highlighted in red color are the values obtained after the Slowloris attack.

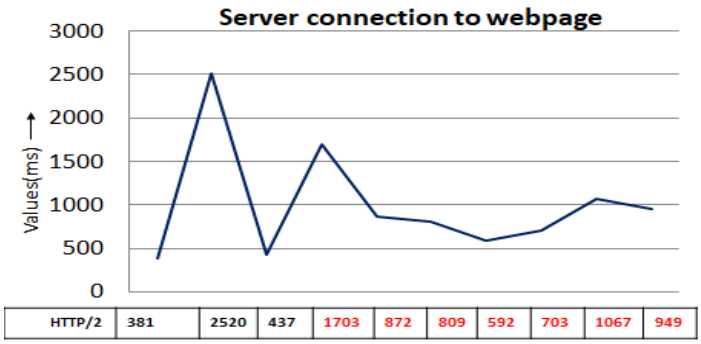

Figure 8(a): Server connection to a webpage in HTTP/2 protocol

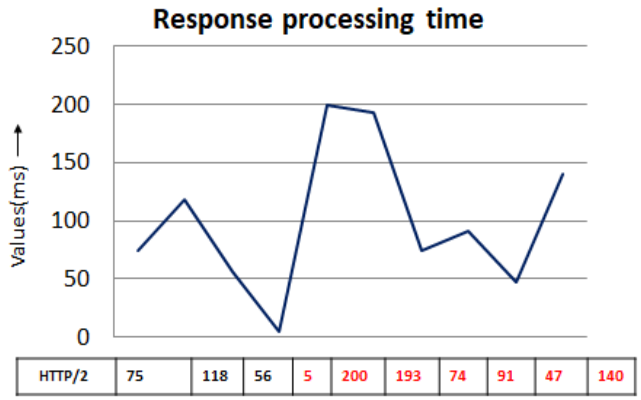

Figure 8(b): Response processing time in HTTP/2 protocol

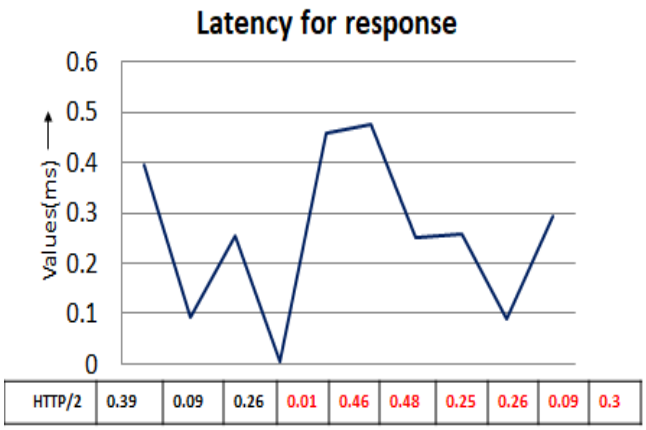

Figure 8(c): Latency for a response from the server in HTTP/2 protocol

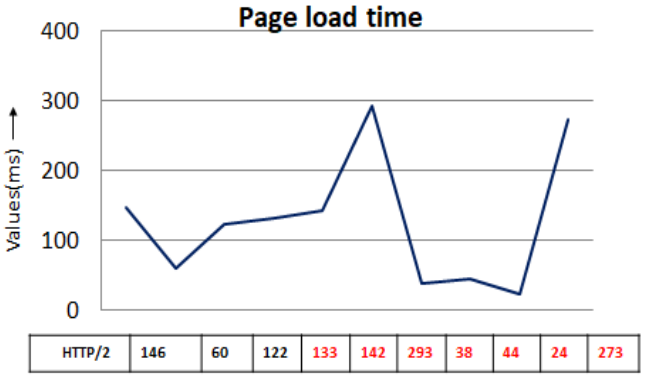

Figure 8(d): Page load time in HTTP/2 protocol

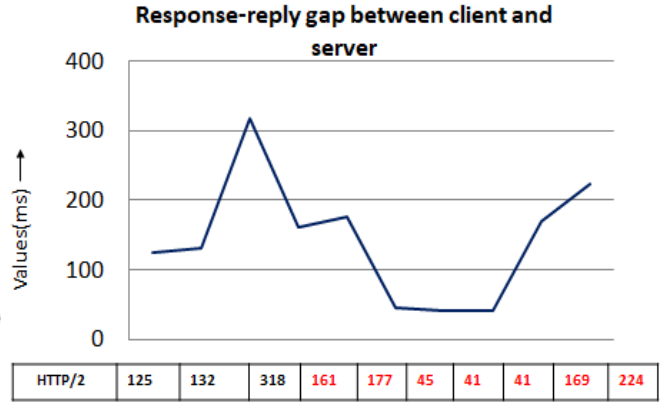

Figure 8(e): Response-reply gap between endpoints in HTTP/2 protocol

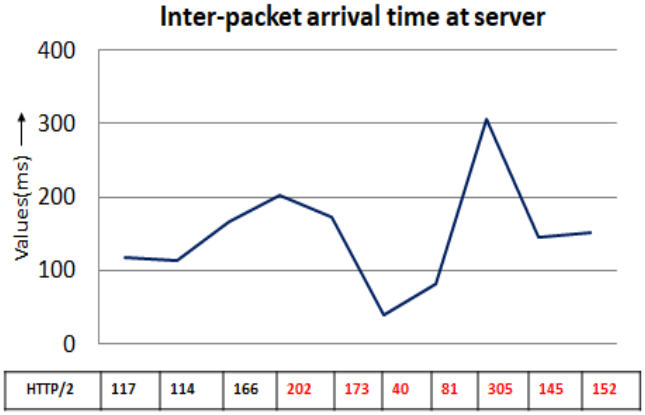

Figure 8(f): Inter-packet arrival time at server in HTTP/2 protocol

5.2 The threshold value set for attack detection

With the obtained result of the server parameter, the threshold is set for parameters- connect time to the webpage and response latency from the server. The values of the threshold to detect the attack are indicated in Table 1. When a random experiment is conducted, and the value of the parameter defined is greater than the set threshold then there is a detection of slow-rate attack in the network. The detection and termination of malicious clients from server HTTP/2 are as shown in Figure 9.

Table 1: Threshold Value for HTTP/2 protocol

	HTTP/2protocol (in ms)
server connection time to a webpage	850
latency for server response	0.1

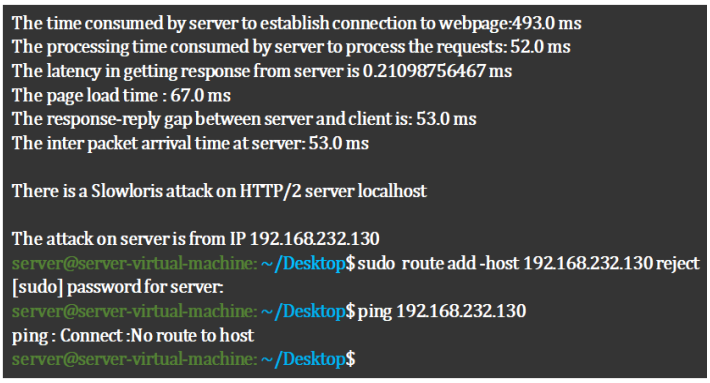

Figure 9: Slow-rate DoS attack detection and termination in HTTP/2 protocol

VI. CONCLUSION

With the number of users increasing the network should be capable of handling many requests with available network resources. This exhaustive usage of the network resources has led to Denial of Service attacks. The binary message framing of HTTP/2 allows a large set of messages to be processed within a short duration of time compared to the HTTP/1.1 protocol. The performance of the HTTP/2 protocol is better than HTTP/1.1 concerning time to load web pages with large resource requests by reducing page load time for clients. This experimental work has analyzed HTTP/2 server parameters which prove the protocol is faster than its lower versions. With the obtained results the attacker was identified and blocked from the network.

VII. FUTURE SCOPE OF THE WORK

The future scope of the paper is to develop a system that can efficiently detect stealthy slow rate-Denial of Service attacks over legitimate network traffic to decrease the request-response time between servers and clients.

REFERENCES

[1] M. Belshe R. Peon, and M. Thomson, "Hypertext Transfer Protocol Version 2 (HTTP/2) " Internet Engineering Task Force (IETF), May 2015

[2] Nick Naziridis, “An introduction to HTTP/2”, SSL.com, https://www.ssl.com/article/an-introduction-to-http2/

[3] Erwin Adi, Zubair Baig, Chiou Peng Lam, and Philip Hingston, "Low-Rate Denial-of-Service Attacks against HTTP/2 Services", The 5th International Conference on IT Convergence and Security At Kuala Lumpur, Malaysia, August 2015

[4] Yihang Zhang, and Yijie Shi, "A Slow Rate Denial-of-Service Attack Against HTTP2", IEEE 4th International Conference on Computer and Communications, 2018

[5] Nikhil Tripathi, and Neminath Hubballi, "Slow Rate Denial of Service Attacks Against HTTP/2 and Detection", Accepted Manuscript, Computers & Security,2017

[6] Amit Praseed, and P. Santhi Thilagam, "Multiplexed Asymmetric Attacks: Next-Generation DDoS on HTTP/2 Servers", IEEE Transactions on Information Forensics and Security,2019

[7] Erwin Adi, Zubair A. Baig, Philip Hingston, and Chiou-Peng Lam, "Distributed denial-of-service attacks against HTTP/2 Services", Springer Science Business Media New York, 11th Jan 2016

[8] Mark Nottingham (IETF HTTPBis Chair), GoodReads, O'Reilly, https://hpbn.co/transport-layer-security-tls/

[9] Hacker Intelligence Initiative, "HTTP/2: In-depth analysis of the top four flaws of the next generation web protocol", Imperva,2016

[10] Catherine(Kate), "HTTP/2 & QUIC-Teaching Good Protocols To Do Bad Things", Cisco publication,2011

[11] Erwin Adi, "Denial-of-service attack modeling and detection for HTTP/2 services", Edith Cowan University- Research Online,2017

[12] Erwin Adi, and Zubair Baig "Stealthy Denial of Service (DoS) Attack Modelling and Detection for HTTP/2 Services", Journal of Network and Computer Applications,2017

[13] David Beckett, and Sakir Sezer. " HTTP/2 Tsunami: Investigating HTTP/2 Proxy Amplification DDoS Attacks", Seventh International Conference on Emerging Security Technologies (EST), 2017

[14] L. M. Bach, B. Mihaljevic, and A. Radovan, "Exploring HTTP/2 advantages and performance analysis using Java 9 ", 40th International Convention on Information and Communication Technology Electronics and Microelectronics, July 2017

[15]Nagy Ramadan Darwish, and Ihab Mohamed Abdelwahab, "Impact of Implementing HTTP/2 in Web Services", International Journal of Computer Applications, Volume 147 – No. 9, August 2016

[16] Mukhtar Abdirahman Abdillahi, Ualikhan Dossetov, and Ali Saqib, " Performance evaluation of HTTP/2 in Modern Web and Mobile Devices", American Journal of Engineering Research (AJER), Volume-6, Issue-4, pp-40-45, 2017