Dos Attack against HTTP protocol using incomplete request and its detection

A Capstone Project Report

Submitted to the Faculty

of the

Bennett University

By

MAYANK AGARWAL

E16CSE145

In Partial Fulfillment of the Requirements

for the Degree of

Bachelor of Technology

A close up of a sign

Description automatically generated

Major Department: Computer Science Engineering

October 2019

Greater Noida-201310, Uttar Pradesh, India

# CERTIFICATE

I hereby certify that the work which is being presented in the B.Tech. Capstone Project Report entitled **“Dos Attack against HTTP protocol using incomplete request and its detection”,** in partial fulfillment of the requirements for the award of the **Bachelor of Technology in Computer Science & Engineering** and submitted to the Department of Computer Science & Engineering of Bennett University Greater Noida UP is an authentic record of my own work carried out during a period from July 2019 to November 2019.

The matter presented in this thesis has not been submitted by me for the award of any other degree elsewhere.

Signature of Candidate

MAYANK AGARWAL

E16CSE145

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

Head

Computer Science Engineering Department

Bennett University Greater Noida UP

ABSTRACT

HTTP defines how the messages are transmitted and formatted to the servers over the internet. It also helps to make TCP connection smoother and an efficient one. But still this protocol is not safe to use and there are vulnerabilities which are not discovered yet. This report focuses on basics of HTTP and some of the threat vectors which could gave one a knowledge how attacks are done in real life and briefs about some of the defense mechanism to be followed to protect our important data by the attacks. In this project I checked different threat vectors of HTTP/2 by launching denial of service attacks which is done by sending incomplete requests of TCP connection on Apache servers in controlled environment. For this I created my own server containing some random data and crafted different and unique HTTP packets which were then send to the server containing some data, which resulted in server to wait for request to get completed so to create connection with the sender and during its waiting time all the service to all other legitimate users were denied and none of the legitimate user wasn’t able to get connection from the server. I also made a prevention technique against these attacks so one can protect themselves from such kind of attacks. For this I made a python script which check each packet that is received by the server and checks if it is incomplete or complete request and drops the packet if it contains incomplete request. This script runs in background. **ACKNOWLEDGEMENTS**

This project has consumed lot of dedication and huge amount of work with proper research. Still, implementation would not have been possible if I did not have a support of many individuals. Therefore, I would like to express my sincere gratitude to all the individual who helped me in doing this project.

I would like to express my special thanks of gratitude to my mentor Dr. Mayank Swarnkar who gave me the opportunity to do this wonderful project with a belief on me and helped me in doing a lot of research and learn about so many new things.

I am extremely great full to my Head of the Department Dr. Deepak Garg who gave me opportunity to do any project and friends who helped me a lot in finalizing this project within the limited time frame.

Mayank Agarwal

E16CSE145

(7th Semester, B.Tech. CSE)

DEDICATION

I would like to dedicate my work for my project to my family members and to my friends. I would also like to express my feeling of gratitude to my loving parents because they are the one whose word always encourage me and motivate me to always do better. I would also like to dedicate gratitude to my faculty who is always ready to help at any time. I will lie to give special thanks to, especially Dr Mayank Swarnkar for helping me to develop and improve my technology skills.

**TABLE OF CONTENTS**

[ABSTRACT 1](#_Toc23530110)

[DEDICATION 3](#_Toc23530111)

[LIST OF TABLES 6](#_Toc23530112)

[LIST OF FIGURES 7](#_Toc23530113)

[LIST OF ABBREVIATIONS 8](#_Toc23530114)

[1. INTRODUCTION 9](#_Toc23530115)

[2. Background Research 10](#_Toc23530116)

[2.1. PROPOSED SYSTEM 12](#_Toc23530117)

[2.2. Goals and Objectives 13](#_Toc23530118)

[3. Project Planning 13](#_Toc23530119)

[3.1. Project Setup 14](#_Toc23530120)

[3.2. Stakeholders 15](#_Toc23530121)

[3.3. Project Resources 15](#_Toc23530122)

[3.4. Assumptions 15](#_Toc23530123)

[4. Project Tracking 16](#_Toc23530124)

[4.1. Tracking 16](#_Toc23530125)

[4.2. Communication Plan 16](#_Toc23530126)

[4.3. Deliverables 17](#_Toc23530127)

[5. SYSTEM ANALYSIS AND DESIGN 18](#_Toc23530128)

[5.1. Overall Description 18](#_Toc23530129)

[5.2. Users and Roles 20](#_Toc23530130)

[5.3. Design diagrams/ UML diagrams/ Flow Charts/ E-R diagrams 21](#_Toc23530131)

[6. User Interface 24](#_Toc23530132)

[6.1. UI Description 24](#_Toc23530133)

[6.2. UI Mockup 25](#_Toc23530134)

[7. Algorithms/Pseudo Code 25](#_Toc23530135)

[8. Project Closure 26](#_Toc23530136)

[8.1. Goals / Vision 26](#_Toc23530137)

[8.2. Delivered Solution 26](#_Toc23530138)

[8.3. Remaining Work 27](#_Toc23530139)

[REFERENCES 28](#_Toc23530140)

LIST OF TABLES

Table Page

[Table 1: Goal and Objectives…………………………………………………………………….13](#_Toc20994041)

[Table 2: Decision Description 14](#_Toc20994042)

[Table 3: Stakeholders 15](#_Toc20994043)

[Table 4: Resources Required 15](#_Toc20994044)

[Table 5: Assumptions 15](#_Toc20994045)

[Table 6: Sample 6 16](#_Toc20994046)

[Table 7: Regularly Scheduled Meetings 16](#_Toc20994047)

[Table 8: Information To Be Shared Within Our Group 16](#_Toc20994048)

[Table 9: Information To Be Provided To Other Groups 17](#_Toc20994049)

[Table 10: Information Needed From Other Groups 17](#_Toc20994050)

[Table 11: Deliverables 17](#_Toc20994051)

[Table 12: Test on ther Server](#_Toc20994052) 19

Table 13 : Users………………………………………………………………………………….20

LIST OF FIGURES

Figure Page

[Figure 1: Testbed Architecture](#_Toc20994053) 14

[Figure 2: Complete vs Incomplete Request……………………………………………………1](#_Toc20994054)9

Figure 3:Usage and Statistical details………………………………………………………….20

gure 4: HTTP/2 Architecture…………………………………………………………………..21

**Figure 5: Complete Request Working**………………………………………………………………………………………………..22

[Figure 6: Incomplete Request…………………………………………………………………..](#_Toc20994057)23

Figure 7:Attack Vectors……… ……………………………………………………………..... 23

Figure 8: Kali Linux Terminal……….……………………………………………………...…25

LIST OF ABBREVIATIONS

HTTP- Hyper Text Transfer Protocol is used to determine how data is transmitted and formatted over the internet and also determine what action does Web browser respond in response

TCP- Transmission Control Protocol is used to establish and maintain smooth network connection between end points so data can be exchanged easily.

DOS- Denial of Service in which service to user are denied because of consumption of all the resources available .

SYN- Short for Synchronize is used to establish connection between two user and it is sent to another computer requesting for the same.

ACK- Acknowledgement is used as confirmation that the packet has been successfully received at the other end.

1. INTRODUCTION

Network Security is a major issue nowadays. As we know that data stored digitally is not safe at all and lot of data is being compromised by the attackers. In today’s world attackers launch lot of attacks on servers or website to make them down and servers/websites wait for complete request and during its waiting time, service to all other legitimate users trying to get connection will be denied. My project deals with one of the methods to break the law and how to safeguard from it. HTTP [1] defines message protocol to be followed for its transmission to servers, but this protocol is not safe and there are some major threat vectors which needs to be understand. This can be done and implemented and checked using different type of servers, if attack is possible or not and try to find out whether the servers are vulnerable to these attacks or not. Also, there is not much literature work about the HTTP/2 [2] so the major threats of HTTP/2 need to be understand in order to develop a prevention or mitigation technique against these attacks so one can protect their important data from such kind of attacks.

**PROBLEM STATEMENT**

Everyone who uses internet and HTTP plays important role in accessing it. Most of the data of organizations/individuals are stored digitally on websites and servers and we know that day by day malicious attacks [4] on different websites and servers are increasing and lot of data is being compromised. This could be due to any of the vulnerability that exists in the websites and one them could be HTTP also. Many organizations like Netflix, Hotstar etc. provide services like of online streaming of videos and if any of these organizations’ servers are down for even few minutes, they experience some loss in their business. So, my project will help to reduce the attacks on servers done through done through the crafted HTTP request and protect server from going down and keeping all the confidential data secure. We know that data is store on the server so if server is not secure then there will be huge loss as attacker can do lot of attack regularly on the server and can also demands money from the organizations and can also misuse the data and also if attacker make the server down so services to legitimate users will be denied so my project will detect this kind of attack done through specially crafted packet on server and automatically detect them.

1. Background Research

The basic thing to do anything is to do research. There are lot of research being done on this project by me and I found that there is not much literature about HTTP/2 but still some research is there which has very interesting results and facts about HTTP. I found that many of people have tried this attack in different way and most of them are successful in some way and also the research on denial of service attack through HTTP/2 is still continue going on and many people are trying to find more and also people are discovering some new vulnerabilities regularly on HTTP. The Slow Loris attack [6] is best slow Denial of service attacks which has been done successfully It is also known as SYN flood attack as it works on the application layer. It sends incomplete HTTP requests to the target to perform denial of service attack because this attack makes the web server to wait for the competition of request, but the attacker never sends the complete HTTP requests to server which results in denial of service to legitimate users. In this attack after connection is timed out or closed, the attacker sends some other parameters to trigger the attack again. The slow read HTTP attack [7] deals with the different types of attacks and their taxonomy. In this paper, the attack on server is done by sending incomplete HTTP request and incomplete attack does not contain /r/n at the end line which is necessary for a complete request. The server waits for complete request but attack never sent the complete request instead attacker send incomplete request again to restart the timer before connection is timed out. This paper [8] proofs some of the validations which have done on this area. In this they have only used HTTP/2 request architecture over TLS and has also worked upon HTTP/1 and showed how both request causes harm to server. In this paper it mainly focuses on HTTP/2 as it contains more threat vectors or vulnerabilities as compared to HTTP/1. Also, people and companies are shifting from HTTP/1 to HTTP/2 so it’s better to work on HTTP/2. In this paper they have done 5 attacks on servers containing images as their data and they have used HTTP/2 request architecture in which they have sent incomplete request and never sent the full request making server wait for few seconds and also some of server disconnect immediately. In this paper [9] they have stolen the HTTP encrypted information using TCP windows by sending larger amount of TCP packets greater than the initial windows setting parameter of HTTP to server which resulted denial of service to the user. This attack also proves that there is no requirement of physical network for network based we attacks. So they were also able to find vulnerabilities using HTTP. In this paper [10] they have used cloud-based platform for detecting HTTP/2 based denial of service attack. They have used machine learning models based on Information Theoretic Entropy and Random Forest ensemble learning algorithm on cloud environment to detect the attacks done on server. Similarly, there are many more projects which same or similar things have done to do same type of attack in a different way. I liked to work in cyber security field as it allows you to explore lot of new things and learn about them. On reading all these research or literature, it makes me confident and excited to this project so I could help other in protecting their data from attackers and I am confident that I will also find new attacks using new methods as mentioned above and a new defense mechanism. This motivates me to do something new and I am confident that will also be able to find many new vulnerabilities in my project also the sniffer I will be making is also different as it will work on finding incomplete request as it will capture the packets and try to find flags used which results in denial of service attack to the legitimate users.

* 1. PROPOSED SYSTEM

We know that day by day malicious attacks on different websites and servers are increasing and lot of data is being compromised. So, my project will help to reduce the attacks on servers done through done through the crafted HTTP request and protect server from going down and keeping all the confidential data secure and all services are available to all legitimate user all the time. We know that data is store on the server so if server is not secure then there will be huge loss as attacker can misuse the data and also if attacker make the server down so services to legitimate users will be denied and we also know that if server belongs to any organization which provides services to its user and earn money from it can result in loss of company even the server is down for few minutes or more so my project will detect this kind of slow attack which includes incomplete request done through specially crafted packet on server and help in dropping those packets and prevent it from reaching it to the servers. So, it will provide more security to the users and all services will be available to user at all the time so one has less chances of being attacked by attacker and their data will be stored safely on the ser

* 1. Goals and Objectives

Table : Goal and Objectives

|  |  |
| --- | --- |
| **#** | **Goal or Objective** |
| 1 | Convert method of this project into a product in form of android application. |
| 2 | Improve the security to very high level so it can’t be broken easily |
| 3 | Discover newer vulnerabilities and threat vector of different severs |
| 4 | Build user interface for non-technical people for ease of use for them |
| 5 | Implement patches in the sever |

1. Project Planning

**3.1 Cyber Security Project Lifecycle**

There are some important steps which are involved in the lifecycle of cyber security projects which are as follows: -

1. **Initiation/ Identify Key Assets**- It is the main phase as in this phase we decide what type of data needs to be protected and which information is more important so it can be protected.
2. **Detect Threat**- In this phase we try to find the vulnerabilities and try to attack them as much as possible to find the key vulnerabilities and parameters which are responsible for it
3. **Patch the Attack Parameters-** In this phase of lifecycle, threats are monitored based on damage it can cause to data and crucial information stored digitally and solutions are provided in order to patch them.
4. **Testing-** After patching the threats its time to test whether the patch provided is successful in order to stop the attack and protect data.
5. **Deployment-** After successful testing solutions are released for everyone so they can patch their system.
   1. Project Setup

Table : Decision Description

|  |  |
| --- | --- |
| **#** | **Decision Description** |
| 1 | Windows 10, Kali Linux, Apache server |
| 2 | Coding for project is in Python, Scapy |
| 3 | Release to open source |
| 4 | A server setup hosting a website |

**A picture containing screenshot

Description automatically generated**

**Figure 1: Testbed Architecture**

In this project to check the threat vectors of HTTP we need one computer as a web server which is running on Windows 10 operating system and other computer running on Kali Linux operating system configured with apache server to handle HTTP request. The attack was launched from Kali Linux Terminal and the Apache server version is 2.4.23. We also use Android phone for testing our server connection after attack.

* 1. Stakeholders

Table : Stakeholders

|  |  |
| --- | --- |
| **Stakeholder** | **Role** |
| Dr. Mayank Swarnkar | Mentor |
| Dr. Mayank Swarnkar | Instructor |
| Rahul Pundir | Support People |
| Jasraj Singh | Support People |

* 1. Project Resources

Table : Resources Required

|  |  |  |
| --- | --- | --- |
| **Resource** | **Resource Description** | **Quantity** |
| Apache Server | Apache server created by me. | 4 |
| Capstone Team | No team member | 0 |
| Dr Mayank Swarnkar | The mentor who helped with the project throughout to provide with technical assistance. | 1 |
| Windows  Workstation | The Windows 10 workstation for hosting web server and capturing the packets. | 1 |
| Kali Linux Workstation | The Kali workstation for sending manually crafted packets | 1 |
| Android Phone | An Android phone to be used as test hardware for the mobile version of the software. | 1 |

* 1. Assumptions

Table : Assumptions

|  |  |
| --- | --- |
| **#** | **Assumption** |
| A1 | The face to face meeting with mentor regularly in week. |
| A2 | New threat vector will be available after 1 month of work. |
| A3 | In two months, new vulnerabilities will be discovered in different types of servers. |
| A4 | All the attacks would be covered within 2 months with good result. |
| A5 | Defense mechanism would be completed within a month after attacks have been discovered. |
| A6 | The captured attack packets (test data) will be enough to create an accurate prediction of user actions. |
| A7 | The Proper user design interface would be created within the time frame |

1. Project Tracking
   1. Tracking

Table : Sample 6

|  |  |  |
| --- | --- | --- |
| **Information** | **Description** | **Link** |
| Code Storage | Project code stored on Github. | <https://github.com/smartmayank98/Mayank-project> |
| Project Documents and Assignments | Project stored on github | <https://github.com/smartmayank98/Mayank-project> |

* 1. Communication Plan

Table : Regularly Scheduled Meetings

|  |  |  |
| --- | --- | --- |
| Meeting Type | Frequency/Schedule | Who Attends |
| Meeting With mentor | Weekly | Mentor |
| Short Meeting | Weekly in class/labs | Mentor |
| Sprint Planning Meeting | Twice in month | Mentor |
| Sprint Review Meeting | End of each sprint (Once in month) | Mentor |

Table : Information To Be Shared Within Our Group

|  |  |  |  |
| --- | --- | --- | --- |
| Who? | What Information? | When? | How? |
| Mentor | Task assignments & Technique Adopted | Weekly | Meetings |

Table : Information To Be Provided To Other Groups

|  |  |  |  |
| --- | --- | --- | --- |
| Who? | What Information? | When? | How? |
| Mentor | Final deliverables | At completion of project | Project specification doc., code, Power Point presentation |
| Mentor | Weekly report | Weekly/Daily | Personal Meeting |

Table : Information Needed from Other Groups

|  |  |  |  |
| --- | --- | --- | --- |
| Who? | What Information? | When? | How? |
| Faculty/Mentor | Techniques and Requirement Changes | Start of each sprint | Personal Meeting |

* 1. Deliverables

Table : Deliverables

|  |  |
| --- | --- |
| **#** | **Deliverable** |
| 1 | Study results |
| 2 | Code |
| 3 | Test and test results |
| 5 | Final report (final PowerPoint presentation, 3-minute video, and final sprint) |
| 6 | Poster |
| 7 | Blog |

1. SYSTEM ANALYSIS AND DESIGN

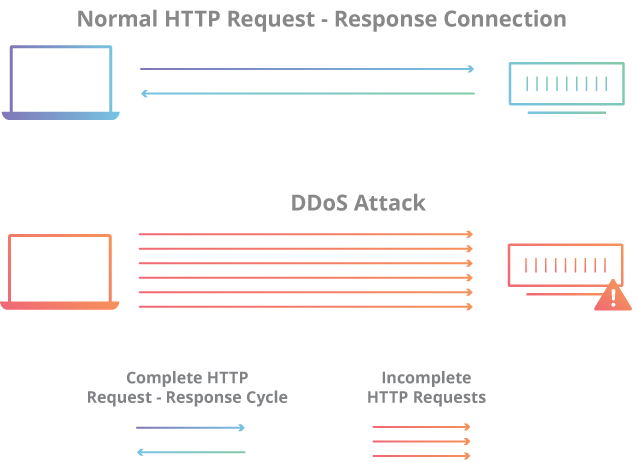
This section describes in detail about the design part of the system.

* 1. Overall Description

This project is an attempt to apply cyber security concepts and techniques to secure data stored on the server from the attacker. As mentioned above in the report we know that attacker send incomplete request to the server which results in denial of service by server.

This attack is mainly an example of slow-rate attacks [3] means attack parameters are slowly sent again and again to server. The attack works by tying up or continuously consuming server resources by sending legitimate incomplete HTTP requests to and making server to wait for complete request and reserve the resources available for open connection waiting for their completion.

The complete HTTP request sent to server always ends with a line as \r \ n, which is always followed by an empty line which signifies the end of the request and it is necessary to complete the request. The attacker removes these lines and sends incomplete request and the server must wait for the end of the request so the half open connection can be completed, and the server waits for a specific timeout before it closes connection. This timeout is restarted again when the attacker sends the next packet containing incomplete request, so the server again wait for same time for request to get completed. Thus, in this Slow HTTP attack, the attacker never sends the final line/ complete request to the server in order to keep the connection half open as long as possible which results in denial of service to another user.

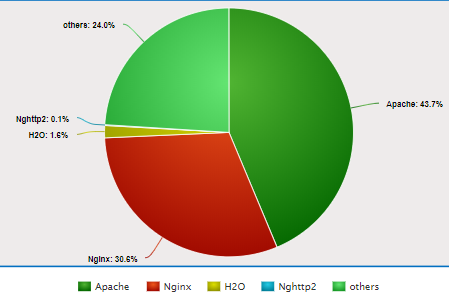


**Figure 2: Complete vs Incomplete Request**

**Table 12: Test on other server**

|  |  |  |
| --- | --- | --- |
| SERVER | MINIMUM WAITING TIME(IN SECONDS) | MAXIMUM WAITING TIME(IN SECONDS) |
| Apache | 300 | Indefinite |
| Nginx | 20 | Indefinite |
| H20 | 10 | Indefinite |
| Nghttp2 | 10 | 600 |

The attack was tested on 3 other different server and the following result was found that the maximum waiting time for Apache, Nginx, H20 are indefinite for all which is only when client does not send any data to server after sending post request



**Figure 3:** **Usage and Statistical details**

These are overall details of various server used globally by different organizations so a single vulnerability in any server can cost huge loss to organizations and lot of very crucial data can be compromised by attackers.

* 1. Users and Roles

Table 13: Users

|  |  |
| --- | --- |
| **User** | **Description** |
| Developer | I am only responsible for managing everything which includes finding new vulnerability, working with the servers, testing on them to find different vulnerable parameters and developing mechanism against them to prevent from such attacks |
| Instructor | Dr. Mayank Swarnkar, professor at Bennett University who is responsible for all guidance and providing some study resources for helping to complete my project. |
| User / Organization | An end user of this vulnerability and prevention mechanism who will get to know more about these vulnerabilities and can protect their data and gets benefits from data analysis of different packets from different server. |

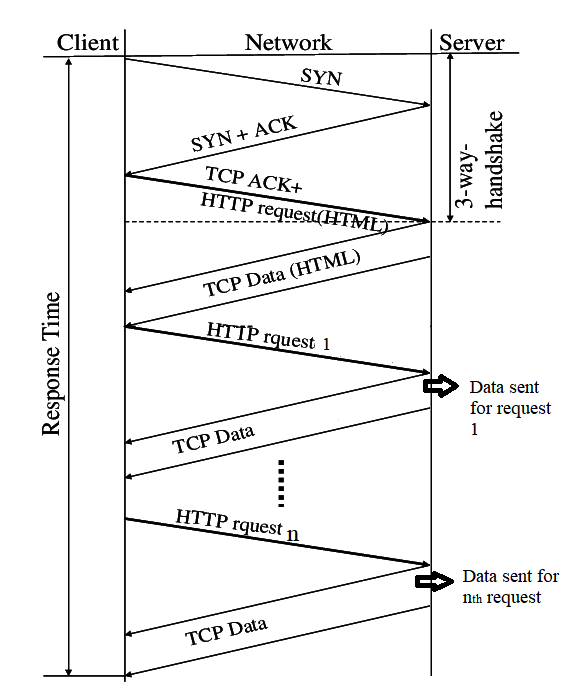
* 1. Design diagrams/ UML diagrams/ Flow Charts/ E-R diagrams

A screenshot of a cell phone

Description automatically generated

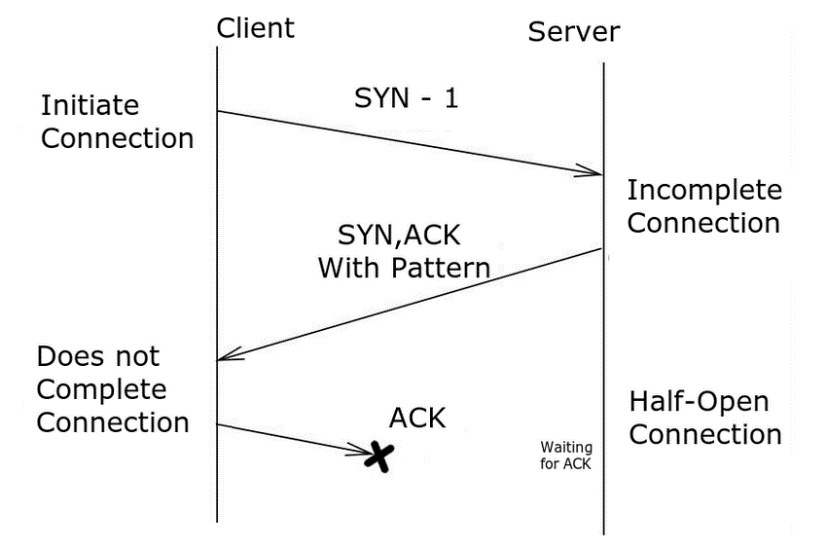
**Figure 4: HTTP/2 Architecture**

This diagram depicts the complete working of HTTP/2 protocol. HTTP/2 messages are divided into different frames having different features like Connection Preface establish the initial settings for HTTP/2 connection, HEADERS and CONTINUATION Frames carry a header block and to transmit remaining parts of header block, DATA Frame carry message body sent by the endpoints, WINDOW UPDATE Frame indicate the number of bytes that the sender accept by its peer, GOAWAY Frame tear off established connection between end points.



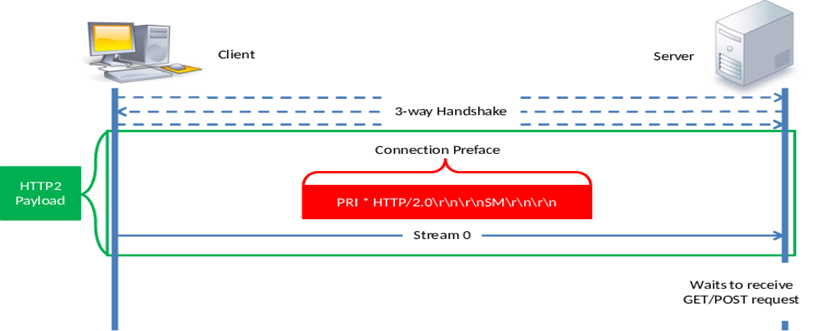
**Figure 5. Complete Request working**

In the complete request, first TCP 3-way handshake is done then the number of complete requests are sent by client to server and for each corresponding request server sent data back to client.



**Figure 6: Incomplete request**

In the incomplete request the attacker never send the complete request and force the server to wait for complete request making the connections half open.



**Figure 7: Attack Vector**

It depicts the attack vector of incomplete request that it should contain /r/n at the end for the request to get completed and if its not there then it is a incomplete request.

1. User Interface
   1. UI Description

As my project is based on cyber security n which I am doing attack on server, so I haven’t created any UI, but my project is very easy to use. The python script must be run from Linux terminal and any normal user can easily do that to check whether their server is vulnerable to my attack or not. Also, one has to run Wireshark to capture packets and run another script which will help to find out IP from incomplete request are coming and one can block them to protect their data store online. In this way one can protect their data stored online.

* 1. UI Mockup

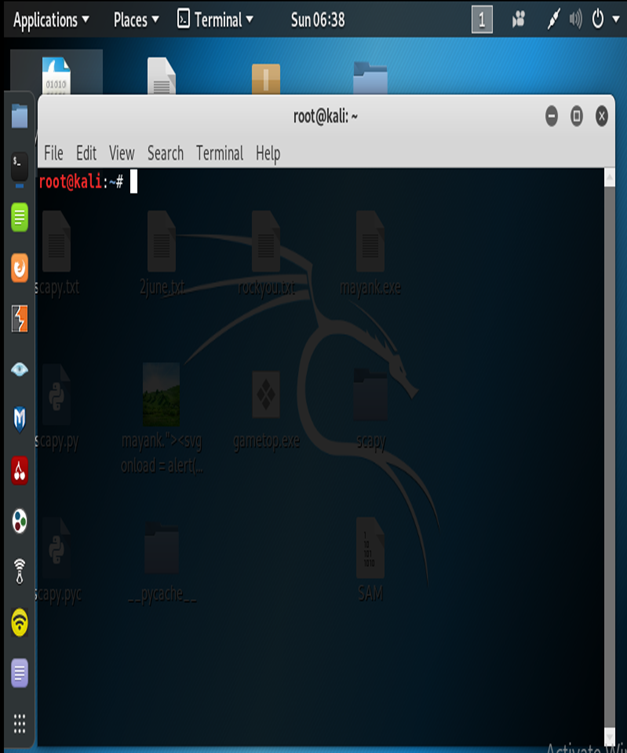


Figure 8: Kali Linux Terminal

Since my project is an attack based so all attack will be done with the help of Kali Linux terminal so there is no such UI created for my project.

1. Algorithms/Pseudo Code

getStr=’GET/HTTP/2\r\nHost:’ + dest +’\r\nAccept-Encoding:gzip,deflate’

This string defines the incomplete request which we used to send to the server. In this there is no /r/n at the end which tell us that request is incomplete.

The algorithm is as follows:

First, we send the SYN packet, then we receive the Get SYNACK packet and then we send ACK package to complete the 3 way handshake and then we send the incomplete request to execute our request.

1. Project Closure

This section elucidates the overall lookup at the project and some of the future works that may enhance the solution.

* 1. Goals / Vision

On basis of Project Initiation Document, I had successfully launched attack oh Apache server and had also made detection scheme that how one can be get protected itself from the attack on the basis of IP address. In the initiation document I mentioned that I will make sniffer but I had written a python script which will detect the Ip so one can manually block them which is similar to the sniffer. As of now I successfully completed one attack and had written script for other attack, but they have accuracy of about 75-80% of successful attack. I have done lot of research which is also a part of my goal or vision.

* 1. Delivered Solution

This attack is mainly an example of slow-rate attacks means attack parameters are slowly sent again and again to server. It was mention that I will do a denial of service attack and that has been delivered successfully. The attack works by tying up or continuously consuming server resources by sending legitimate incomplete HTTP requests to and making server to wait for complete request and reserve the resources available for open connection waiting for their completion. The complete HTTP request sent to server always ends with a line as \r \ n, which is always followed by an empty line which signifies the end of the request and it is necessary to complete the request. The attacker removes these lines and sends incomplete request and the server must wait for the end of the request so the half open connection can be completed. This timeout is restarted again when the attacker sends the next packet containing incomplete request, so the server again wait for same time for request to get completed. Also, a python script is there which has detects IP address which sent incomplete request has been successfully delivered by me.

* 1. Remaining Work

Furthermore, to this project I would like to make a very interactive and attractive UI for my project in which one can just simply press a button and everything will be done automatically at the backend and IP are automatically listed blocked so their packets are drop out and never reach to the server.

Also, I would find some more better vulnerabilities and their defense mechanism so I could help many more people and I will do my level best to achieve that in near future. Secondly, I would try to convert my project into product for organizations and also make it to work for android and iOS devices also. This project can also be improved by adding some different attack vectors because it will add other layer of security to my project.

REFERENCES

1. Belshe, M., Thomson, M., & Peon, R. (2015). Hypertext transfer protocol version 2 (http/2)
2. Cambiaso, E., Papaleo, G., Chiola, G., & Aiello, M. (2013). Slow DoS attacks: definition and categorisation. International Journal of Trust Management in Computing and Communications, 1(3-4), 300-319.
3. Douligeris, C., & Mitrokotsa, A. (2004). DDoS attacks and defense mechanisms: classification and state-of-the-art. *Computer Networks*, *44*(5), 643-666.
4. Adi, E., Baig, Z., Lam, C. P., & Hingston, P. (2015, August). Low-rate denial-of-service attacks against http/2 services. In *2015 5th International Conference on IT Convergence and Security (ICITCS)* (pp. 1-5). IEEE.
5. Garcia-Teodoro, P., Diaz-Verdejo, J., Maciá-Fernández, G., & Vázquez, E. (2009). Anomaly-based network intrusion detection: Techniques, systems and challenges. computers & security, 28(1-2), 18-28.
6. Aiello, M., Cambiaso, E., Mongelli, M., & Papaleo, G. (2014, October). An on-line intrusion detection approach to identify low-rate DoS attacks. In *2014 International Carnahan Conference on Security Technology (ICCST)* (pp. 1-6). IEEE.
7. Mantas, G., Stakhanova, N., Gonzalez, H., Jazi, H. H., & Ghorbani, A. A. (2015). Application-layer denial of service attacks: taxonomy and survey. Ints[ernational Journal of Information and Computer Security, 7(2-4), 216-239.
8. Tripathi, N., & Hubballi, N. (2018). Slow rate denial of service attacks against HTTP/2 and detection. Computers & security, 72, 255-272
9. Vanhoef, M., & Van Goethem, T. (2016). HEIST: HTTP Encrypted Information can be Stolen through TCP-windows. Black Hat USA, 2016, 1.
10. Idhammad, M., Afdel, K., & Belouch, M. (2018). Detection system of HTTP DDoS attacks in a cloud environment based on information theoretic entropy and random forest. Security and Communication Networks, 2018