Homework - 2

Kaustav Vats - 2016048

10-Oct-2018

Part I

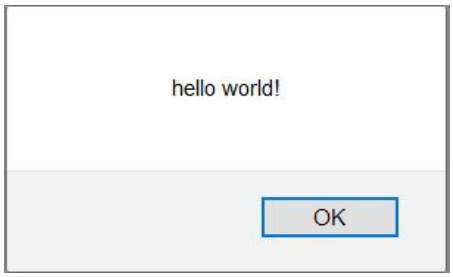
1. In IIITD, each user is assigned a static IP. Users are verified and given access based on the mac address of their device. To use Internet Service provided by IIIT, each machine need to be registered first. IIITD uses 802.1x Security authentication for wireless network. 802.1x is used by access point to implement WPA. Users are authenticated using WPA. For wireless network switches uses 802.1x to do port based authentication. Each time when user connects through wired medium, its assign a port for communication, after disconnecting port state is changed to unauthorized state. 802.1x basically pass the authentication information between client and the authentication server. 802.1x allows multiple authentication techniques. It uses EAP. Network access is only given after user is authenticated. Each user is given a time duration, after which it check net status if not active, disconnect user.

2. The communications that are happening in the IIIT network, cannot be sniffed by any outside user, if he/she doesn't have access to IIITD network. Whereas outsider can sniff packets that are leaving from IIITD network to some other network. If an outsider has access to IIITD network only then he/she can sniff packets present in the college Network. To sniff packets from an organization network, a user must first be connected to the network. Which require user to be authenticated by the authentication server of the network.

3. Internet usage session is basically giving a limited time access of the network. A session consist of some duration. On expiring of that duration a new session is created only if user is active, else close the connection. To access Network again user need to get authenticated by the authentication server and only after this the service will be provided. If the web application is not secure then the attacker can access information from the user and server using packet sniffing. Packet sniffing attacks are know an man-in-middle attacks. Both parties are not aware, that some user is accessing their conversation or not. Packet sniffing is also used for observing the conversation between user and server, and find vulnerabilities to exploit.

Part 2

1. On writing javascript:alert('Hello World!'); creates an alert box saying Hello World! With a button 'OK'.

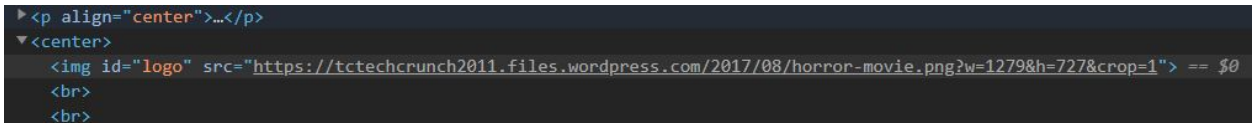


Basically javascripts can be executed from the browser url address bar. This give access for the website and can change the look of the website as well as behaviour of the website from user view.

Browser filter the input entered on the url address bar. I observed that most of the browsers have realised this vulnerability, to protect the user from XSS attacks they have limited the input in address bar by cutting out javascript: when pasting javascript:some_action. Well it's still possible by manually writing the javascript.

Browser identify protocol http://d4rkc0de.iiitd.edu.in:8096/. For any website request it can identify the protocol added with website address and then treat it as http request.

2. Javascript given in the question picked a DOM node with id='logo' and updated src with the new src present in the javascript.



Source code of the page remain same from the server side but can change for a particular user. Eg:- if two user are accessing same website and one user entered this javascript, then only this user will be able to see the change made by the Js.

If the page contains tags like forms then attacker can modify those forms and add malicious content on those forms.
<form id ="contact"> … </form>
Attacker can get element by id attribute in form and add malicious content in the form.

Whenever a user clicks a particular link it will change content of the website for that particular user. This way the source code of the website will remain same from the server side, but will change for the user if he/she clicked some malicious link on the website.
Eg:- `document.ElementById("contact").innerHTML="Malicious Content";`
This script directs browser to change innerHTML by some malicious content.

3. After updating the link, a <IMG> tag with id='logo' was updated with new src attribute value.

```
<div class="board"> Hi <a href="javascript:void(document.getElementById('logo').src='
https://www.wpblog.com/wp-content/uploads/2017/08/wordpress-site-is-hacked.jpg');"> check out this cool link </a> </div>
```

4. Enter this in discussion board then click on 'click me'.

```
<a
href="javascript:void(document.getElementById('logo').src='https://www.wpblog.
com/wp-content/uploads/2017/08/wordpress-site-is-hacked.jpg');">click me
</a>
```

How script works is already explained in above parts.

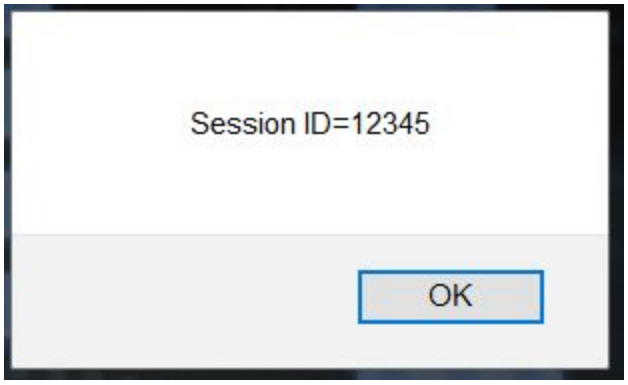
5. On clicking Go to IIITD Homepage open a new page with iiitd website. After clicking the mentioned link it updates Go to IIITD Homepage href attribute to another website link(Google in this case).
Source code of the webpage was not updated. But using "inspect" in Chrome, changes can be seen for a particular user.
Attacker can basically add a link in user in discussion forum, on clicking the link it will update Go to IIITD with some malicious website link and can trick the user.

```
<a
href="javascript:void(document.getElementsByTagName('a')[0].href='http://d4rkc
0de.iiitd.edu.in:8096/');">click me
</a>
```

6. On clicking the mentioned link take user to gmail login page. Attacker can add a link in discussion board which update "Check your email" link with some phishing page(Fake page similar to original page). When user enters his/her username and password, attacker will get the credentials of the user.

```
<a
href="javascript:void(document.getElementsByTagName('a')[1].href='https://ww
w.reddit.com/r/hacking/');">click me
</a>
```

7. Attacker can add a link in discussion board. On clicking that link it will take user to some other page with his/her cookie of the previous page attached with the url. Cookie of the user can be passed inline with the URL. Attacker can have a php server, where it can extract the cookie that is carried by the URL. Attacker can use that cookie to impersonate about specific person.

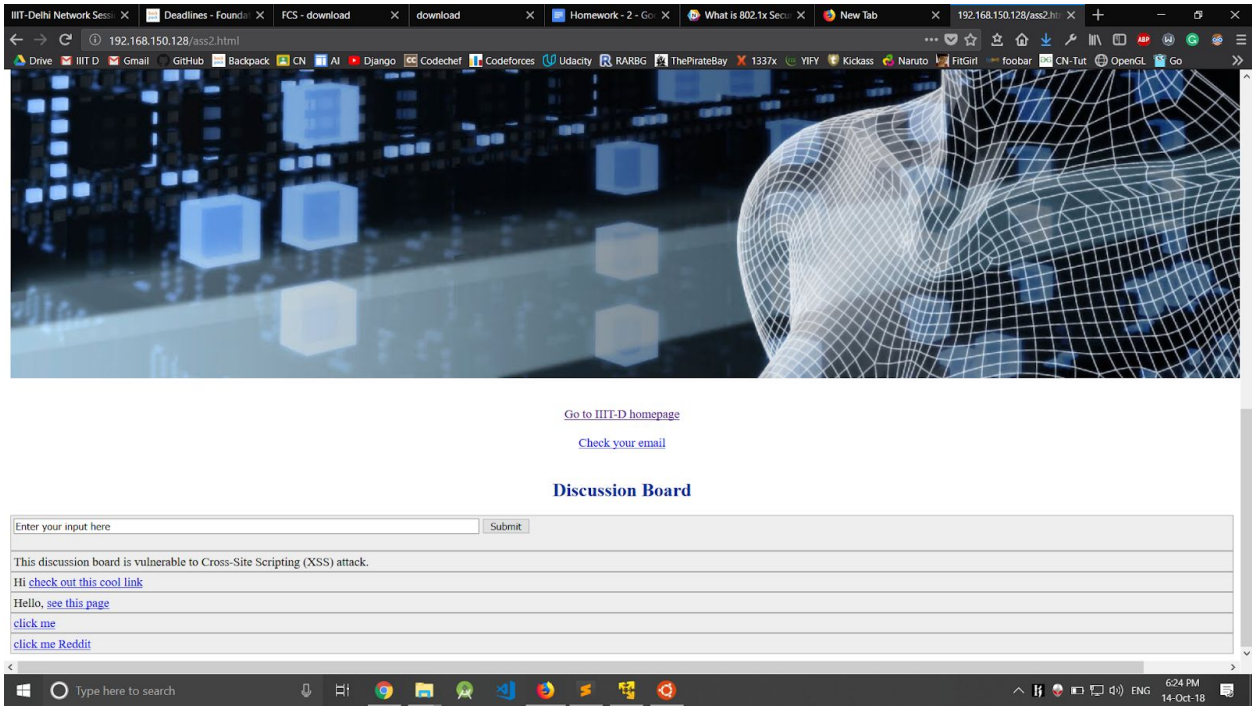


8. XSS can be prevented by adding input validation of text entered by the user. Use Output encoding, encode html data, script before rendering, can also do language specific output encoding etc.
   Update the Write function by:-

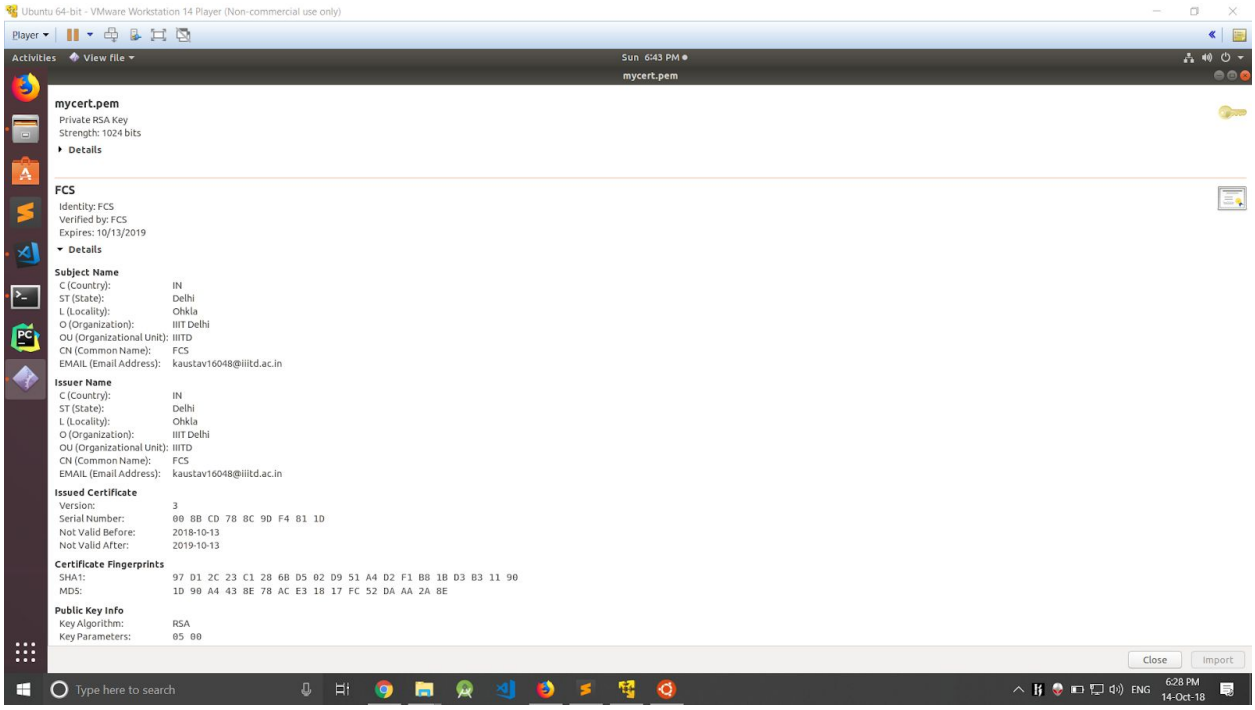
```
var userInput = document.getElementById('userInput').value;
var _body = document.getElementsByTagName('body') [0];
var a = "<Center><div class='board'>";
var b = "</div></Center>";
newDiv = document.createElement("div");
userInput = userInput.replace(/</g, "&lt;").replace(/>/g, "&gt;");
newDiv.innerHTML = a+userInput+b;
_body.appendChild(newDiv);
```

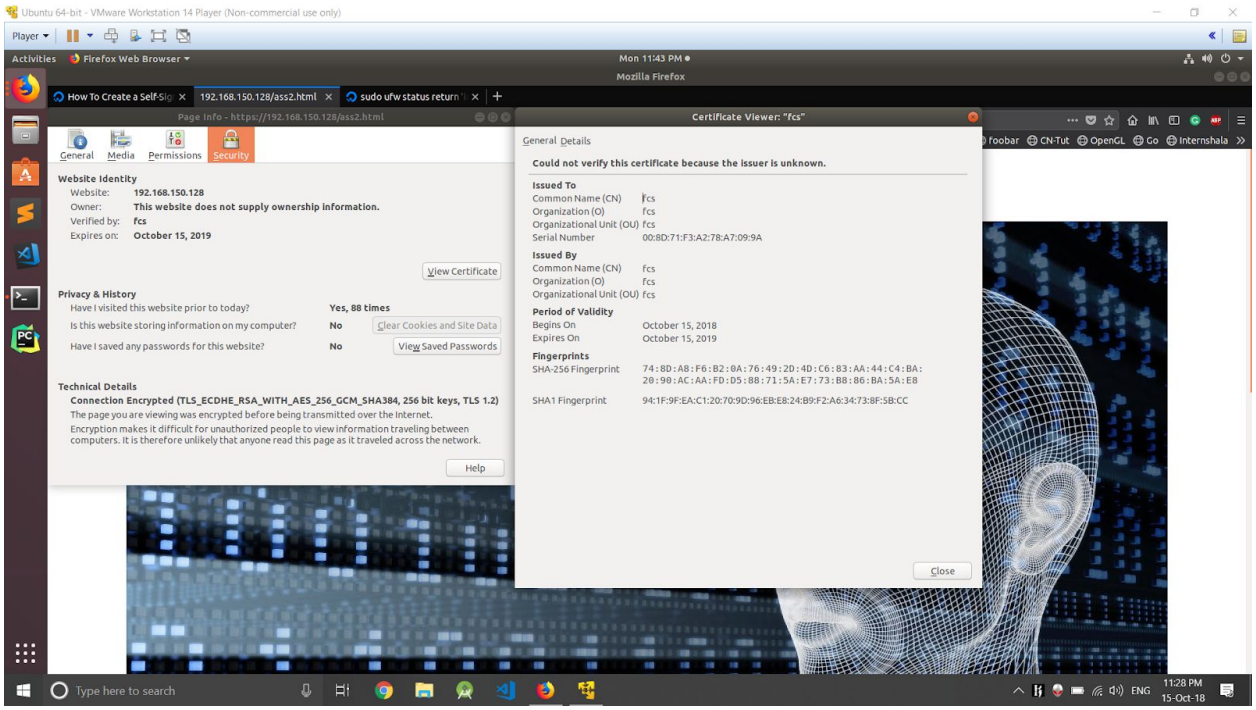
   This bold line will replace angular brackets and display any input in plain text. Whereas a attacker can still add url in plain text and ask reader to visit this link.

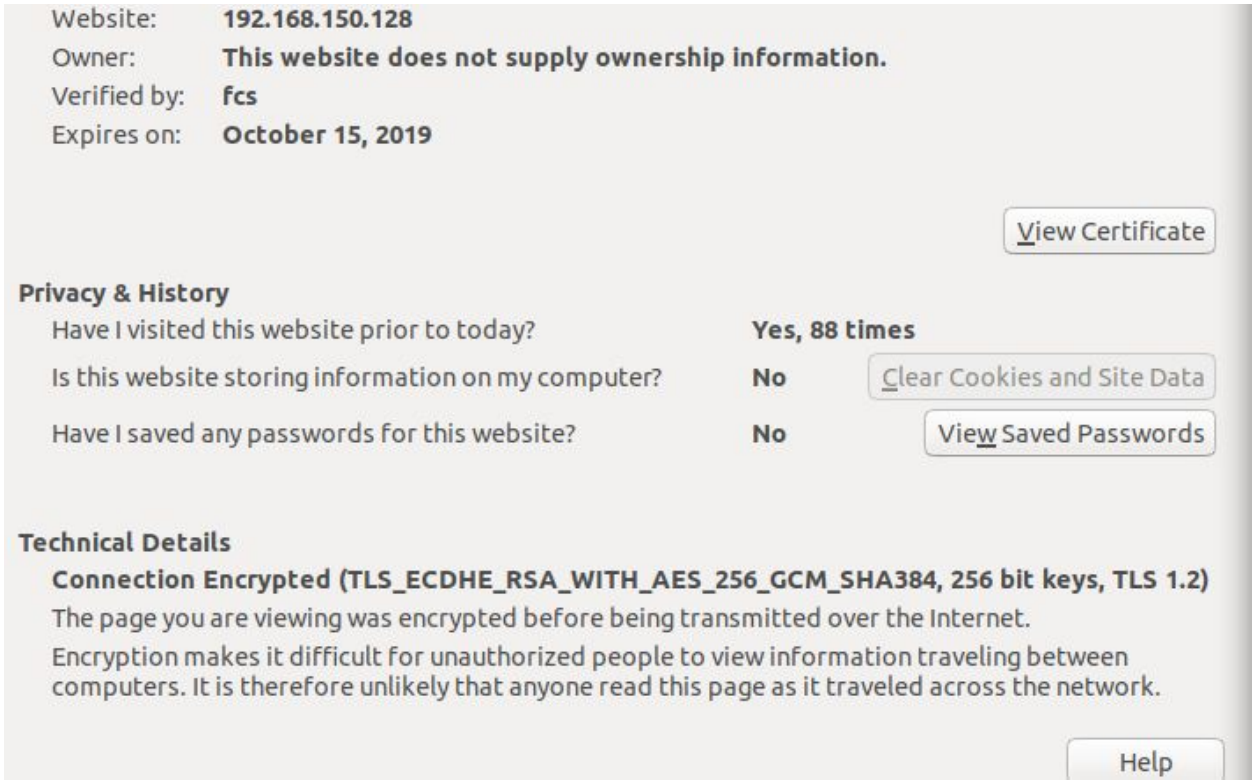
9. Did this on VM.



With php server on the backend for extracting cookie passed using url
Below is the self signed certificate



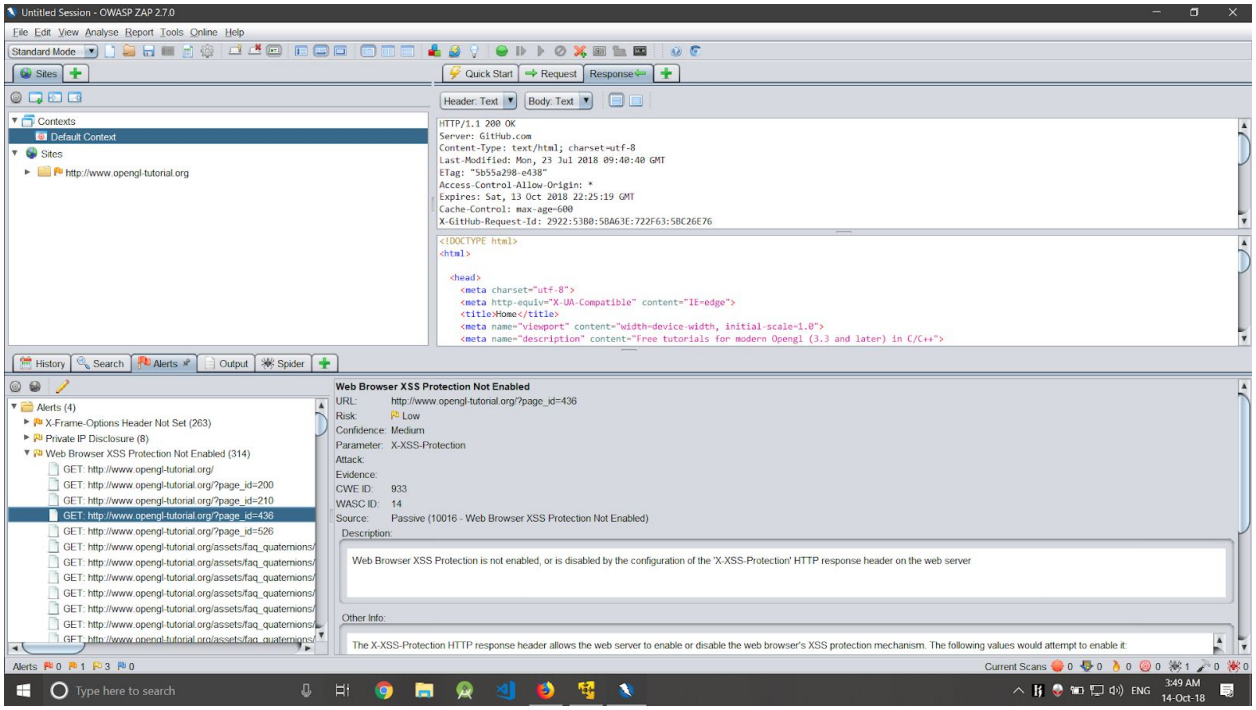




SSL Encryption

10. Https will only provide secure data transmission between user and server. But if website is prone to XSS attacks then Https encryption doesn't really help in providing security. XSS attacks are use by exploiting user input section on the website and many other ways. For all the above mentioned attack, HTTPS won't help in preventing XSS attacks.

Part 3



- These tools are used to find vulnerabilities in a system. They act as a proxy between client and the server. Every request can be seen using these tool. Running for a website give all the vulnerabilities existing on the web application. It can capture html, xml, css of the website. It can capture user credentials if not encrypted. It also check url for any request page parameters(booleans). It also gives Https Info. It can also generate token to test for CSRF protection. It also tries to discover file structure using forced browsing. It also allow to change the http request and the responses.

- Some of the vulnerabilities are that:-
    - Data is not encrypted, sent to server in plain text. Then that data can be used to find vulnerabilities and exploit further pages in the web application.
    - Some web application pass a boolean in a URL to denote authenticity of the user, basically whether the user has logged in or not. This URL can be

observed and further can be used to access restricted pages or pages that require user to be authenticated first.
- Can retrieve information regarding file structure. Which can be exploited and get all the sensitive data including the system information.

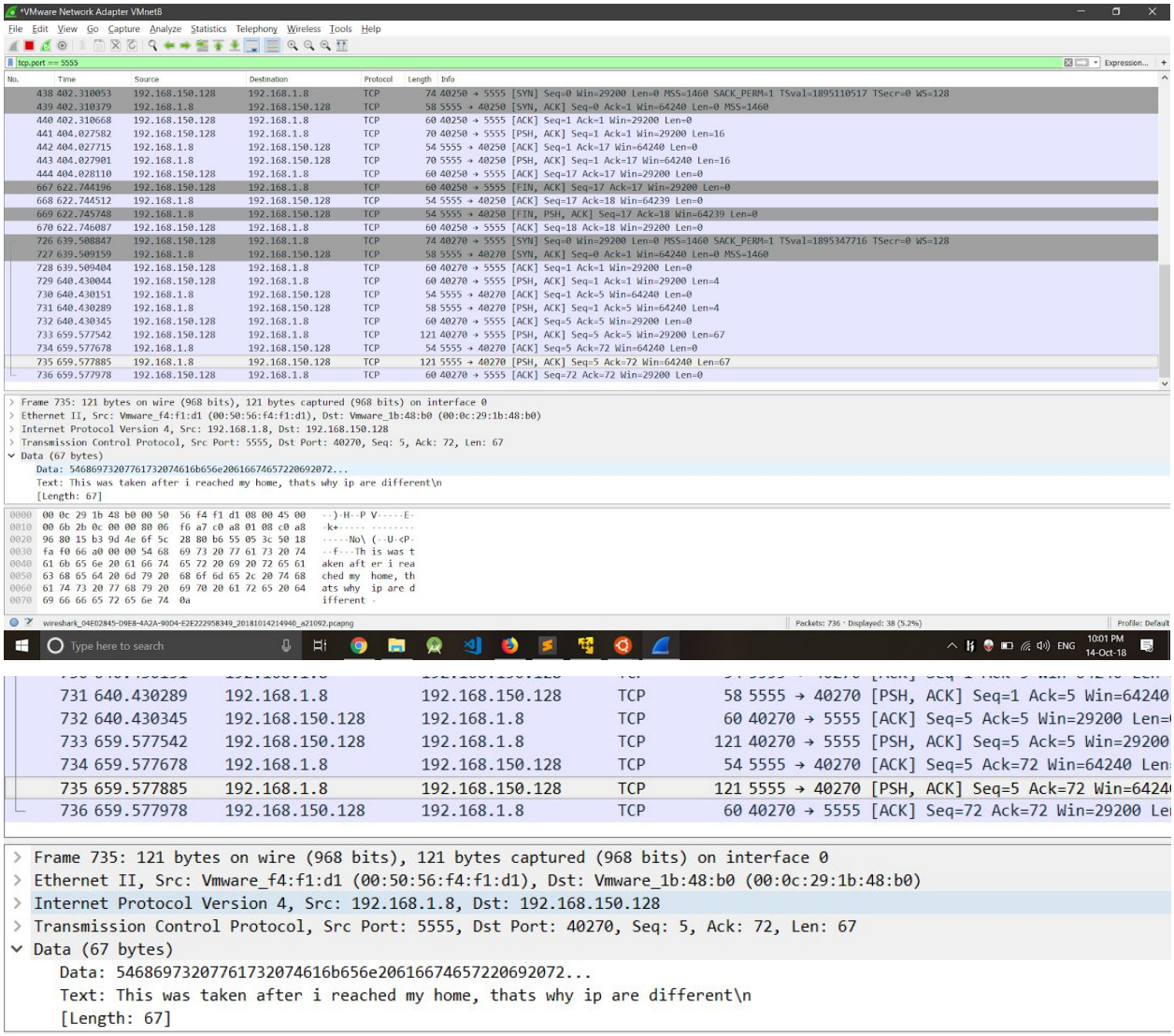
- Well as a developer i would firstly make conversation secure between server and client, by providing encryption and reducing for man in middle attack.
  I will include input validation in my web application to protect from any XSS attack. I will include CSRF token in my web application.
  Add SSL encryption to the website. Include checks for user input. To prevent from DOM attacks and various other XSS attacks. Add K anonymity to the data. Add login check for pages that require user to be authenticated first. Don't pass any parameter with url.

Part 4

1) F function: Given hash ‘H’ created at at time ‘t’ and date ‘d’.
   - Pass H as a input string to generate a new hash value in MessageDiegst(“SHA-256”).
   - Then pass new hash value again in Hash generating function to generate a new hash.
   - Repeat above steps some N no of time(eg N = 10)
   - From the final hash. Extract all numeric digits and select first 4 digits as OTP.

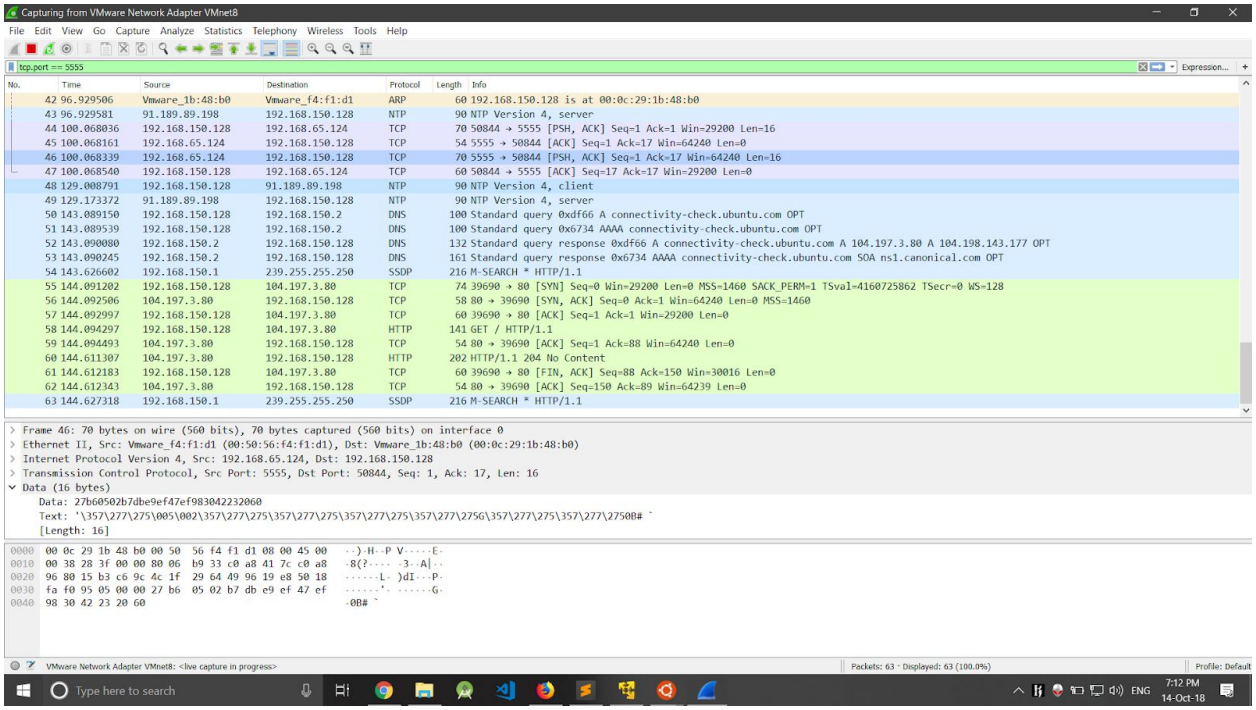
   Even if the hash stored in file is leaked to attackers. Even then they won’t be able to extract OTP from the hash if N is very large. They won’t be able to brute force if N is very high.

2) I created a echo server which basically prints back whatever the user has entered. Below image show that text is not encrypted.



731 640.430289 192.168.1.8 192.168.150.128 TCP 58 5555 → 40270 [PSH, ACK] Seq=1 Ack=5 Win=64240
732 640.430345 192.168.150.128 192.168.1.8 TCP 60 40270 → 5555 [ACK] Seq=5 Ack=5 Win=29200 Len=0
733 659.577542 192.168.150.128 192.168.1.8 TCP 121 40270 → 5555 [PSH, ACK] Seq=5 Ack=5 Win=29200
734 659.577678 192.168.1.8 192.168.150.128 TCP 54 5555 → 40270 [ACK] Seq=5 Ack=72 Win=64240 Len
735 659.577885 192.168.1.8 192.168.150.128 TCP 121 5555 → 40270 [PSH, ACK] Seq=5 Ack=72 Win=6424
736 659.577978 192.168.150.128 192.168.1.8 TCP 60 40270 → 5555 [ACK] Seq=72 Ack=72 Win=29200 Le
Frame 735: 121 bytes on wire (968 bits), 121 bytes captured (968 bits) on interface 0
Ethernet II, Src: Vmware_f4:f1:d1 (00:50:56:f4:f1:d1), Dst: Vmware_1b:48:b0 (00:0c:29:1b:48:b0)
Internet Protocol Version 4, Src: 192.168.1.8, Dst: 192.168.150.128
Transmission Control Protocol, Src Port: 5555, Dst Port: 40270, Seq: 5, Ack: 72, Len: 67
Data (67 bytes)
Data: 5468697320776173207461732074616b656e20616674657220692072...
Text: This was taken after i reached my home, thats why ip are different\n
[Length: 67]

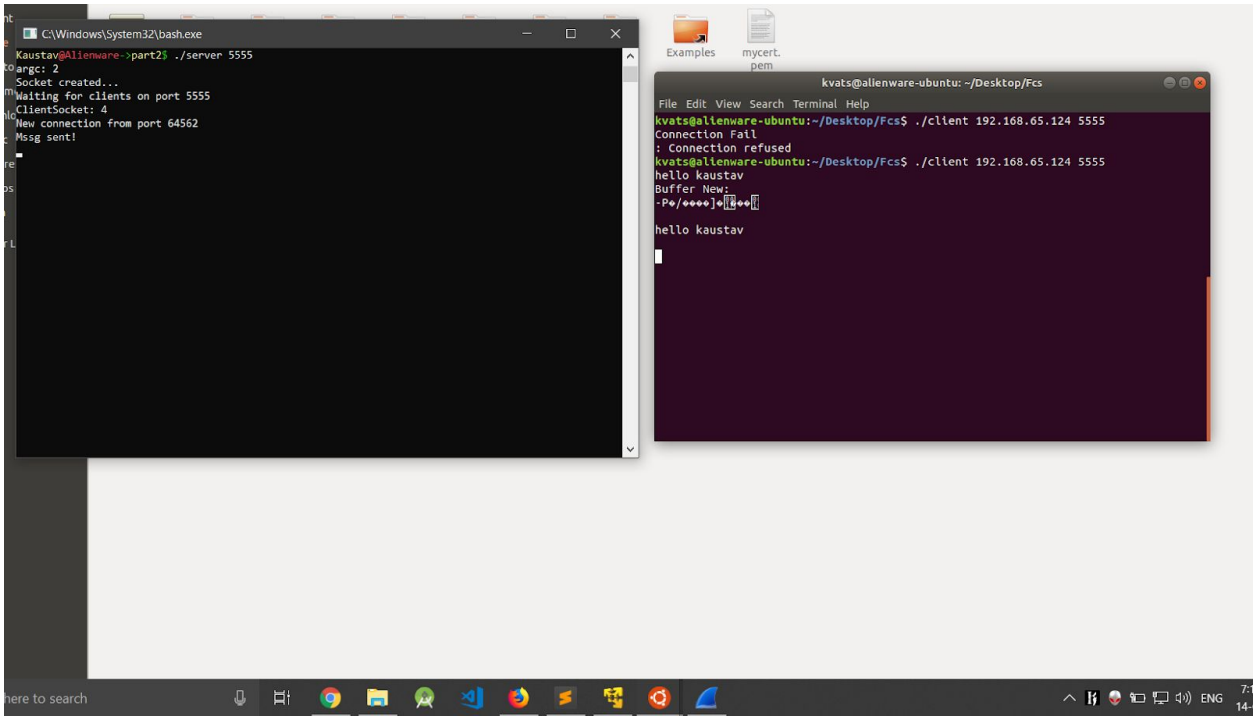
Below image is after encryption



93	186.375134	117.18.232.200	192.168.150.128	TCP	54	443 → 49868 [ACK] Seq=9258 Ack=712 Wi
94	186.379468	117.18.232.200	192.168.150.128	TLSv1.2	92	Application Data
95	186.387460	117.18.232.200	192.168.150.128	TLSv1.2	329	Application Data
96	186.388518	192.168.150.128	117.18.232.200	TCP	60	49868 → 443 [ACK] Seq=712 Ack=9571 Wi
97	191.582047	Vmware_1b:48:b0	Vmware_f4:f1:d1	ARP	60	Who has 192.168.150.2? Tell 192.168.1
98	191.582138	Vmware_f4:f1:d1	Vmware_1b:48:b0	ARP	42	192.168.150.2 is at 00:50:56:f4:f1:d1
99	193.510704	192.168.150.128	91.189.89.198	NTP	90	NTP Version 4, client
100	193.693972	91.189.89.198	192.168.150.128	NTP	90	NTP Version 4, server

```
> Frame 46: 70 bytes on wire (560 bits), 70 bytes captured (560 bits) on interface 0
> Ethernet II, Src: Vmware_f4:f1:d1 (00:50:56:f4:f1:d1), Dst: Vmware_1b:48:b0 (00:0c:29:1b:48:b0)
> Internet Protocol Version 4, Src: 192.168.65.124, Dst: 192.168.150.128
> Transmission Control Protocol, Src Port: 5555, Dst Port: 50844, Seq: 1, Ack: 17, Len: 16
v Data (16 bytes)
    Data: 27b60502b7dbe9ef47ef983042232060
    Text: '\357\277\275\005\002\357\277\275\357\277\275\357\277\275\357\277\275G\357\277\275\357\277\2750B# `
    [Length: 16]
```

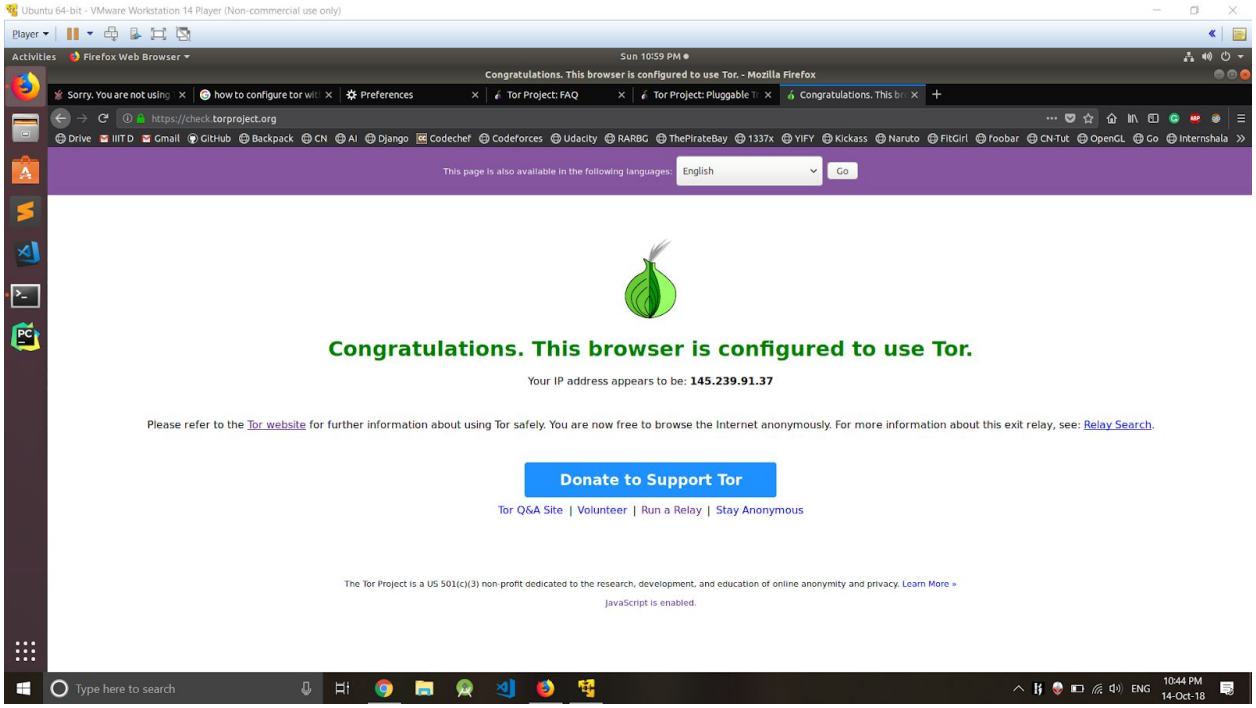
```
0000  00 0c 29 1b 48 b0 00 50  56 f4 f1 d1 08 00 45 00   ··)·H··P V·····E·
0010  00 38 28 3f 00 00 80 06  b9 33 c0 a8 41 7c c0 a8   ·8(?···· ·3··A|··
0020  96 80 15 b3 c6 9c 4c 1f  29 64 49 96 19 e8 50 18   ······L· )dI···P·
0030  fa f0 95 05 00 00 27 b6  05 02 b7 db e9 ef 47 ef   ······'· ······G·
0040  98 30 42 23 20 60                                 ·0B# `
```



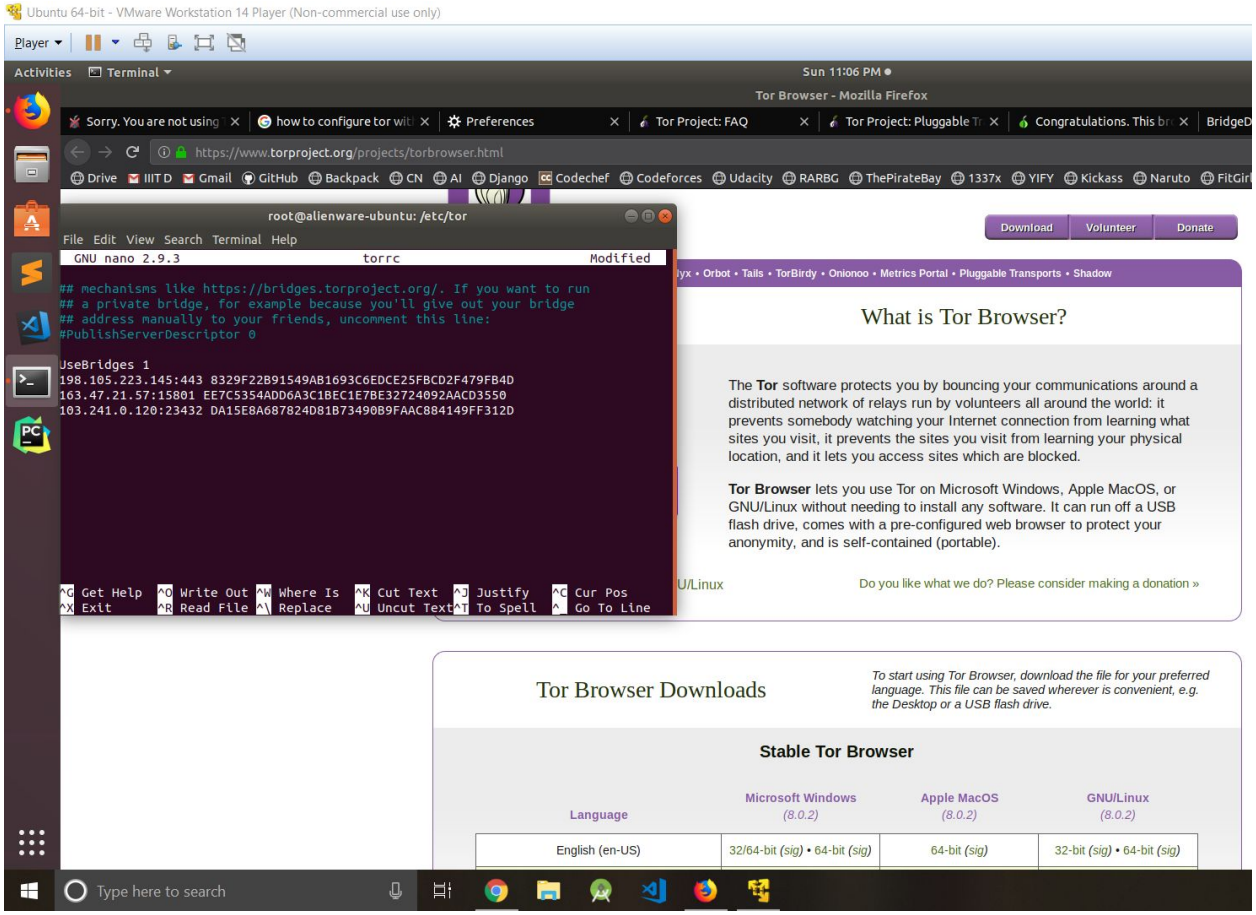
Buffer New is the encrypted text which was encrypted by client and then send to an echo server, Client decrypts and print the message.

To sniff packets i created Server on Main OS & Client on VM.

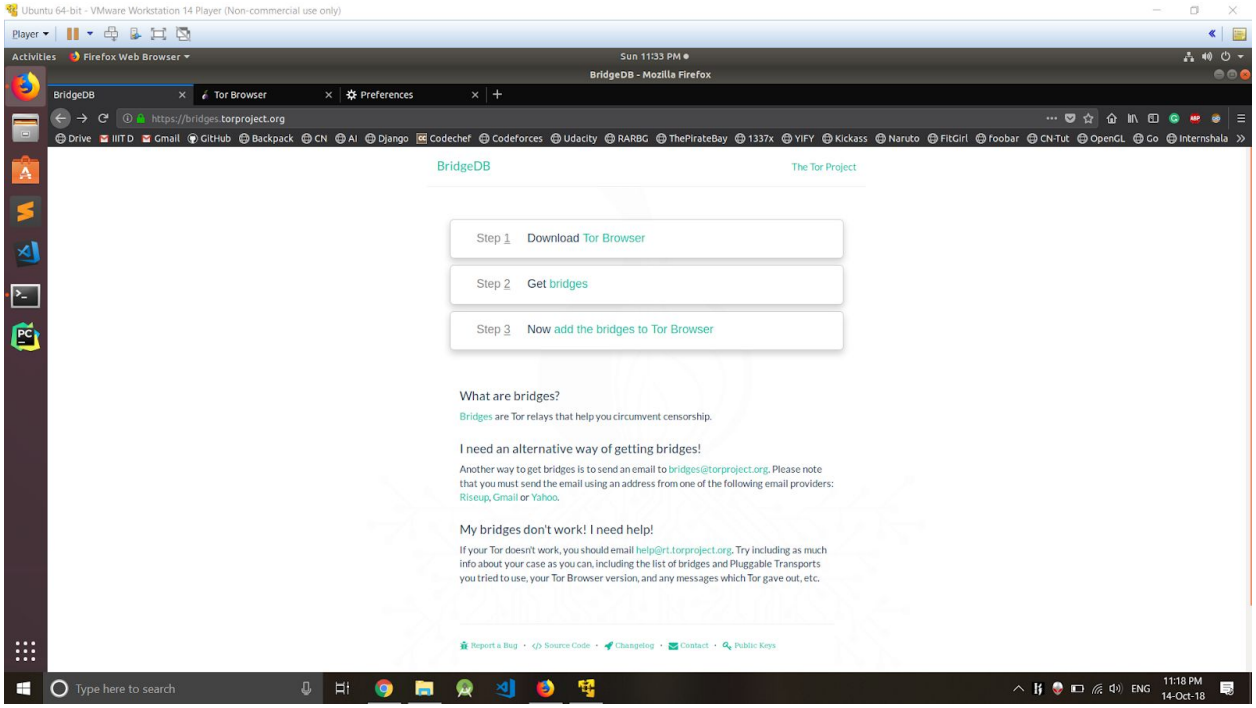
3) Configured my firefox to use Tor.



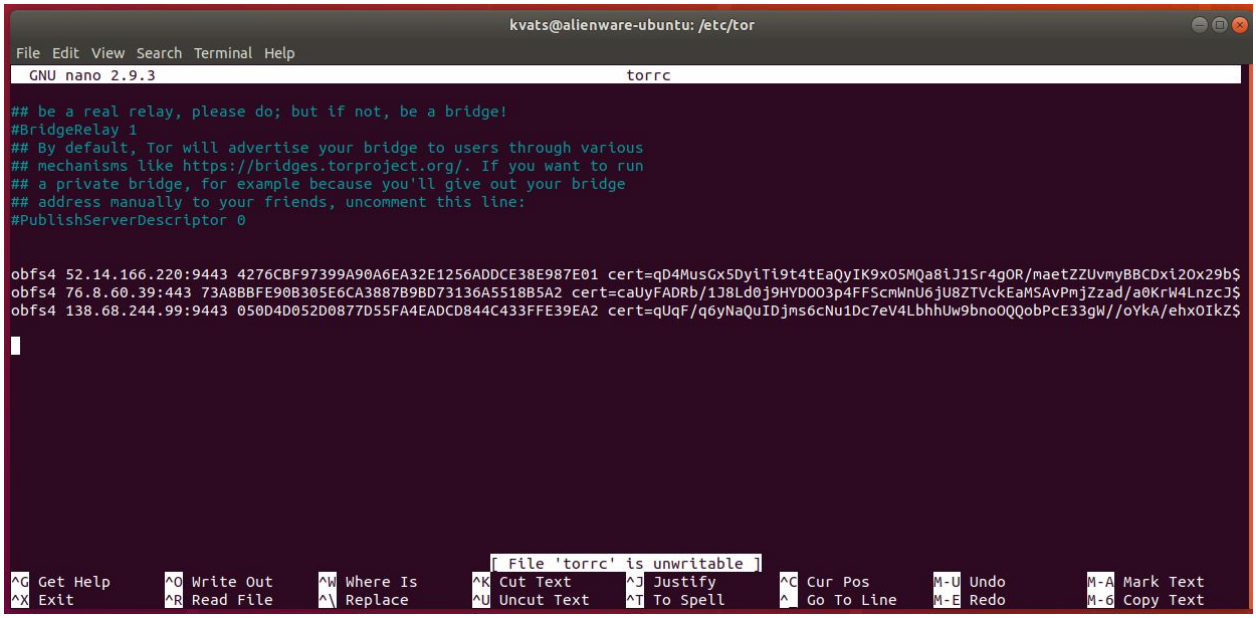
Used this link to configure my browser.
https://www.torproject.org/docs/debian.html.en



Configured for using bridges.
Website used: https://bridges.torproject.org/

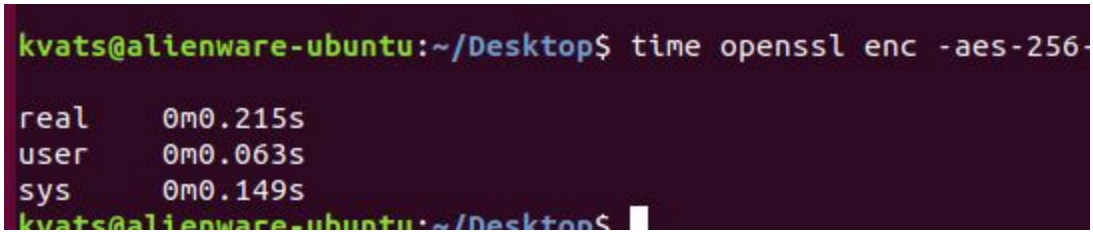


Above website for using Tor bridge

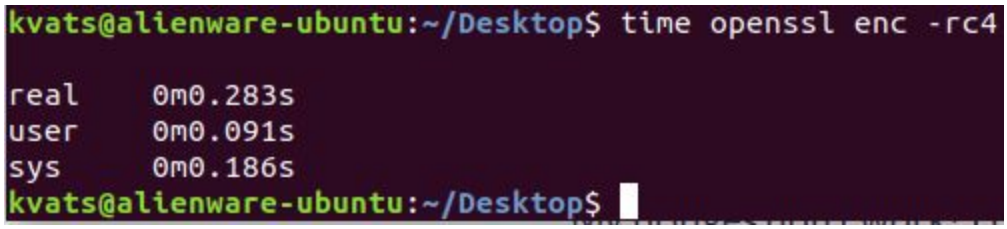


Configured for using obs4.

4) Time take is attached below



Time taken by AES-256 Encryption



Time taken by RC4

RSA was taking much more time than RC4 and AES where as AES-256 performs best as compared to other algorithms.