EXPERIMENT - 7

1.Build a web server using apache

1.1 Create directory structure and granting permission

```
kali@kali:/$ sudo mkdir -p /var/www/userabc.com
kali@kali:/$ sudo mkdir -p /var/www/userzyx.com
kali@kali:/$
kali@kali:/$ sudo chown -R $USER:$USER /var/www/userabc.com
kali@kali:/$ sudo chown -R $USER:$USER /var/www/userzyx.com
kali@kali:/$
```

1.2 Creating demo page for each virtual host

```
kali@kali:/$ sudo vim /var/www/userabc.com/index.html
kali@kali:/$ cp /var/www/userabc.com/index.html /var/www/userzyx.com/index.html
kali@kali:/$ sudo vim /var/www/userzyx.com/index.html
kali@kali:/$
```

```
<html>
        <head>
                <title>userabc</title>
        </head>
        <body>
                <h1>Welcome to userabc.com</h1>
        </body>
</html>
```

```
<html>
        <head>
                <title>usezyx</title>
        </head>
        <body>
                <h1>Welcome to userzyx.com</h1>
        </body>
</html>
```

1.3 Creating new virtual host files

Created two virtual hosts configuration files userabc.com.conf and userzyx.com.conf.

```
kali@kali:/$ sudo cp /etc/apache2/sites-available/000-default.conf /etc/apache2/sites-available/userabc.com.conf
kali@kali:/$ sudo vim /etc/apache2/sites-available/userabc.com.conf
kali@kali:/$ sudo cp /etc/apache2/sites-available/userabc.com.conf /etc/apache2/sites-available/userzyx.com.conf
kali@kali:/$ sudo vim /etc/apache2/sites-available/userzyx.com.conf
```

```
<VirtualHost *:80>

        ServerAdmin admin@userabc.com
        ServerName userabc.com
        ServerAlias www.userabc.com
        DocumentRoot /var/www/userabc.com
        ErrorLog ${APACHE_LOG_DIR}/error.log
        CustomLog ${APACHE_LOG_DIR}/access.log combined

</VirtualHost>
```

```
<VirtualHost *:80>

        ServerAdmin admin@userzyx.com
        ServerName userzyx.com
        ServerAlias www.userzyx.com
        DocumentRoot /var/www/userzyx.com
        ErrorLog ${APACHE_LOG_DIR}/error.log
        CustomLog ${APACHE_LOG_DIR}/access.log combined

</VirtualHost>
```

1.4 Enable new virtual host file

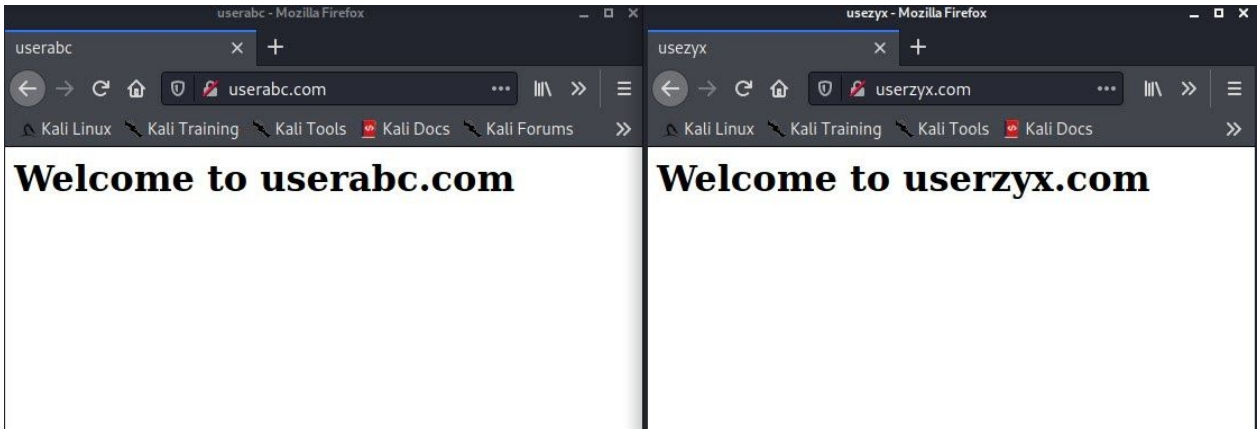
```
kali@kali:/$ sudo a2ensite userabc.com.conf
Enabling site userabc.com.
To activate the new configuration, you need to run:
  systemctl reload apache2
kali@kali:/$ sudo a2ensite userzyx.com.conf
Enabling site userzyx.com.
To activate the new configuration, you need to run:
  systemctl reload apache2
kali@kali:/$ sudo systemctl restart apache2
```

1.5 Set up local hosts file

Enter *sudo vim /etc/hosts* to get into local hosts file and created entry for our domains.

```
127.0.0.1       localhost
127.0.1.1       kali
127.0.0.1       userabc.com
127.0.0.1       userzyx.com
# The following lines are desirable for IPv6 capable hosts
::1     localhost ip6-localhost ip6-loopback
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
```

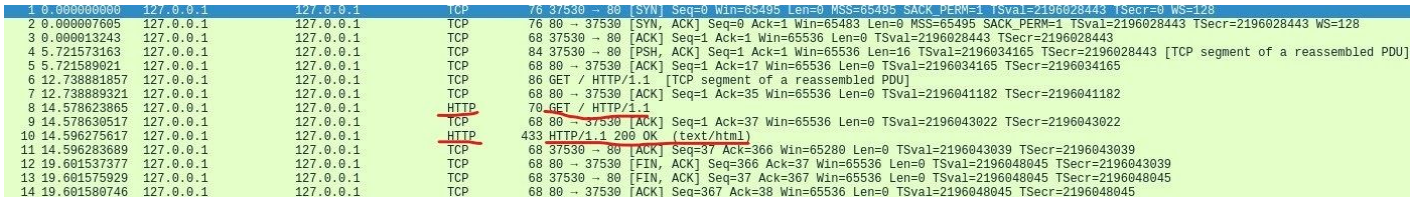
1.6 Testing the domains using browser



1.7 Telnet to the virtual host

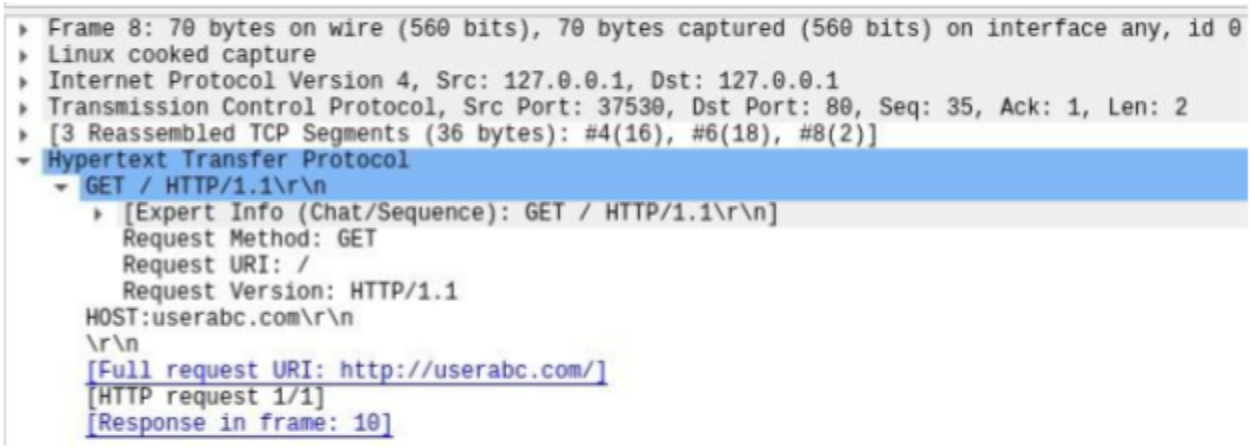
```
kali@kali:/$ sudo telnet userabc.com 80
Trying 127.0.0.1...
Connected to userabc.com.
Escape character is '^]'.
GET / HTTP/1.1
HOST:userabc.com

HTTP/1.1 200 OK
Date: Sun, 14 Mar 2021 07:59:38 GMT
Server: Apache/2.4.46 (Debian)
Last-Modified: Sun, 14 Mar 2021 07:59:47 GMT
ETag: W/"70-5bd7bccf568e6"
Accept-Ranges: bytes
Content-Length: 112
Vary: Accept-Encoding
Content-Type: text/html

<html>
        <head>
                <title>userabc</title>
        </head>
        <body>
                <h1>Welcome to userabc.com</h1>
        </body>
</html>
```

1.8 Wireshark packet captured



1	0.000000000	127.0.0.1	127.0.0.1	TCP	76 37530 → 80 [SYN] Seq=0 Win=65495 Len=0 MSS=65495 SACK_PERM=1 TSval=2196028443 TSecr=0 WS=128
2	0.000007605	127.0.0.1	127.0.0.1	TCP	76 80 → 37530 [SYN, ACK] Seq=0 Ack=1 Win=65483 Len=0 MSS=65495 SACK_PERM=1 TSval=2196028443 TSecr=2196028443 WS=128
3	0.000013243	127.0.0.1	127.0.0.1	TCP	68 37530 → 80 [ACK] Seq=1 Ack=1 Win=65536 Len=0 TSval=2196028443 TSecr=2196028443
4	5.721573163	127.0.0.1	127.0.0.1	TCP	84 37530 → 80 [PSH, ACK] Seq=1 Ack=1 Win=65536 Len=16 TSval=2196034165 TSecr=2196028443 [TCP segment of a reassembled PDU]
5	5.721589021	127.0.0.1	127.0.0.1	TCP	68 80 → 37530 [ACK] Seq=1 Ack=17 Win=65536 Len=0 TSval=2196034165 TSecr=2196034165
6	12.738881857	127.0.0.1	127.0.0.1	TCP	86 GET / HTTP/1.1 [TCP segment of a reassembled PDU]
7	12.738889321	127.0.0.1	127.0.0.1	TCP	68 80 → 37530 [ACK] Seq=1 Ack=35 Win=65536 Len=0 TSval=2196041182 TSecr=2196041182
8	14.578623865	127.0.0.1	127.0.0.1	HTTP	70 GET / HTTP/1.1
9	14.578630517	127.0.0.1	127.0.0.1	TCP	68 80 → 37530 [ACK] Seq=1 Ack=37 Win=65536 Len=0 TSval=2196043022 TSecr=2196043022
10	14.596275617	127.0.0.1	127.0.0.1	HTTP	433 HTTP/1.1 200 OK (text/html)
11	14.596283689	127.0.0.1	127.0.0.1	TCP	68 37530 → 80 [ACK] Seq=37 Ack=366 Win=65280 Len=0 TSval=2196043039 TSecr=2196043039
12	19.601537377	127.0.0.1	127.0.0.1	TCP	68 80 → 37530 [FIN, ACK] Seq=366 Ack=37 Win=65536 Len=0 TSval=2196048045 TSecr=2196043039
13	19.601575929	127.0.0.1	127.0.0.1	TCP	68 37530 → 80 [FIN, ACK] Seq=37 Ack=367 Win=65536 Len=0 TSval=2196048045 TSecr=2196048045
14	19.601580746	127.0.0.1	127.0.0.1	TCP	68 80 → 37530 [ACK] Seq=367 Ack=38 Win=65536 Len=0 TSval=2196048045 TSecr=2196048045

Initially while telnetting to host *userabc.com.* A TCP 3-way handshake takes place inorder to establish a TCP connection between server and client. After establishing the connection, HTTP GET request is sent to the server and it's corresponding HTTP response packet received. After some time, the connection is terminated by sending TCP FIN packets from both sides. Both the HTTP headers of request and response packet are shown as expanded below.



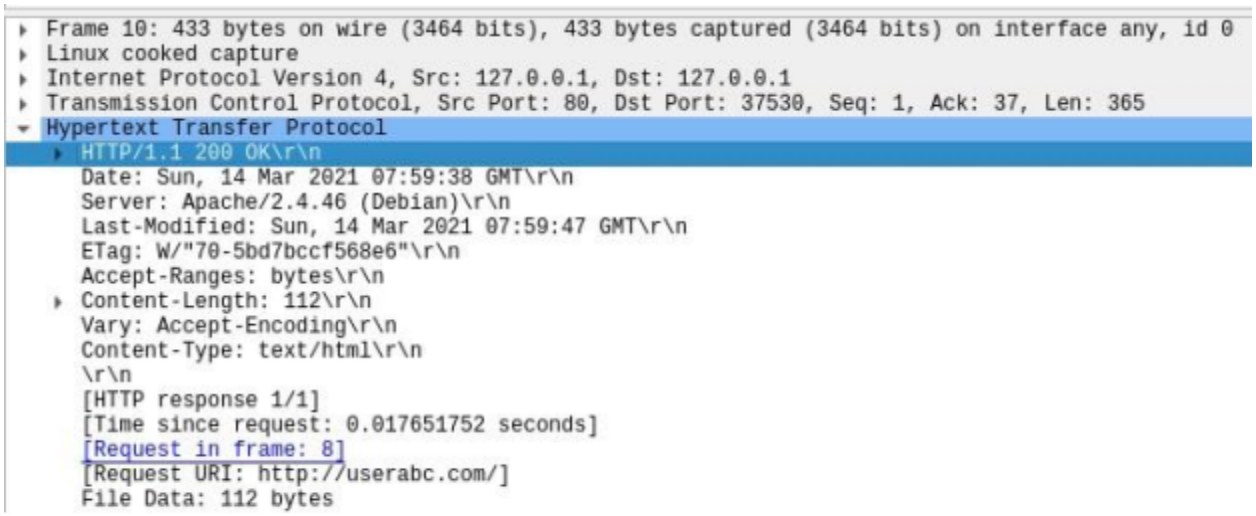
```
▸ Frame 8: 70 bytes on wire (560 bits), 70 bytes captured (560 bits) on interface any, id 0
▸ Linux cooked capture
▸ Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
▸ Transmission Control Protocol, Src Port: 37530, Dst Port: 80, Seq: 35, Ack: 1, Len: 2
▸ [3 Reassembled TCP Segments (36 bytes): #4(16), #6(18), #8(2)]
▾ Hypertext Transfer Protocol
  ▾ GET / HTTP/1.1\r\n
    ▸ [Expert Info (Chat/Sequence): GET / HTTP/1.1\r\n]
      Request Method: GET
      Request URI: /
      Request Version: HTTP/1.1
    HOST:userabc.com\r\n
    \r\n
    [Full request URI: http://userabc.com/]
    [HTTP request 1/1]
    [Response in frame: 10]
```

(HTTP Request packet)

The first line of the HTTP request is called the request line and consists of 3 parts:

- GET method
- The path given here is '/'. Which means the root file for the server.
- The "protocol" part contains "HTTP" and the version, which is usually 1.1

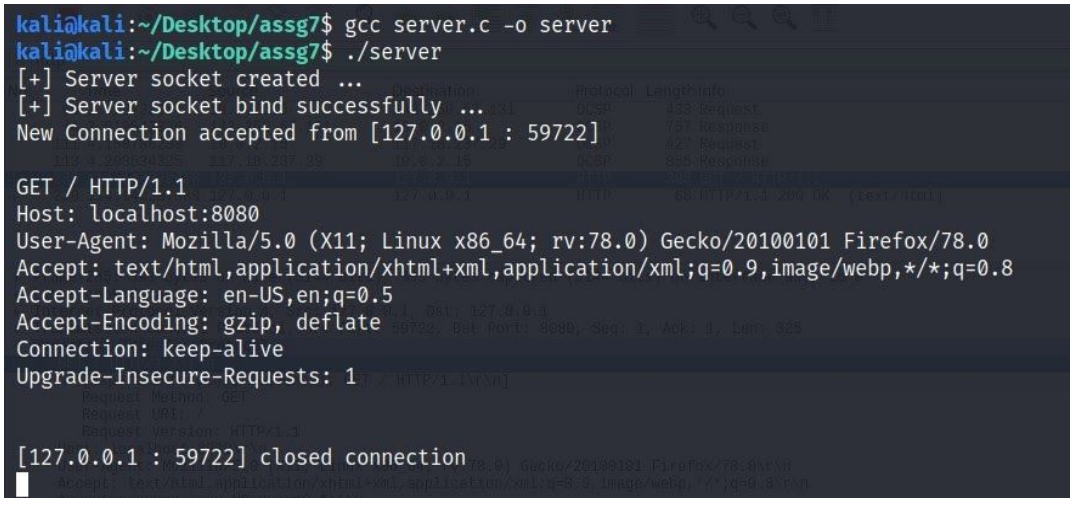
The remainder of the request contains HTTP headers as "Name: Value" pairs on each line. These contain various information about the HTTP request and your browser.



(HTTP Response Packet)

The first piece of data is the protocol. This is again usually HTTP/1.1.The next part is the status code followed by a short message. Code 200 means that our GET request was successful and the server will return the contents of the requested document, right after the headers.Then it is followed by a number of HTTP header “Name: value” pairs. It shows the date, server software and version, Last modified date etc. Content Length indicates the no.of bytes of data in the message. Here the Content-Type is text/html which tells the browser that data is an html file so that browser can interpret according to the type of document.

2.Very Simple Web Server



A server program running on localhost port no 8080. Whenever a client initiate a HTTP GET request at ‘/’, Then server sends the HTTP response packet to client as text.