# Assignment 3 Report COL759 : Heartbleed Vulnerability OpenSSL

2019MCS2574

Vivek Singh

## 1 What is OpenSSL

OpenSSL is a open source implementation of SSL/TLS protocol. The core program is written in C programming language. OpenSSL is a software library for applications that secure communications over computer networks against eavesdropping or need to identify the party at the other end. It is widely used by Internet servers, including the majority of HTTPS websites.

## 2 Working of OpenSSL

Since OpenSSL is based on SSL/TSL protocol it works similarly.

- The SSL/TSL protocol allows the server and client to:

  Authenticate one another.

  Negotiate and finalize an encryption and MAC algorithm

  Finalize and share the cryptographic keys to protect and securely transmit payload data.

- Once Client and Server makes a connection, the client request for a secure connection.

- After receiving the request server chooses a most secure option that is compatible with both client and server, and then sends a security certificate signed with the server's public key.

- The client verifies the certificate and generates a secret key to send to the server, encrypted with the server's public key.

- The client and server use the secret key to generate pair of symmetric keys (or two pairs of public-private keys), and communication commences securely.

## 3 Heartbleed Vulnerability

The OpenSSL provides secure transmission of data from client and server by using cryptographic methods. But in OpenSSL implementation of v1.01 and some other Beta versions a significant vulnerability was discovered called HeartBleed.

By exploiting this vulnerability a attacker may gather and revel information upto 64K of memory data in stored in server cache.

Hence by reading the server memory attacker can gain access to sensitive data such as login Ids, passwords and even senders private keys. Knowledge of private key will compromise the whole confidentiality of the data and the attacker can also perform man-in-middle attack.

## 4 Why and How of HeartBleed Vulnerability

One important part of the SSL/TSL protocols is what's called a heartbeat. When client and server are connected to one another using the secure connection, then to let each other know that they are still alive and connected client send the server a special request called **Heartbeat request**. The server responds to the client using the same message back.

The heartbeat request contains a payload message along with a size parameter passed which indicates the size of payload.

When server receives the heartbeat request it copies the payload of the given size present in the memory buffer and send it back to the client which requested the message to check the connection.

**Heartbleed code: memcpy(bp, pl, payload);**

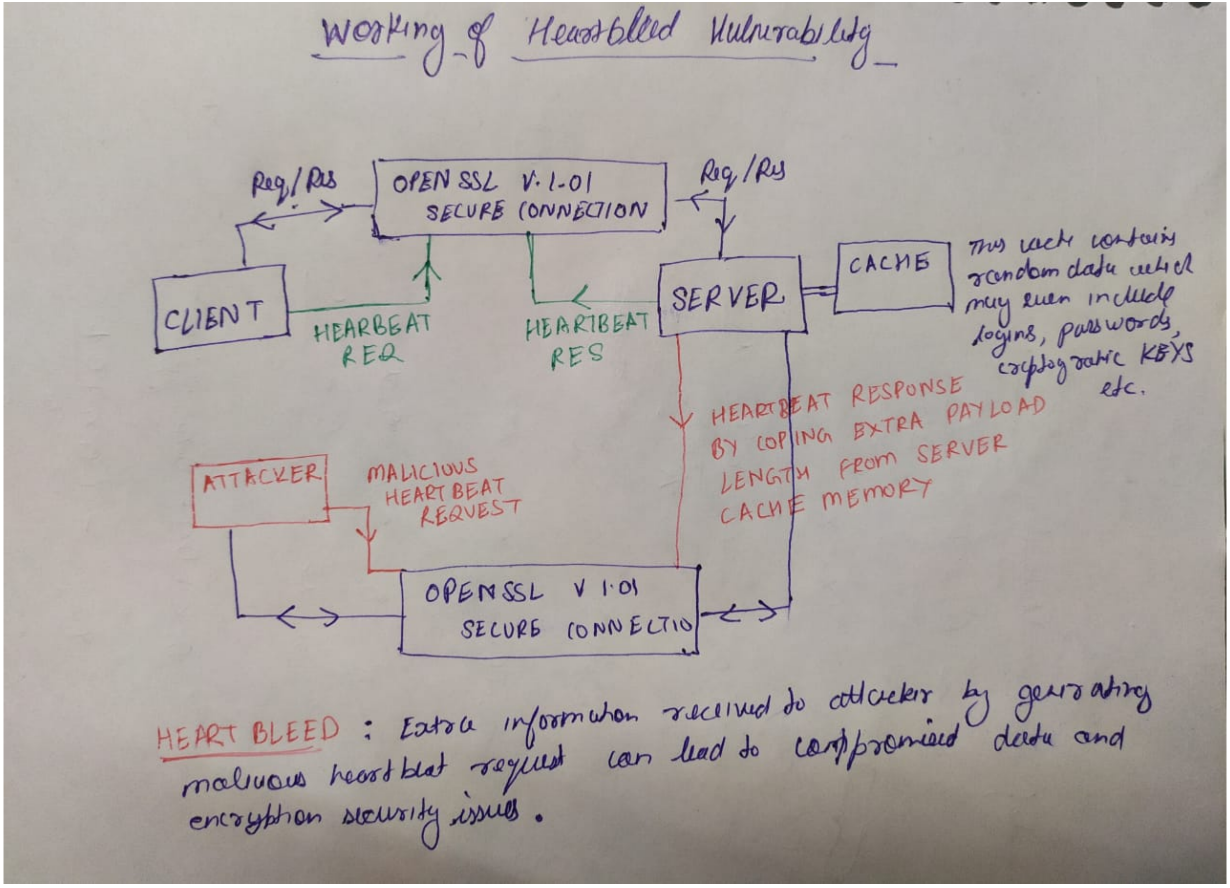
memcpy() is the command that copies data. bp is the place it's copying it to, pl is where it's being copied from, and payload is the length of the data being copied. The problem is that there's never any attempt to check if the amount of data in pl is equal to the value given of payload. This part of code written in C caused the venerability in OpenSSL v1.01.

Example of exploiting the vulnerability.

- As an attacker you will send a heartbeat request to the server.
- Let the request contains 10K byte of information
- The request is asking for 60K byte of information in payload length.
- This difference of 50K byte of information is filled in the memory buffer using the server cache. This cache can contain garbage values, login ids, last request or even cryptographic keys.
- So the OpenSSL memcpy() will add all the extra padding in the heartbeat response causing major information loss and compromise the security and confidentiality of the data.

# 5 Flow-Diagram of simulation

Figure 1: **Flow diagram showing the exploitation of heartbeat request by attacker**



This cache contains random data which may even include logins, passwords, cryptographic KEYS etc.

HEART BLEED : Extra information received to attacker by generating malicious heart beat request can lead to compromised data and encryption security issues.

# 6 Simulation of heart-Bleed vulnerability

- The simulation is based on following assumptions

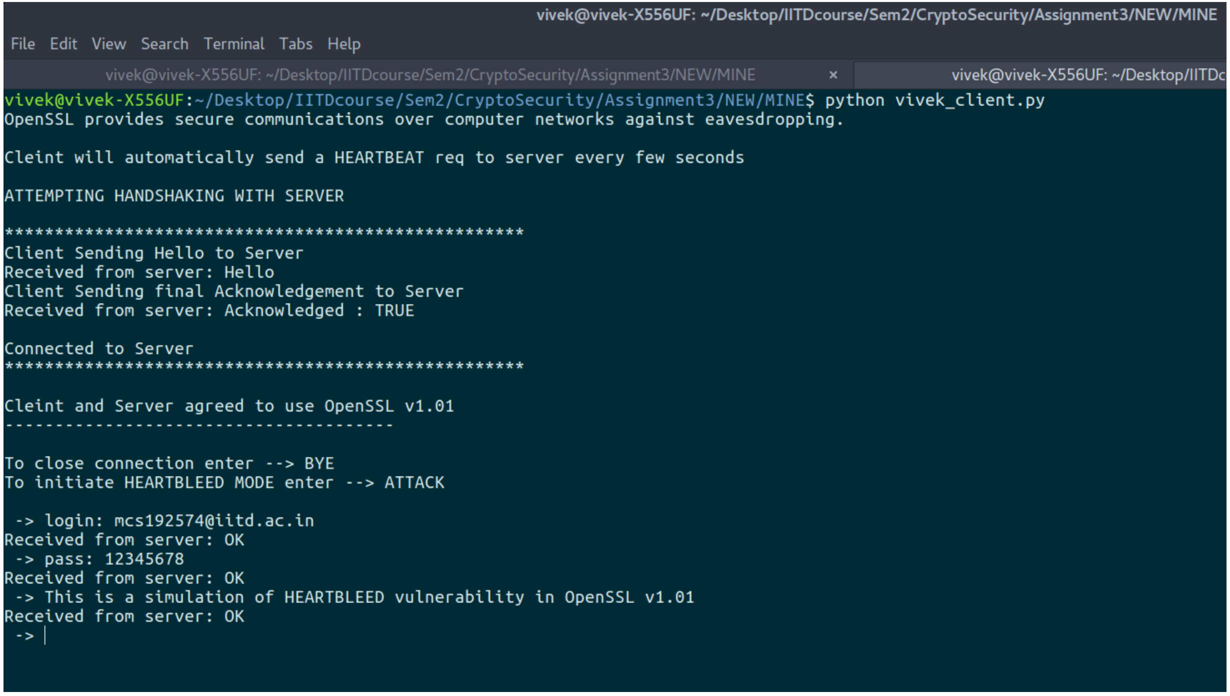
  Connection between server and other computer is secured by OpenSSL v1.01.

  A server is connected to client and communication is going on a secure channel.

  Client can also attack the server by sending carefully crafted malicious HEARTBEAT request to server.

- Server contains a cache memory which contains garbage values as well as recent information of the client server communication, request and responses by server to a client.

Figure 2: **Client is connected to server and communicating over OpenSSL secure connection.**

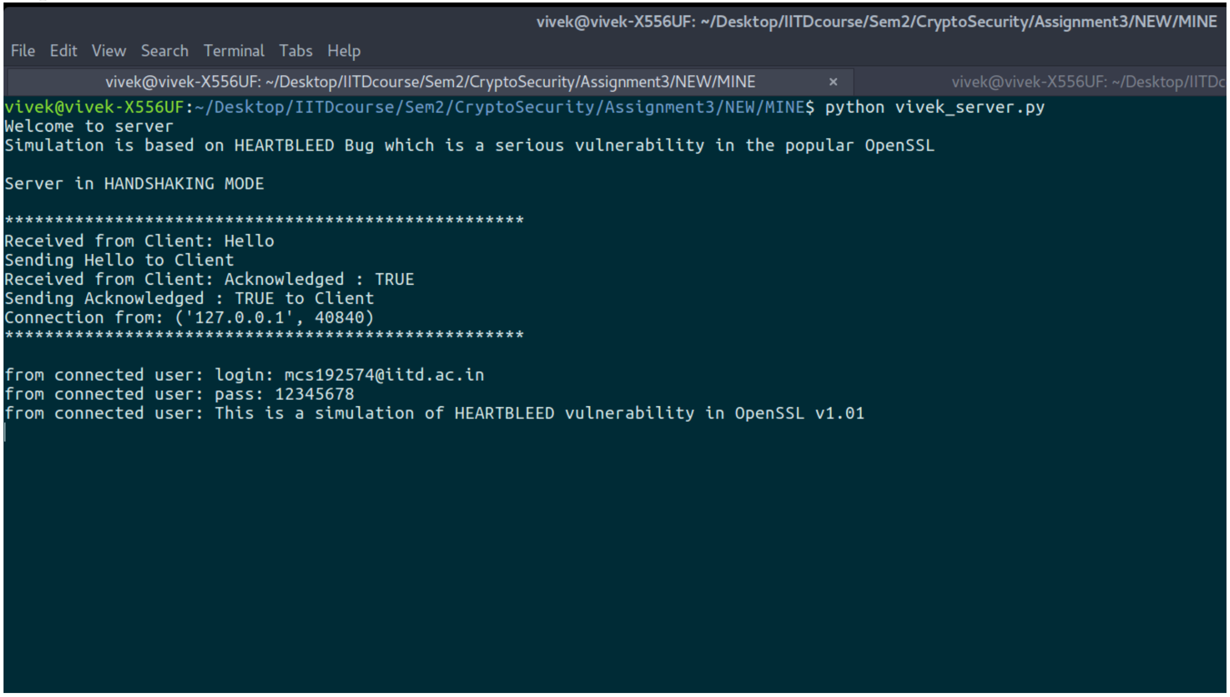


vivek@vivek-X556UF: ~/Desktop/IITDcourse/Sem2/CryptoSecurity/Assignment3/NEW/MINE
File Edit View Search Terminal Tabs Help
vivek@vivek-X556UF: ~/Desktop/IITDcourse/Sem2/CryptoSecurity/Assignment3/NEW/MINE
×
vivek@vivek-X556UF: ~/Desktop/IITDc

```
vivek@vivek-X556UF:~/Desktop/IITDcourse/Sem2/CryptoSecurity/Assignment3/NEW/MINE$ python vivek_client.py
OpenSSL provides secure communications over computer networks against eavesdropping.

Cleint will automatically send a HEARTBEAT req to server every few seconds

ATTEMPTING HANDSHAKING WITH SERVER

****************************************************
Client Sending Hello to Server
Received from server: Hello
Client Sending final Acknowledgement to Server
Received from server: Acknowledged : TRUE

Connected to Server
****************************************************

Cleint and Server agreed to use OpenSSL v1.01
---------------------------------------

To close connection enter --> BYE
To initiate HEARTBLEED MODE enter --> ATTACK

 -> login: mcs192574@iitd.ac.in
Received from server: OK
 -> pass: 12345678
Received from server: OK
 -> This is a simulation of HEARTBLEED vulnerability in OpenSSL v1.01
Received from server: OK
 -> |
```

- Client send server heartbeat request regularly and periodically to insure that the connection is maintained.
- Once server receives the heartbeat request it replies with appropriate heartbeat response according to the payload information received from in request parameter.

Figure 3: **Server side of the connection where it communicates with the client and also ensures safe connection by responding to heartbeat request.**

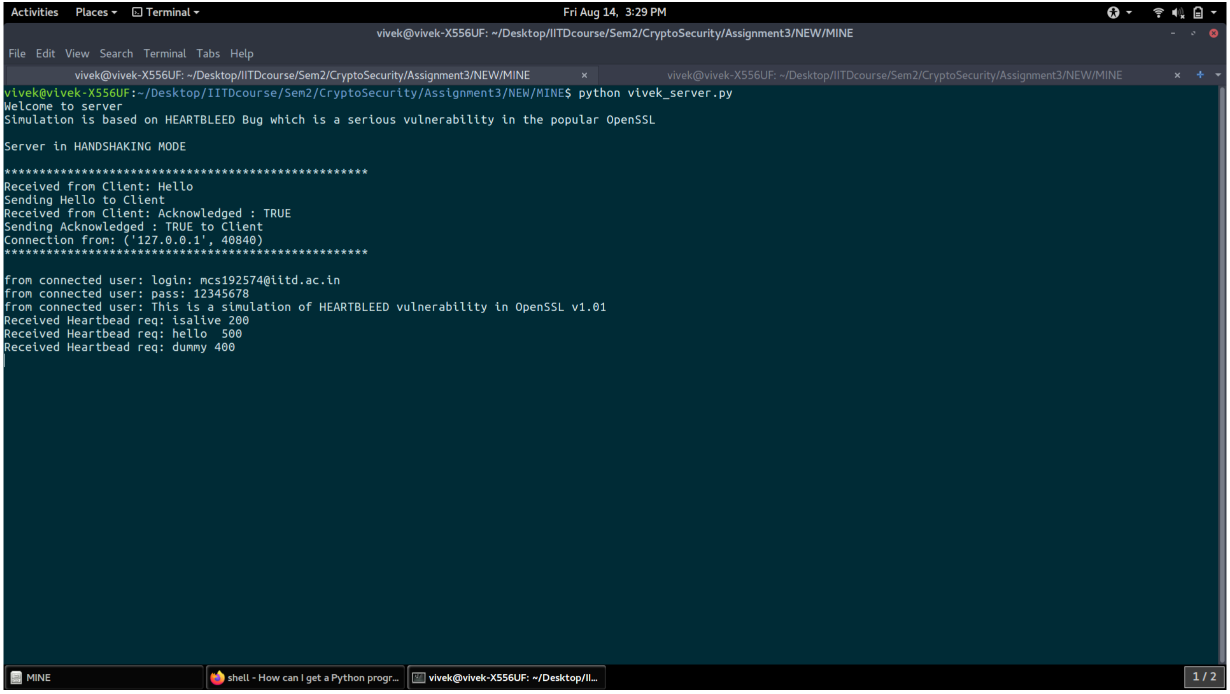

vivek@vivek-X556UF: ~/Desktop/IITDcourse/Sem2/CryptoSecurity/Assignment3/NEW/MINE
File Edit View Search Terminal Tabs Help
vivek@vivek-X556UF: ~/Desktop/IITDcourse/Sem2/CryptoSecurity/Assignment3/NEW/MINE
vivek@vivek-X556UF: ~/Desktop/IITDc

```
vivek@vivek-X556UF:~/Desktop/IITDcourse/Sem2/CryptoSecurity/Assignment3/NEW/MINE$ python vivek_server.py
Welcome to server
Simulation is based on HEARTBLEED Bug which is a serious vulnerability in the popular OpenSSL

Server in HANDSHAKING MODE

****************************************************
Received from Client: Hello
Sending Hello to Client
Received from Client: Acknowledged : TRUE
Sending Acknowledged : TRUE to Client
Connection from: ('127.0.0.1', 40840)
****************************************************

from connected user: login: mcs192574@iitd.ac.in
from connected user: pass: 12345678
from connected user: This is a simulation of HEARTBLEED vulnerability in OpenSSL v1.01
```

- The client and server are connected to each other after a handshaking protocol and agree upon a common protocol that is OpenSSL v1.01 to communicate with each other.

- Now Clinet goes into attacking mode by exploiting OpenSSL vulnerrability, generate carefully crafted heartbeat request to perform a heartbleed attack.

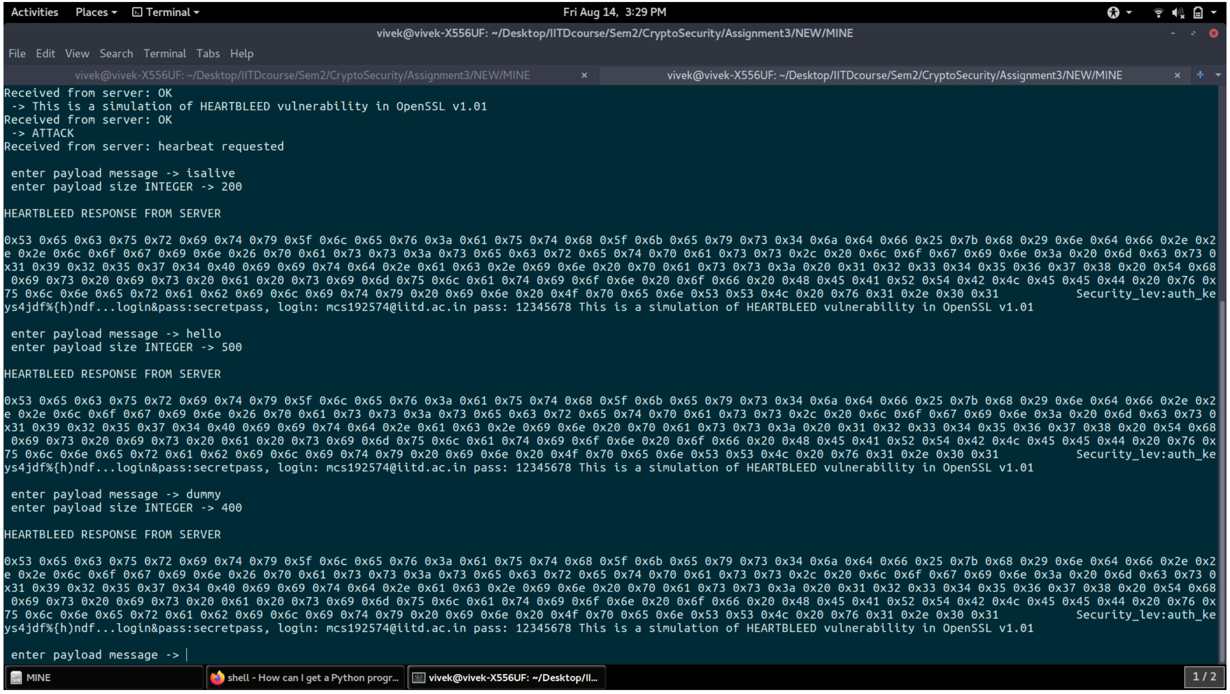
Figure 4: **Server side of the connection where it communicates with the attacker and respond to the malicious request generated by attacker and sends sensitive information stored in its cache.**


Activities
Places
Terminal
Fri Aug 14, 3:29 PM
vivek@vivek-X556UF: ~/Desktop/IITDcourse/Sem2/CryptoSecurity/Assignment3/NEW/MINE
File Edit View Search Terminal Tabs Help
vivek@vivek-X556UF: ~/Desktop/IITDcourse/Sem2/CryptoSecurity/Assignment3/NEW/MINE
vivek@vivek-X556UF: ~/Desktop/IITDcourse/Sem2/CryptoSecurity/Assignment3/NEW/MINE
MINE
shell - How can I get a Python progr...
vivek@vivek-X556UF: ~/Desktop/II...
1 / 2

```
vivek@vivek-X556UF:~/Desktop/IITDcourse/Sem2/CryptoSecurity/Assignment3/NEW/MINE$ python vivek_server.py
Welcome to server
Simulation is based on HEARTBLEED Bug which is a serious vulnerability in the popular OpenSSL

Server in HANDSHAKING MODE

****************************************************
Received from Client: Hello
Sending Hello to Client
Received from Client: Acknowledged : TRUE
Sending Acknowledged : TRUE to Client
Connection from: ('127.0.0.1', 40840)
****************************************************

from connected user: login: mcs192574@iitd.ac.in
from connected user: pass: 12345678
from connected user: This is a simulation of HEARTBLEED vulnerability in OpenSSL v1.01
Received Heartbead req: isalive 200
Received Heartbead req: hello  500
Received Heartbead req: dummy 400
```

- Here we can see that server when connected to new user(attacker) has served 3 heartbeat request and responded to the request.
- These response might contain information extracted from its cache.

Figure 5: **Attacker sending lager payload length to capture information in server's cache**


Activities
Places
Terminal
Fri Aug 14, 3:29 PM
vivek@vivek-X556UF: ~/Desktop/IITDcourse/Sem2/CryptoSecurity/Assignment3/NEW/MINE
File Edit View Search Terminal Tabs Help
vivek@vivek-X556UF: ~/Desktop/IITDcourse/Sem2/CryptoSecurity/Assignment3/NEW/MINE
vivek@vivek-X556UF: ~/Desktop/IITDcourse/Sem2/CryptoSecurity/Assignment3/NEW/MINE
MINE
shell - How can I get a Python progr...
vivek@vivek-X556UF: ~/Desktop/II...
1 / 2

```
Received from server: OK
 -> This is a simulation of HEARTBLEED vulnerability in OpenSSL v1.01
Received from server: OK
 -> ATTACK
Received from server: hearbeat requested

 enter payload message -> isalive
 enter payload size INTEGER -> 200

HEARTBLEED RESPONSE FROM SERVER

0x53 0x65 0x63 0x75 0x72 0x69 0x74 0x79 0x5f 0x6c 0x65 0x76 0x3a 0x61 0x75 0x74 0x68 0x5f 0x6b 0x65 0x79 0x73 0x34 0x6a 0x64 0x66 0x25 0x7b 0x68 0x29 0x6e 0x64 0x66 0x2e 0x2
e 0x2e 0x6c 0x6f 0x67 0x69 0x6e 0x26 0x70 0x61 0x73 0x73 0x3a 0x73 0x65 0x63 0x72 0x65 0x74 0x70 0x61 0x73 0x73 0x2c 0x20 0x6c 0x6f 0x67 0x69 0x6e 0x3a 0x20 0x6d 0x63 0x73 0
x31 0x39 0x32 0x35 0x37 0x34 0x40 0x69 0x69 0x74 0x64 0x2e 0x61 0x63 0x2e 0x69 0x6e 0x20 0x70 0x61 0x73 0x73 0x3a 0x20 0x31 0x32 0x33 0x34 0x35 0x36 0x37 0x38 0x20 0x54 0x68
 0x69 0x73 0x20 0x69 0x73 0x20 0x61 0x20 0x73 0x69 0x6d 0x75 0x6c 0x61 0x74 0x69 0x6f 0x6e 0x20 0x6f 0x66 0x20 0x48 0x45 0x41 0x52 0x54 0x42 0x4c 0x45 0x45 0x44 0x20 0x76 0x
75 0x6c 0x6e 0x65 0x72 0x61 0x62 0x69 0x6c 0x69 0x74 0x79 0x20 0x69 0x6e 0x20 0x4f 0x70 0x65 0x6e 0x53 0x53 0x4c 0x20 0x76 0x31 0x2e 0x30 0x31           Security_lev:auth_ke
ys4jdf%{h)ndf...login&pass:secretpass, login: mcs192574@iitd.ac.in pass: 12345678 This is a simulation of HEARTBLEED vulnerability in OpenSSL v1.01

 enter payload message -> hello
 enter payload size INTEGER -> 500

HEARTBLEED RESPONSE FROM SERVER

0x53 0x65 0x63 0x75 0x72 0x69 0x74 0x79 0x5f 0x6c 0x65 0x76 0x3a 0x61 0x75 0x74 0x68 0x5f 0x6b 0x65 0x79 0x73 0x34 0x6a 0x64 0x66 0x25 0x7b 0x68 0x29 0x6e 0x64 0x66 0x2e 0x2
e 0x2e 0x6c 0x6f 0x67 0x69 0x6e 0x26 0x70 0x61 0x73 0x73 0x3a 0x73 0x65 0x63 0x72 0x65 0x74 0x70 0x61 0x73 0x73 0x2c 0x20 0x6c 0x6f 0x67 0x69 0x6e 0x3a 0x20 0x6d 0x63 0x73 0
x31 0x39 0x32 0x35 0x37 0x34 0x40 0x69 0x69 0x74 0x64 0x2e 0x61 0x63 0x2e 0x69 0x6e 0x20 0x70 0x61 0x73 0x73 0x3a 0x20 0x31 0x32 0x33 0x34 0x35 0x36 0x37 0x38 0x20 0x54 0x68
 0x69 0x73 0x20 0x69 0x73 0x20 0x61 0x20 0x73 0x69 0x6d 0x75 0x6c 0x61 0x74 0x69 0x6f 0x6e 0x20 0x6f 0x66 0x20 0x48 0x45 0x41 0x52 0x54 0x42 0x4c 0x45 0x45 0x44 0x20 0x76 0x
75 0x6c 0x6e 0x65 0x72 0x61 0x62 0x69 0x6c 0x69 0x74 0x79 0x20 0x69 0x6e 0x20 0x4f 0x70 0x65 0x6e 0x53 0x53 0x4c 0x20 0x76 0x31 0x2e 0x30 0x31           Security_lev:auth_ke
ys4jdf%{h)ndf...login&pass:secretpass, login: mcs192574@iitd.ac.in pass: 12345678 This is a simulation of HEARTBLEED vulnerability in OpenSSL v1.01

 enter payload message -> dummy
 enter payload size INTEGER -> 400

HEARTBLEED RESPONSE FROM SERVER

0x53 0x65 0x63 0x75 0x72 0x69 0x74 0x79 0x5f 0x6c 0x65 0x76 0x3a 0x61 0x75 0x74 0x68 0x5f 0x6b 0x65 0x79 0x73 0x34 0x6a 0x64 0x66 0x25 0x7b 0x68 0x29 0x6e 0x64 0x66 0x2e 0x2
e 0x2e 0x6c 0x6f 0x67 0x69 0x6e 0x26 0x70 0x61 0x73 0x73 0x3a 0x73 0x65 0x63 0x72 0x65 0x74 0x70 0x61 0x73 0x73 0x2c 0x20 0x6c 0x6f 0x67 0x69 0x6e 0x3a 0x20 0x6d 0x63 0x73 0
x31 0x39 0x32 0x35 0x37 0x34 0x40 0x69 0x69 0x74 0x64 0x2e 0x61 0x63 0x2e 0x69 0x6e 0x20 0x70 0x61 0x73 0x73 0x3a 0x20 0x31 0x32 0x33 0x34 0x35 0x36 0x37 0x38 0x20 0x54 0x68
 0x69 0x73 0x20 0x69 0x73 0x20 0x61 0x20 0x73 0x69 0x6d 0x75 0x6c 0x61 0x74 0x69 0x6f 0x6e 0x20 0x6f 0x66 0x20 0x48 0x45 0x41 0x52 0x54 0x42 0x4c 0x45 0x45 0x44 0x20 0x76 0x
75 0x6c 0x6e 0x65 0x72 0x61 0x62 0x69 0x6c 0x69 0x74 0x79 0x20 0x69 0x6e 0x20 0x4f 0x70 0x65 0x6e 0x53 0x53 0x4c 0x20 0x76 0x31 0x2e 0x30 0x31           Security_lev:auth_ke
ys4jdf%{h)ndf...login&pass:secretpass, login: mcs192574@iitd.ac.in pass: 12345678 This is a simulation of HEARTBLEED vulnerability in OpenSSL v1.01

 enter payload message -> 
```

- Here we can see that attacker is able to extract sensitive information from server cache.
- It includes recent communication along with login ids and password excahnged by client and server.
- A README is provided in the code to help run the simulation. Make sure to send **bye** mssg by client and end connection with it before starting the attacker script.
- Hence attacker was able to gain access to senstive data using OpenSSL heartbeat request resulting in major security flaw.
- Later updates of OpenSSL have fixed this issue by keeping a check on memcpy() function to detect any malicious request.