# 50.020 Network Security Lab 3 pt 2| 1002853 Wong Chi Seng

## Task 3.1

Before the running the attack, be sure to obtain the correct script from eDimension as the ones found online may not be compatible with the python version on the lab machine. Change the file to be executable, if not you will get nasty errors. Nawt kewl

Getting the username and password through the heartbleed attack:

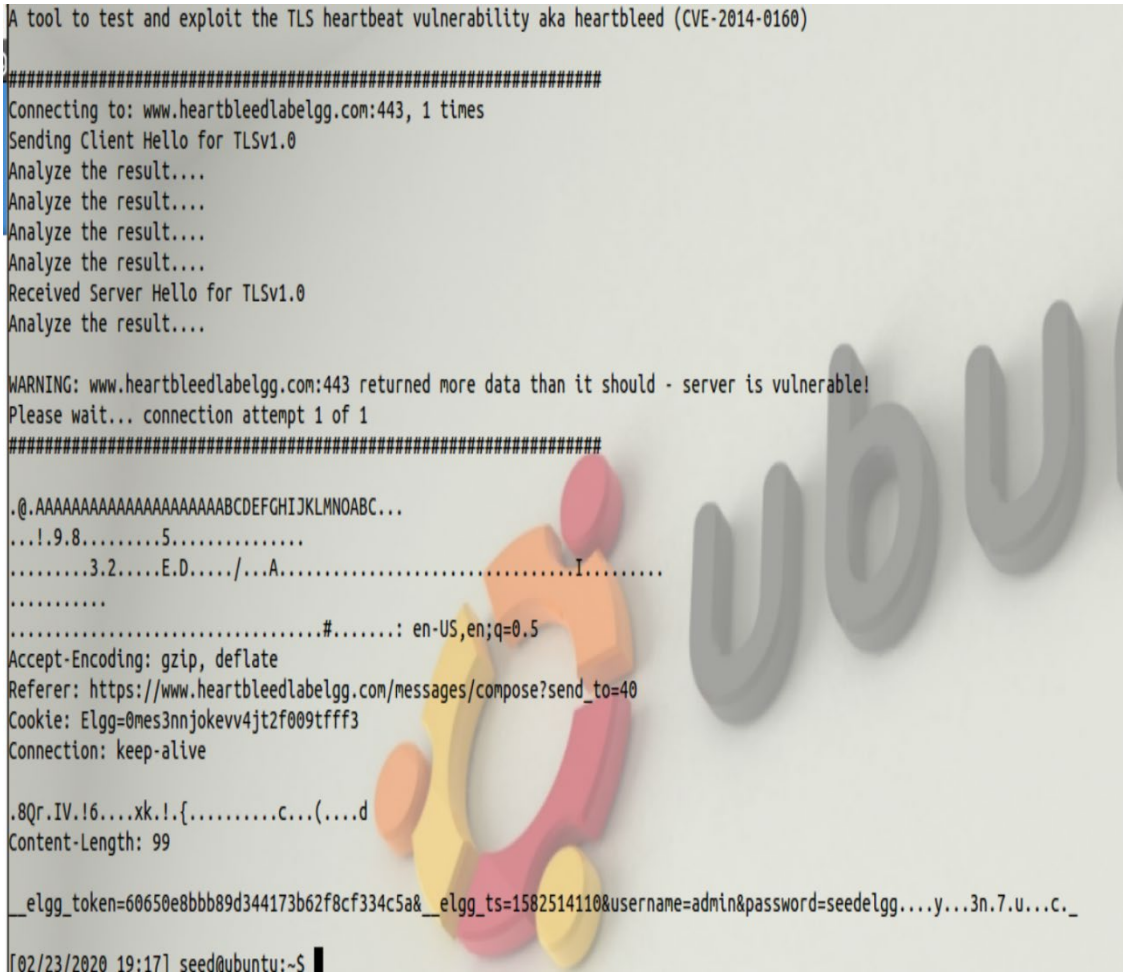


*Figure 1 user and password*

Getting the user activity from the attack. Shows that the admin is sending a message to the user with id 42.


*Figure 2 activity*

Getting the secret message shared between the admin and user “subject=test&body=testmessage”:

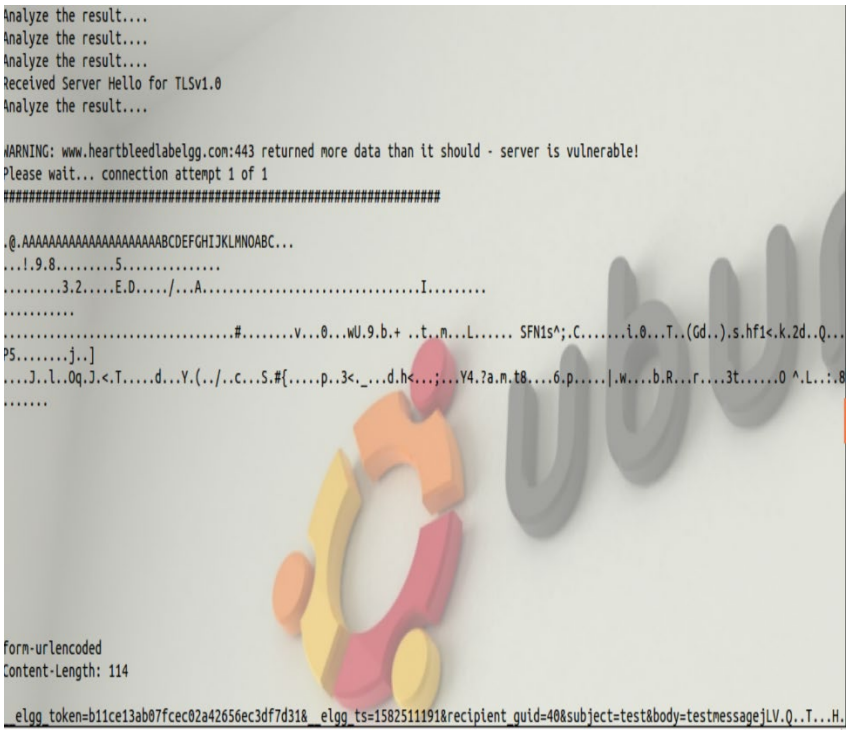


*Figure 3 subject and body*

## Task 3.2

**As the length variable decreases, what kind of difference can you observe?**

The payload returned gets lesser as the length variable decreases. This is shown in the screenshots taken with different length variables used, 100 and 500. The reason for this is that there is less space for the server to return data back to the attacker since the specified payload length is smaller.

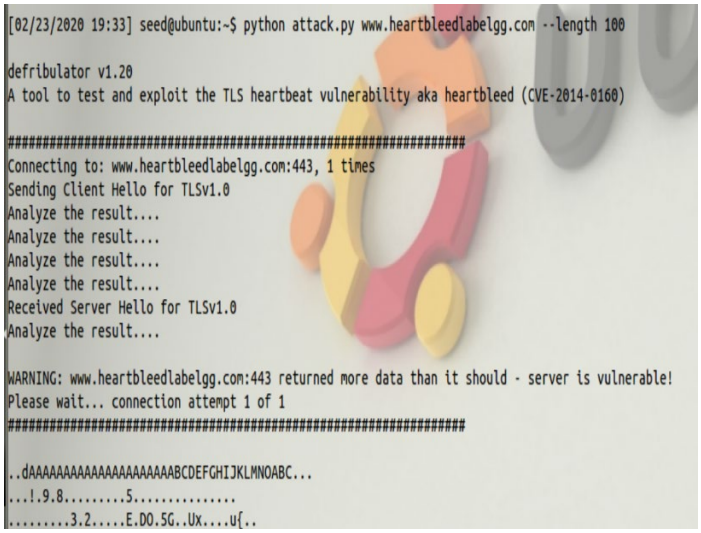


*Figure 4 100*


*Figure 5 500*

**As the length variable decreases, there is a boundary value for the input length variable.**
**At or below that boundary, the Heartbeat query will receive a response packet without attaching any extra data (which means the request is benign). Please find that boundary length.**

The boundary length is 23. This is show in the screenshots below taken of boundary length and boundary length + 1.


*Figure 6 failed*

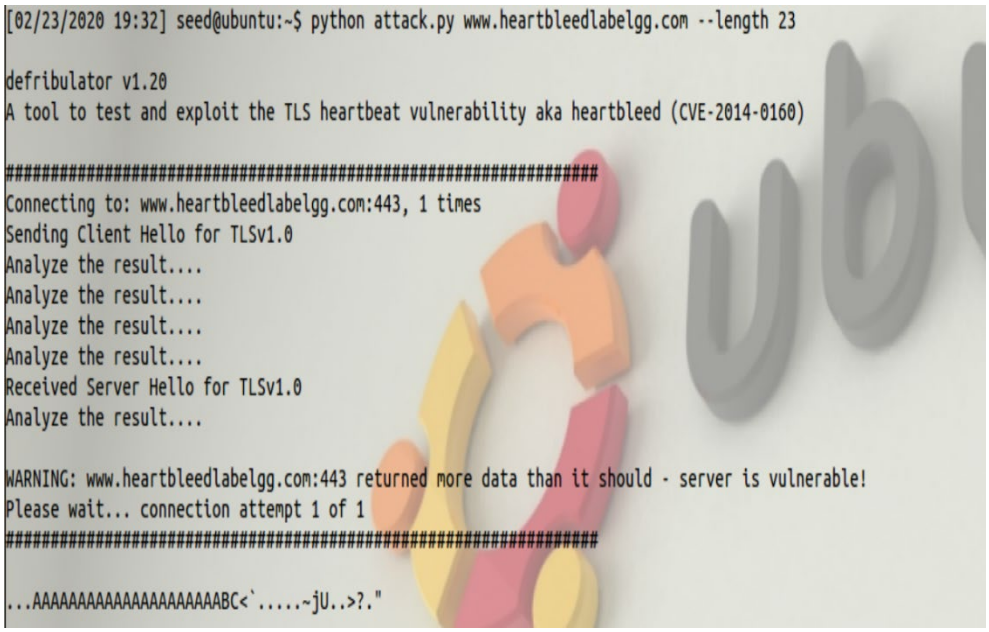


*Figure 7 successful*

**Try your attack again after you have updated the OpenSSL library. Please describe your observations.**

The attack no longer works. The script does not detect that the service is vulnerables


**The objective of this task is to figure out how to fix the Heartbleed bug in the source code.**

The following code snippet causes the vulnerability.

```
// copy payload
memcpy(bp, pl, payload); /* pl is the pointer which
                          * points to the beginning
                        * of the payload content */
```

The problem is that the payload variable contains 16 bits worth of information which is then copied to the buffer. The memcpy function does not check the length of payload passed to it. The value of "payload" may be much more than what was sent over as a heartbeat message. This causes old data in the memory in the space allocated to the buffer to be sent over to the client. One way to fix it would be to but a bound checking on the size of "pl" and the value of payload to make sure they are the same before allocating memory to "bp".

**Please comment on the following discussions by Alice, Bob, and Eva regarding the fundamental cause of the Heartbleed vulnerability: Alice thinks the fundamental cause is missing the boundary checking during the buffer copy; Bob thinks the cause is missing the user input validation; Eva thinks that we can just delete the length value from the packet to solve everything.**

Alice is right, as adding the boundary checking will not allow attackers to request for malicious payload lengths that can return more information from the server than necessary. Bob is wrong as sanitization can most of the time have a loophole. Eva is wrong as deleting the length value might not allow for transfer of important information from server to client if necessary.