Heartbleed is a bug that some have described as the most serious vulnerability since the beginning of the Internet. OpenSSL — the encryption technology used by millions of websites to encrypt communication and to protect sensitive data such as emails, password or banking information, has a critical flaw that allows cyber criminals to extract data from the system memory.

The prolific use of the OpenSSL library for secure communications means that the impact is widespread, affecting IP phones, email servers, routers and Smart TVs as well as websites. The challenge for IT managers is to determine where the vulnerabilities in their IT ecosystem are and how best to mitigate against them. Now is the time to take stock of security and make best practice improvements to defend against future attacks. This paper explains how the Heartbleed bug works and offers some advice for developing a sound ‘Defence-in-depth’ security strategy.

HEARTBLEED — WHAT’S THE BIG FUSS?

How does the Heartbleed bug work?

For SSL to work, your computer (client) communicates to the server by sending a ‘heartbeat’ informing the server that your computer is still online or active.

The actual message sent in the ‘heartbeat’ can be simplified in human terms to be a conversation that sounds like this:

- Client to Server:
  ‘Server are you still there?’ ‘If so, reply with the word, for example, “FOO” spelt with 3 characters’

- Server to Client:
  ‘FOO’

The required format of the above ‘heartbeat’ conversation asks for acknowledgment from the server by responding with an arbitrary word the client computer has chosen with a description of that word defined as the number of characters it contains. This format creates an interesting response if the conversation was to be manipulated like so:

- Client to Server:
  ‘Server are you still there?’ ‘If so, reply with the word “FOO” spelt with 500 characters’

- Server to Client:
  ‘FOO..USERIDXXXPASSWORDXXXPRIVATEKEYXXXXETC’

In the above malformed ‘heartbeat’ conversation, the client is asking the server for acknowledgment with the response being the word ‘FOO’ this time containing 500 characters. The server’s reaction to such a malicious request is to incredibly respond with the word ‘FOO’ plus whatever information was residing in its memory bank (up to 64Kb/heartbeat response) in order to fulfil the 500-character definition! What information is stored in its memory bank at the time can be anything from username and passwords, private keys or credit card information! These heartbeat messages are now bleeding sensitive data and will continue to bleed some more with every malicious heartbeat request as more chunks of memory is being accessed — hence the name heartbleed. This, in technical parlance, is what is called a buffer over-read vulnerability.

What systems are vulnerable?

The serious nature of this bug lends itself to the prolific use of the OpenSSL library in providing secure communications for everything ranging from IP phones, email servers, routers, Smart TV sets and not just websites. With such an unprecedented wide impact area, the challenge facing all IT managers is based on answering the following two questions:

- Where am I vulnerable?
- How can I remediate against this vulnerability?

To assist with answering the first question a number of sites have popped up on the Internet offering an online test tool that does a quick check of any queried website to determine the presence of the vulnerability in its OpenSSL implementation.

The vulnerable OpenSSL Library is version 1.0.1, which was released on March 14, 2012 with the heartbeat code enabled by default. Despite its default enablement there is no way to manually turn it off (without recompiling the code). Another scary fact is realising that this vulnerability was out there in the wild for two years before its eventual discovery in April 2014. From discovery came the inevitable patch released as version 1.0.1g shortly thereafter.
Steps for minimising exposure

For IT managers, patching affected infrastructure can take time depending on the scope and change management process. To minimise the risk of exposure once the identified systems have been patched requires the following remediation steps:

**Step 1:** Request revocation of the current SSL certificates currently being used.

**Step 2:** Regenerate any private keys currently being used.

**Step 3:** Request and replace all current SSL certificates.

**Step 4:** Change all passwords (i.e. static passwords) in your administrative domain.

Due to the stealthy nature inherent in this vulnerability and given the fact that it is not detectable in logs, the question often asked is whether one can in fact detect and defend against such an exploit. Intrusion detection and prevention system (IDS/IIPS) rules have been developed by vendors for the Heartbleed vulnerability. Due to encryption, differentiating between legitimate use and attack cannot be based on the content of the request, but the attack can be detected by comparing the size of the request against the size of the reply. The size of the reply will be abnormally larger than the request as explained earlier.

Are attacks like this impossible to defend against?

At a macro level this vulnerability has triggered many important discussions, among them, the security implications this bug poses on zero day vulnerabilities or any other unknown attack vectors that a security professional must keep an eye out for.

Most security experts would agree that there is no single way to detect a zero day vulnerability-based attack against an institution. However, attacks are not composed of a single action but rather a collection of activities with multiple stages to form a complex plan aimed at achieving a certain goal for the attacker. So whilst the zero day vulnerability may go unnoticed, the different stages of the attack should trigger alarms in different parts of the institution.

Building a sound security strategy – ‘Defence-in-depth’

From an institution-wide standpoint, even if the Heartbleed vulnerability was used – other controls should have helped to detect an attack. This is the principle of the ‘Defence-in-depth’ security strategy. It might be an end point security product or an IDS/IIPS deployment working together in a mutually supportive defensive system guarding against single-points of failures in any specific technology or protection method for the institution.

A defence-in-depth strategy would also accommodate the security challenges of cloud-based solutions, a prime concern for the education sector. For example, for the enterprise Software as a Service (SaaS) portfolio, most institutions intrinsically trust their providers and rely solely on the authentication chain to secure their data. This false sense of security can be most vulnerable to the Heartbleed bug if users’ passwords are compromised. Then attackers would be able to gain access to an institution's data without going through any other security defence lines and with no visibility or mitigation capabilities.

Conclusion

The Heartbleed bug provides a timely reminder to IT Managers of the fragile defences their institutions have today for protecting enterprise data in cloud applications in particular. This can be changed by building defence-in-depth into the institutions cloud strategy. Customised solutions will assist institutions to improve their security posture on campus and in the cloud so that the ‘next’ Heartbleed bug is not as big a fuss as the original.

About the Author

IuPati Tumaalii is part of AARNet's Enterprise Services team of independent consulting engineers that supply IT consultancy services exclusively for AARNet customers. IuPati specialises in enterprise and wide area networking for campus network, campus edge and data centre optimisation, resiliency and security.

IuPati Tumaalii
Consulting Network Engineer
AARNet Pty Ltd
iupati.tumaalii@aarnet.edu.au

*Opinions expressed in this white paper may change over time.*