**OpenSSL:**

A comprehensive encryption tool called OpenSSL provides an open-source implementation for the TLS protocol. Users can create CSRs (Certificate Signing Requests), generate private keys, and activate SSL certificates, among other SSL-related duties.

**Uses:**

Demand for SSL credentials is currently strong. Since Google started its "HTTPS Everywhere" campaign, the security environment has undergone a significant shift.

To encourage the installation of digital certificates, they first improved SEO; later, Chrome effectively made HTTPS required for all users. Popular platforms like browsers such as Firefox and Chrome will flag your site as not secure if you don't use an SSL certificate. With OpenSSL, you can set up the SSL files on your computer and register for a digital certificate (Generate the Certificate Signing Request). You can perform a variety of verifications and transform your certificate into different SSL forms.

Due to the fact that not all sites offer user-friendly online interfaces for SSL administration, OpenSSL may be the only option for some platforms to import and setup your certificate. The Open SSL instructions used on Linux systems are also used on Windows.

The TLS and SSL encryption algorithms are implemented in OpenSSL using open-source software. Basic cryptographic functions are implemented in the core package, which is written in the C computer language and offers a number of useful features. A number of programming languages have wrappers that make it possible to use the OpenSSL library.

**Difference between SSL and OpenSSL:**

Secure Sockets Layer, also known as SSL, is a deprecated cryptographic system that protects network conversations between two computer apps. When it comes to discuss SSL certificates, what we're referring to is TLS (Transport Layer Security) certificates since TLS is SSL's replacement. Alternatively, OpenSSL is a command-line tool for managing the creation, distribution, and authentication of SSL/TLS certificates.

**Some common commands used in OpenSSL:**

OpenSSL version can be checked by the following command:

* OpenSSL version -a

There are a few necessary stages to installing an SSL certificate on a website, and they apply to all servers and email clients. When someone does not have an online interface or wants to speed up the entire process, OpenSSL is particularly useful. To create the Certificate Signing Request as well as the secret key, combine files, one must check the details of the certificate, and resolve any possible problems OpenSSL commands.

The CSR code can be generated using OpenSSL. A piece of text called a CSR contains information about the site and business. The Certification Authority must approve the CSR before someone can proceed. A person can also use an online CSR generator application to generate CSR. A secret key must be provided with the certificate request in order to generate the public key. It's advised to always produce a fresh private key at any time one wants to establish a CSR, even though they may utilize an existing key. It's time to make the CSR after the generation of the private key. It shall be in PEM form and contain information about the company in addition to the public key that was created using the private key provided.

* openssl req -new -key yourdomain.key -out yourdomain.csr

One must select the key algorithm, key size, and possible password in order to create the private key. Although RSA is the common secret method, ECDSA can also be used in some circumstances. When utilizing the RSA key technique, choose a key size of 2048 bits, whereas when employing the ECDSA key technique, choose a key size of 256 bits. Any key size below 2048 is not safe, and key sizes above 2048 may cause speed issues.

Finally, choose whether or not the secret key requires a passphrase. Some sites will not take passphrases and private keys.

* openssl genrsa -out yourdomain.key 2048

The data of the key can be viewed by the following:

* cat yourdomain.key

To make sure only the correct information is being submitted, we can verify the CSR information by the following command:

* openssl req -text -in yourdomain.csr -noout –verify

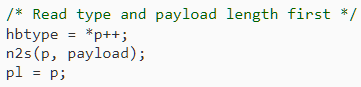
**Major Vulnerabilities involving OpenSSL:**

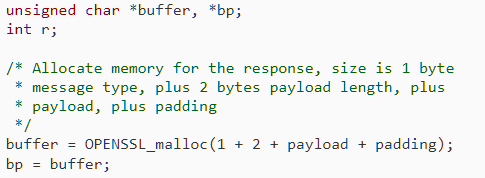
1. **Heartbleed**

Through improper memory management in the TLS heartbeat extension, a flaw in OpenSSL could enable a remote attacker to reveal private information, potentially including user authentication passwords and secret keys. The TLS/DTLS pulse feature is implemented incorrectly in OpenSSL version 1.0.1. Using the vulnerable OpenSSL library, this vulnerability enables an attacker to obtain confidential memory of a program in segments of 64k at a time. You should be aware that a hacker can frequently exploit the flaw to recover as many 64k portions of memory as needed to obtain the desired secrets. This flaw allows for the retrieval of the following private data:

* passwords and private keys used by susceptible sites
* Memory addresses and information used by susceptible services to access confidential data and get around attack mitigations

In order to establish a safe link using TLS, the exploit begins by transmitting a handshake to the website





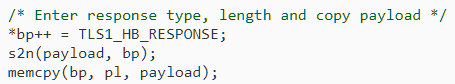
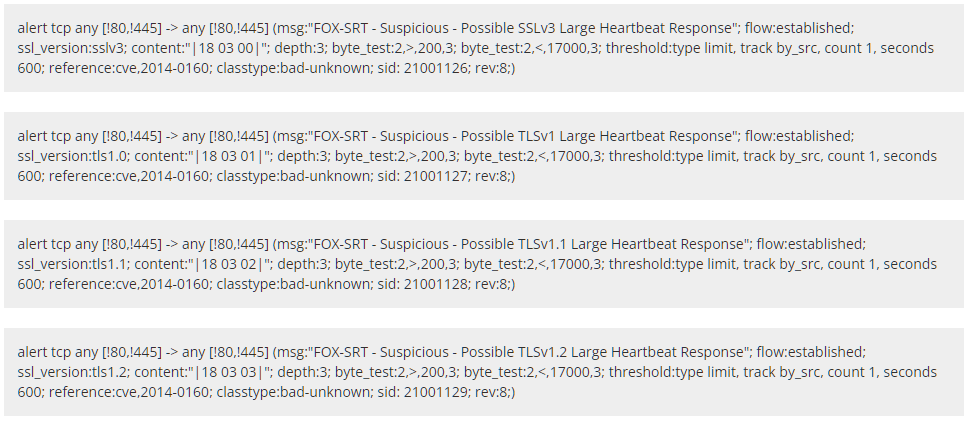


Fig 1. Code snippet from Sean Cassidy’s Heartbleed diagnosis.

If the person making the request wasn't truthful when she claimed to have provided the data bytes? What if pl actually only takes up one byte? Then, within the same operation, the read command from memcpy will read any memory that was close to the SSLv3 record. And it seems like there is a plethora of things close by.

The actions listed below are the recommended measures for every SSL service that is at risk.

* OpenSSL should be updated to 1.0.1g

Fig. 2. Heartbleed signatures

* Request the existing SSL certificate's cancellation.
* Make a new copy of your secret key.
* Ask for and change the SSL certificate

By looking at the network data, it is feasible to identify effective exploitation of this vulnerability. To identify successful abuse of the "heartbleed bug" researchers have created two groups of Snort signatures (Fig. 3). These can be used to detect any bleeds. The vulnerability is still active in few webpages and can lead to issues from to time. However, there are many defensive methods developed against this.

1. **CCS Injection Vulnerability**

The well-known OpenSSL encryption software library has a severe flaw called the CCS Injection flaw. The SSL/TLS encryption system is implemented by OpenSSL and is used to safeguard the confidentiality of Internet contact. Numerous websites as well as programs like instant messaging, emails, and VPNs use OpenSSL. A skilled adversary can compel the implementation of weak keying material in OpenSSL SSL and TLS encryption clients as well as servers in some versions of the protocol. A man-in-the-middle attack that allows the perpetrator to decrypt and alter data between the targeted client and server can take advantage of this.

According to recent reports, this vulnerability has been around for at least ten years, but it was only recently discovered. Given the large number of companies that have deployed servers running OpenSSL, businesses that were concerned about or negatively impacted by Heartbleed bug might wish to take careful note about this susceptibility as well. According to early accounts, attacks leave almost no evidence, making it very challenging, perhaps even unattainable, to identify their history. As a result, even if a company isn't susceptible right now, it might have been affected in the past. If this is the case, the company needs to take quick action to fix the problem. Only if the problem affects both the server and the client can it be abused. There is no danger of harm in a situation that just one of both of them is weak.

Masashi Kikuchi for identifying and studying this problem. On May 1, 2014, this problem was brought to OpenSSL's attention. On the basis of an initial patch from Masashi Kikuchi, Stephen Henson of the OpenSSL core team partially created the remedy. Only if the server is also prone to CVE-2014-0224 will services operating over SSL from PAN-OS / Panorama to third-party machines (such as syslog server, directory services server) be susceptible to a potential MITM attack. Verify that the machine hosting the third-party application is not using an outdated version of OpenSSL.

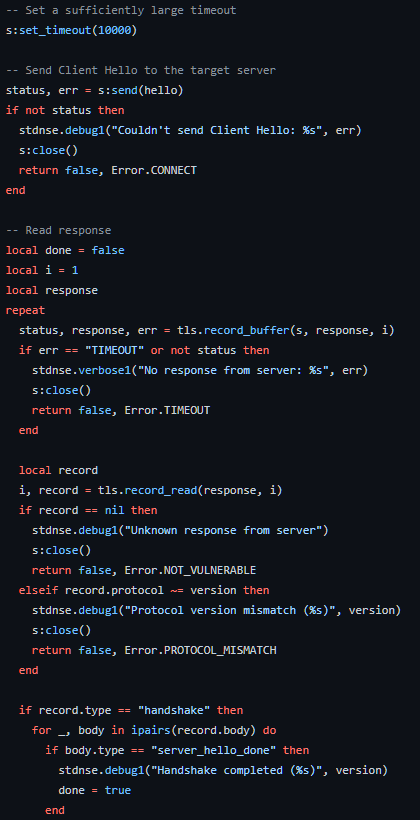


Fig. 3. – Implementation of CCS Injection by Github User: r00t-3xp10it

1. **DROWN**

DROWN or Decrypting RSA with Obsolete and Weakened Encryption corresponds to a cross-protocol security flaw that targets sites that support new SSLv3/TLS protocol packages by exploiting their support for the outdated, insecure SSL v2 protocol to launch assaults on connections that would normally be protected using more recent protocols. It makes it simpler for remote adversaries to decode TLS ciphertext data by using a Bleichenbacher RSA padding oracle because it requires a server to deliver a ServerVerify packet prior to verifying that an end user holds certain plaintext RSA data.

When SSLv2 was in use, the master secret was immediately encrypted using RSA, and when 40-bit export ciphersuites were used, just 40 bits of the master secret were encrypted, leaving the remaining 88 bits in unencrypted. The 48-byte SSLv3/TLS obfuscated RSA cipher is "trimmed" to 40-bit portions and is subsequently utilized for the SSLv2 ClientMasterKey message that the website's server considers as the 40-bit component of the SSLv2 master secret in which the rest of the 88 bits allow for anything provided by the user as unencrypted. It is possible to use the ServerVerify message to serve as oracle by employing brute force the 40-bit encryption.

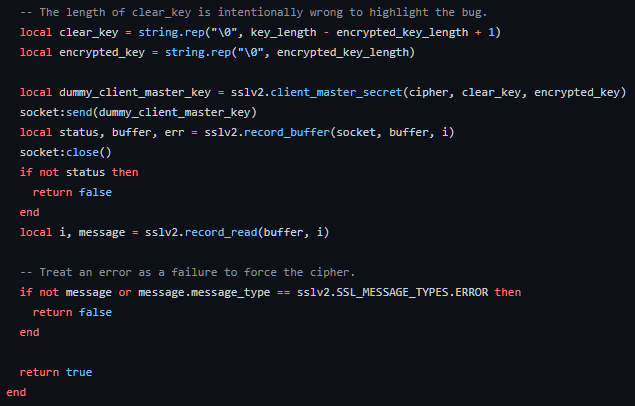


Fig. 4. – Implementation of DROWN vulnerability by Github User: bonsaiviking

* It makes it possible for man-in-the-middle attackers to bypass network security, eavesdrop, relay, and perhaps even change user and device messages.
* A potential attacker could access and steal confidential data.
* Attackers may pretend to be legitimate websites, steal their traffic, or modify their content.
* Data in ciphertext form could be decrypted by attackers.

Server administrators must make absolutely certain that their secret keys are never used with server-side software that supports SSLv2 communications in order to defend against DROWN. This applies to all applications that accept SSL/TLS, such as web servers, SMTP servers, IMAP and POP servers. In order to address the issue, the OpenSSL group has published a security warning and a number of updates that disable support for antiquated protocols and ciphers. However, the patched servers are still susceptible if the certificate of the server is utilized on other platforms that enable SSLv2. The vulnerability should be fixed as quickly as feasible by site administrators, according to numerous reports.

**References:**

1. S. (n.d.). *GitHub - secretnonempty/CVE-2014-0224*. GitHub. https://github.com/secretnonempty/CVE-2014-0224
2. *CVE-2014-0224 OpenSSL Man-in-the-middle vulnerability*. (2014, June 9). CVE-2014-0224 OpenSSL Man-in-the-middle Vulnerability. https://security.paloaltonetworks.com/CVE-2014-0224
3. *OpenSSL CCS Injection Vulnerability (CVE-2014-0224) Alert - Red Hat Customer Portal*. (2016, May 20). Red Hat Customer Portal. https://access.redhat.com/articles/904433
4. *DCX*. (n.d.). DCX. https://success.trendmicro.com/dcx/s/solution/1103813-trend-micro-products-and-the-ccs-injection-vulnerability--cve-2014-0224-openssl-vulnerability?language=en\_US&sfdcIFrameOrigin=null#:~:text=What%20is%20the%20CCS%20Injection,the%20privacy%20of%20Internet%20communication
5. Langley, A. (n.d.). *ImperialViolet - Early ChangeCipherSpec Attack*. ImperialViolet - Early ChangeCipherSpec Attack. https://www.imperialviolet.org/2014/06/05/earlyccs.html
6. Inc., O. F. (n.d.). */index.html*. /index.html. https://www.openssl.org/
7. Inc., O. F. (n.d.). */news/vulnerabilities.html*. /News/vulnerabilities.html. https://www.openssl.org/news/vulnerabilities.html
8. *OpenSSL - Wikipedia*. (1998, January 1). OpenSSL - Wikipedia. https://en.wikipedia.org/wiki/OpenSSL
9. Fruhlinger, J. (n.d.). *The Heartbleed bug: How a flaw in OpenSSL caused a security crisis*. CSO Online. https://www.csoonline.com/article/3223203/the-heartbleed-bug-how-a-flaw-in-openssl-caused-a-security-crisis.html
10. *The Heartbleed Bug, explained*. (2014, June 19). Vox. https://www.vox.com/2014/6/19/18076318/heartbleed
11. *OpenSSL “Heartbleed” vulnerability (CVE-2014-0160) | CISA*. (2016, October 5). Cybersecurity and Infrastructure Security Agency CISA. https://www.cisa.gov/news-events/alerts/2014/04/08/openssl-heartbleed-vulnerability-cve-2014-0160
12. R. X. (n.d.). *RC-exploiter/ssl-ccs-injection.nse at master · r00t-3xp10it/RC-exploiter*. GitHub. https://github.com/r00t-3xp10it/RC-exploiter
13. *sean cassidy : Diagnosis of the OpenSSL Heartbleed Bug*. (2014, April 7). Sean Cassidy : Diagnosis of the OpenSSL Heartbleed Bug. https://www.seancassidy.me/diagnosis-of-the-openssl-heartbleed-bug.html
14. *SSLv2-Drown Vulnerability in OpenSSL | Trend Micro Help Center*. (n.d.). SSLv2-Drown Vulnerability in OpenSSL | Trend Micro Help Center. https://helpcenter.trendmicro.com/en-us/article/tmka-19804