Attacks on TLS [Graded-Optional, 4/9-14/9]

Wild at Heart: OpenSSL's Heartbleed

⇥ BEAST Attack

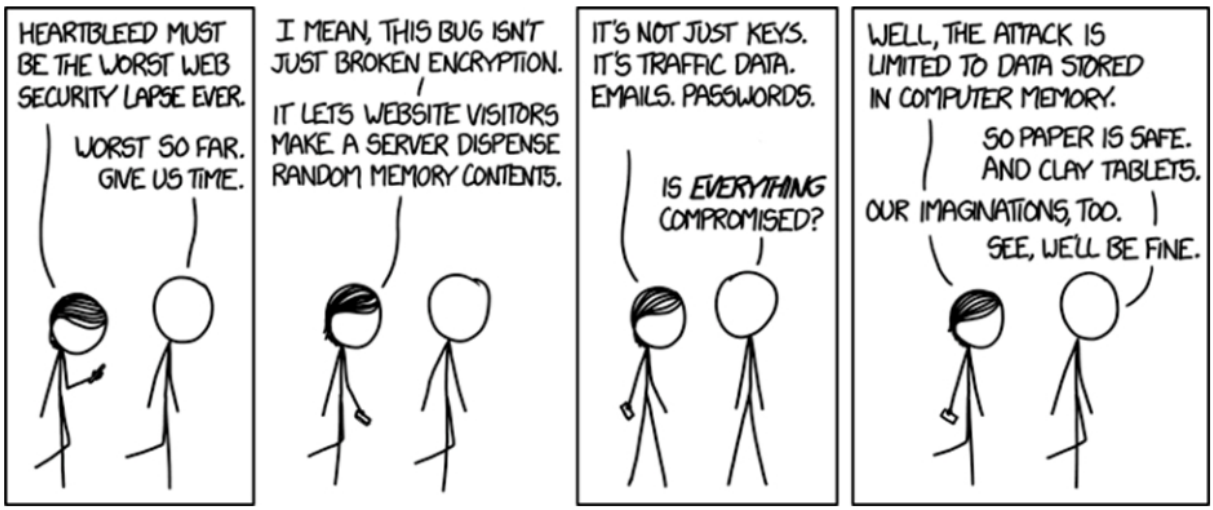
Logjam attack ⇤

Display replies flat, with oldest first

Settings

The cut-off date for posting to this forum is reached so you can no longer post to it.

Wild at Heart: OpenSSL's Heartbleed
by رسول کامکار - Wednesday, 8 Azar 1402, 9:03 PM



What is OpenSSL

OpenSSL is an all-around cryptography library that offers an open-source application of the TLS protocol. It allows users to perform various SSL-related tasks, including CSR (Certificate Signing Request) and private keys generation, and SSL certificate installation.

OpenSSL is crucial to a website's security and success, and is used all around the internet, so any vulnerability in this library could threaten everything.

What is Heartbleed

The Heartbleed Bug is a serious vulnerability (CVE-2014-0160) 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).

It got its horrific name because it is a flaw in OpenSSL's implementation of the Heartbeat Extension for the TLS and DTLS protocols. It's also categorized as Buffer Over-read. The worst part is that no traces are left during the attack, and it's executed without any credentials.

Heartbleed birth, discovery and fix dates

Based on CVE-2014-0160, Heartbleed was discovered and publicly announced as an official common vulnerability in 2014.

The OpenSSL version that introduced this bug is believed to be 1.0.1, which was unleashed into the wild in 14th of March 2012.

OpenSSL 1.0.1g released on 7th of April 2014 fixes the bug.

In general, OpenSSL 1.0.1 through 1.0.1f (inclusive) are vulnerable.

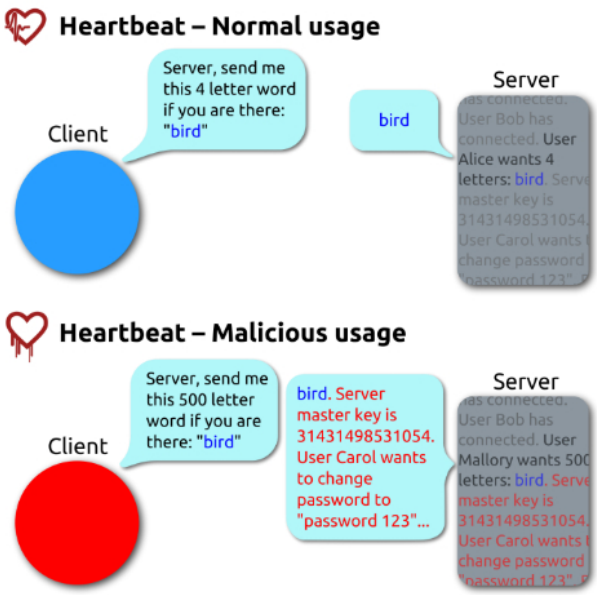
The 31-year-old German developer Robin Seggelmann added the faulty Heartbeat feature — lacking a validation process of a variable

containing length — to an experimental version of OpenSSL in late 2011. Then his features' submission passed through OpenSSL review, but the developer there also didn't catch the flawed code.

Where does it bleed from?

This serious flaw is a missing bounds check before a memcpy() call that uses non-sanitized user input as the length parameter. An attacker can trick OpenSSL into allocating a 64KB buffer, copy more bytes than is necessary into the buffer, send that buffer back, and thus leak the contents of the victim's memory, 64KB at a time.

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.



How sensitive the leaked data is?

This vulnerability allows an attacker to extract memory contents from the webserver through the vulnerability in the heartbeat. As a result, an attacker may be able to access sensitive information such as the private keys used for SSL/TLS.

These are the crown jewels, the encryption keys themselves. Leaked secret keys allow the attacker to decrypt any past and future traffic to the protected services and to impersonate the service at will. Any protection given by the encryption and the signatures in the X.509 certificates can be bypassed.

Not only that, there are for example the user credentials (user names and passwords) used in the vulnerable services. All session keys and session cookies should be invalidated and considered compromised.

Even worse! the actual content handled by the vulnerable services is also compromised. It may be personal or financial details, private communication such as emails or instant messages, documents or anything seen worth protecting by encryption.

How many hearts?

As of April 2014, the open-source web servers Apache and Nginx, which use OpenSSL, made up two-thirds of the market share of all active sites online. This gives an idea of the massive scope of the Heartbleed vulnerability. It affected digital companies and public bodies like the Canada Revenue Agency.

Since attacks leave no traces in the logs, intrusion detection and estimation of the actual exploitation attempts and successes of the Heartbleed bug are difficult.

Were Intelligence Agencies and Blackhats Using Heartbleed?

we don't know how many attacks leveraged this bug, and how devastating the impact's been, but there are stories and reports stating intelligence agencies used this bug in November 2013, I'll leave the link to the article in case you're interested:

https://www.eff.org/deeplinks/2014/04/wild-heart-were-intelligence-agencies-using-heartbleed-november-2013

references:

https://heartbleed.com/

https://www.ssldragon.com/blog/what-is-openssl/

https://owasp.org/www-community/vulnerabilities/Heartbleed_Bug

https://xkcd.com/1353/

https://crashtest-security.com/prevent-heartbleed/

https://www.malwarebytes.com/blog/news/2019/09/everything-you-need-to-know-about-the-heartbleed-vulnerability

Sum of ratings: 50 (1) Permalink

در پاسخ به: **Wild at Heart: OpenSSL's Heartbleed**
by مهسا ساحلی - Saturday, 11 Azar 1402, 6:44 PM

The Heartbleed vulnerability was a critical flaw in OpenSSL's implementation of the TLS/SSL heartbeat extension, which potentially allowed attackers to access sensitive information from the memory of affected servers. The fix for the Heartbleed attack involved applying a patch to OpenSSL, specifically in version 1.0.1g.

The Fix (Patch in OpenSSL 1.0.1g):

The patch involved implementing a bounds check using the correct record length in the SSL3 structure. The code snippet below illustrates part of the fix:

```
hbtype = *p++;
n2s(p, payload);
if (1 + 2 + payload + 16 > s->s3->rrec.length)
  return 0; /* silently discard per RFC 6520 sec. 4 */
pl = p;
```

This code snippet checks the bounds to ensure that the length of the HeartbeatMessage is within the expected range, preventing the vulnerability from being exploited.

Defending Against the Vulnerability:

For Website Owners:

1. Verify and Upgrade OpenSSL:

Verify if your OpenSSL version is vulnerable and upgrade to OpenSSL 1.0.1g or later.

How common are the vulnerable OpenSSL versions?

The vulnerable versions have been out there for over two years now and they have been rapidly adopted by modern operating systems. A major contributing factor has been that TLS versions 1.1 and 1.2 came available with the first vulnerable OpenSSL version (1.0.1) and security community has been pushing the TLS 1.2 due to earlier attacks against TLS (such as the BEAST).

How can OpenSSL be fixed?

Even though the actual code fix may appear trivial, OpenSSL team is the expert in fixing it properly so fixed version 1.0.1g or newer should be used. If this is not possible software developers can recompile OpenSSL with the handshake removed from the code by compile time option -DOPENSSL_NO_HEARTBEATS.

2. Reissue Security Certificates:

Due to the potential compromise of private keys, reissue security certificates for SSL/TLS.

How revocation and reissuing of certificates works in practice?

If you are a service provider you have signed your certificates with a Certificate Authority (CA). You need to check your CA how compromised keys can be revoked and new certificate reissued for the new keys. Some CAs do this for free, some may take a fee.

3. Implement Perfect Forward Secrecy (PFS):

Enhance security by implementing Perfect Forward Secrecy, which uses per-session random keys for additional protection against various attacks.

4. Change Passwords:

If OpenSSL was used to protect login and password information, assume a potential breach, inform users, and prompt them to change their passwords. Implement a system for one-time forced password changes after login.

For Users:

1. Change Passwords:

If you have accounts on websites that may have been affected, consider changing your passwords as a precaution.

2. Proactive Measures:

Users who entered passwords in the last few days should proactively change them. However, according to Errata Security, the urgency varies depending on recent password entry activity.

In summary, the Heartbleed fix involved applying a patch to OpenSSL, and website owners needed to take additional steps such as certificate reissuing, implementing PFS, and prompting password changes. Users were advised to change passwords as a precautionary measure, especially if they had entered passwords recently.

References:

owasp.org

heartbleed.com

Permalink Show parent

⇥ BEAST Attack

Logjam attack ⇤