*Attacking SSL/TLS Implementations*

Shawn Philip Babu  
Information Systems Security  
Concordia UniversityMontreal, Canada  
shawn0111babu@gmail.com  
  
  
Sai Sandeep Vendra  
Information Systems Security  
Concordia UniversityMontreal, Canada  
sandeepvendra1404@gmail.com line 1: 3rd Given Name Surname  
line 2: *dept. name of organization (of Affiliation)*  
line 3: *name of organization (of Affiliation)*line 4: City, Country  
line 5: email address or ORCID  
  
line 1: 4th Given Name Surname  
line 2: *dept. name of organization (of Affiliation)*  
line 3: *name of organization (of Affiliation)*line 4: City, Country  
line 5: email address or ORCID

line 1: 5th Given Name Surname  
line 2: *dept. name of organization (of Affiliation)*  
line 3: *name of organization (of Affiliation)*line 4: City, Country  
line 5: email address or ORCID

line 1: 6th Given Name Surname  
line 2: *dept. name of organization (of Affiliation)*  
line 3: *name of organization (of Affiliation)*line 4: City, Country  
line 5: email address or ORCID

*Abstract*—SSL stands for Secure Sockets Layer and, it's the standard technology for keeping an internet connection secure and safeguarding any sensitive data that is being sent between two systems, preventing criminals from reading and modifying any information transferred, including potential personal details. TLS (Transport Layer Security) is just an updated, more secure, version of SSL. We still refer to security certificates as SSL because it is a more commonly used term, but when you are [buying SSL](https://www.websecurity.digicert.com/ssl-certificate?inid=infoctr_buylink_sslhome) from DigiCert you are actually buying the most up to date TLS certificates with the option of [ECC, RSA or DSA encryption](https://www.websecurity.digicert.com/security-topics/how-ssl-works). However, there are undeniable differences between the libraries that implement SSL/TLS protocol and vulnerabilities in these libraries. Hence, the two main questions asked are: what’s the difference between TLS vs SSL? And is it something we need to worry about? In this report, we summarize some of the limitations by considering implementations of each along with review of past protocol-based and software-based vulnerabilities.

# Introduction

JEMY & EKASMEET

(Times new roman 10)

# Limitation of SSL/TLS Implementations

SHAWN & HARSHITHA

(Times new roman 10)

# How SSL/TLS secure data ?

VENKY

(Times new roman 10)

# Comparisions of DIfferent versions of SSL/TLS

TLS (Transport Layer Security) and OpenSSL are both important cryptographic protocols used to secure communication on the internet. OpenSSL is a widely used open-source implementation of SSL/TLS protocols. Here is a comparison of different versions of TLS and OpenSSL:

TLS 1.0 - This version of TLS is now considered insecure due to several vulnerabilities, including POODLE and BEAST attacks. OpenSSL 1.0.1g or later supports TLS 1.0.

TLS 1.1 - This version of TLS addresses some of the vulnerabilities present in TLS 1.0, but it is also considered insecure due to certain vulnerabilities such as Lucky13. OpenSSL 1.0.1g or later supports TLS 1.1.

TLS 1.2 - This version of TLS is currently the most widely used and is considered secure. It has improved security features compared to TLS 1.1, such as stronger cipher suites, and support for authenticated encryption with associated data (AEAD). OpenSSL 1.0.1g or later supports TLS 1.2.

TLS 1.3 - This version of TLS is the latest and most secure version of TLS. It provides better security and performance compared to TLS 1.2, including faster handshakes and improved forward secrecy. OpenSSL 1.1.1 or later supports TLS 1.3.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Version** | **TLS 1.0** | **TLS 1.1** | **TLS 1.2** | **TLS 1.3** | **OpenSSL 0.9.8** | **OpenSSL 1.0.1** | **OpenSSL 1.1.1** |
| Release Date | 1999 | 2006 | 2008 | 2018 | 2005 | 2012 | 2018 |
| Security | Weak | Medium | Strong | Very Strong | Weak | Medium | Strong |
| Cipher Suites | Limited | Expanded | Expanded | Expanded | Limited | Expanded | Expanded |
| Handshake | Slow | Faster | Faster | Fastest | Slow | Faster | Fastest |
| Certificate Handling | Limited | Expanded | Expanded | Expanded | Limited | Expanded | Expanded |
| Support | Widely supported | Widely supported | Widely supported | Limited support | Widely supported | Limited support | Widely supported |

OpenSSL also has different versions, and the latest version is OpenSSL 3.0.0.

OpenSSL 1.0.2 - This version is no longer supported and has reached its end of life. It supports up to TLS 1.2.

TABLE

OpenSSL 1.1.0 - This version introduced support for TLS 1.3 and Elliptic Curve Cryptography (ECC).

OpenSSL 1.1.1 - This version introduced several improvements, including support for TLS 1.3, ChaCha20-Poly1305 cipher suites, and Ed25519 and Ed448 elliptic curves.

Some of the major differences are listed below based on the characteristics

***Cipher suites***

SSL protocol offers support for Fortezza cipher suite. TLS does not offer support. TLS follows a better standardization process that makes defining of new cipher suites easier like RC4, Triple DES, AES, IDEA, etc.

***Alert messages***

SSL has the “No certificate” alert message. TLS protocol removes the alert message and replaces it with several other alert messages.

***Record Protocol***

SSL uses Message Authentication Code (MAC) after encrypting each message while TLS on the other hand uses HMAC — a hash-based message authentication code after each message encryption.

***Handshake process***

In SSL, the hash calculation also comprises the master secret and pad while in TLS, the hashes are calculated over handshake message.

***Message Authentication***

SSL message authentication adjoins the key details and application data in ad-hoc way while TLS version relies on HMAC Hash-based Message Authentication Code.

# Vulnerabilities in SSL/TLS-Protocol based

VANI

(Times new roman 10)

**CRIME(CVE-2012-4929):**

Compression Ratio Info-leak Made Easy (CRIME) is a security exploit against secret web cookies over connections using the HTTPS and SPDY protocols that also use data compression. It enables an attacker to perform session hijacking on an authenticated web session, enabling the start of additional attacks, when used to retrieve the content of private authentication cookies. Although CRIME is a client-side attack, the client can be protected by the server by asking it not to use feature combinations that can be exploited. The drawback of CRIME is Deflate compression. This alert is issued if the server accepts Deflate compression.

**Working:**

Cybercriminals can take advantage of a flaw in the SSL/TLS protocol and the SPDY protocol's compression method to carry out a CRIME attack by decrypting the HTTPS cookies that a website has established. This can then compel a user to view a malicious website while the attack is being carried out by forcing their browser to forward HTTPS requests to that site. Afterward, the attackers control the path for new requests. Cybercriminals have access to information about the client browser's sent ciphertext size. They can then observe how the size of the compressed request payload, which includes the hidden cookie sent by the browser and the harmful content injection, varies. When the compressed content gets smaller, it means that some of the confidential content they want access to have probably been partially mirrored by the injected content. The value of the user's session cookie may be ascertained by observing the fluctuation in length, compression ratio, or varying content.

**FREAK(CVE-2015-0204):**

The term "FREAK vulnerability" describes a flaw in the Secure Sockets Layer (SSL)/Transport Layer Security (TLS) protocols brought on using "export-grade" encryption. The term is an acronym for "Factoring RSA Export Keys."

**Working:**

The FREAK vulnerability enables hackers to intercept HTTPS communications between clients and susceptible websites in order to obtain a website's secret key. As a result, they are now able to decode HTTPS connection logon cookies, passwords, credit card data, and other susceptible data.

Because the client is required to use a "export-grade" key or 512-bit export RSA key, which is much simpler to trace and crack than current encryption standards, private communications are essentially in danger.

How exactly does this work? An attacker can ask for ‘export RSA’ instead of the standard RSA cipher suites through the client’s Hello message. The server then answers with a 512-bit-long export cipher key instead of today’s high-security keys. The response is signed with its long-term key.

The website client takes in the weak ‘export-grade’ key, allowing the Man-in-the-Middle attacker to get the RSA decryption key and use the ‘pre-master secret’ to gain access to the TLS’ master secret’, which is employed for symmetric encryption of messages in the connection. Afterward, the attacker can inject malicious code into the plaintext file — the essence of command injection risks.

**DROWN(CVE-2016-0800):**

HTTPS and other services that depend on SSL and TLS, two crucial cryptographic protocols for Internet security, are vulnerable to the severe flaw known as DROWN. With the help of these protocols, anyone with access to the Internet can use email, instant messaging, and web browsing without worrying about a third party being able to read the communication.

DROWN allows attackers to break the encryption and read or steal sensitive communications, including passwords, credit card numbers, trade secrets, or financial data.

**Working:**

The DROWN attack goes through several stages. First, the attacker must observe and record sessions between the server and the client that use any version of SSL or TLS. For DROWN to work, these sessions must also use RSA cipher suites. Eventually, one of these recorded sessions will be decrypted. At the second stage of the attack, the attacker has captured the usual client/server handshake. They then create multiple connections to the server using the cross-protocol vulnerability. They establish SSLv2 connections to the server, and since the server allows these connections, it is open to exposure. These connections are modified handshake messages that target the RSA ciphertext because unpadded RSA, as used in SSLv2 can be changed. At the second stage of the attack, the attacker has captured the usual client/server handshake. They then create multiple connections to the server using the cross-protocol vulnerability. They establish SSLv2 connections to the server, and since the server allows these connections, it is open to exposure. These connections are modified handshake messages that target the RSA ciphertext because unpadded RSA, as used in SSLv2 can be changed.

# Vulnerabilities in SSL/TLS-Software based

PRANATHI & SARAT

(Times new roman 10)

# Concluding Thoughts

##### References

1. G. Eason, B. Noble, and I. N. Sneddon, “On certain integrals of Lipschitz-Hankel type involving products of Bessel functions,” Phil. Trans. Roy. Soc. London, vol. A247, pp. 529–551, April 1955. *(references)*
2. J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.
3. I. S. Jacobs and C. P. Bean, “Fine particles, thin films and exchange anisotropy,” in Magnetism, vol. III, G. T. Rado and H. Suhl, Eds. New York: Academic, 1963, pp. 271–350.
4. K. Elissa, “Title of paper if known,” unpublished.
5. R. Nicole, “Title of paper with only first word capitalized,” J. Name Stand. Abbrev., in press.
6. Y. Yorozu, M. Hirano, K. Oka, and Y. Tagawa, “Electron spectroscopy studies on magneto-optical media and plastic substrate interface,” IEEE Transl. J. Magn. Japan, vol. 2, pp. 740–741, August 1987 [Digests 9th Annual Conf. Magnetics Japan, p. 301, 1982].
7. M. Young, The Technical Writer’s Handbook. Mill Valley, CA: University Science, 198