On the Robustness of Applications Based on the SSL and TLS Security Protocols

Diana Berbecaru and Antonio Lioy

Politecnico di Torino, Dip. di Automatica e Informatica
Corso Duca degli Abruzzi, No. 24, 10129, Torino (Italy)

Abstract. The SSL and TLS security protocols have been designed and implemented to provide end-to-end data security. This includes data integrity that is the data cannot be modified, replayed or reordered by an attacker without being detected at the receiving endpoint. SSL and TLS however does not provide data delivery integrity, in the sense they do not guarantee that all the sent data will actually arrive at the other side. This is because, for example, SSL/TLS cannot know in advance which is the exact size of the data to be sent over the secured channel. The mosts recent versions (SSLv3 and TLSv1) provide some form of protection against loss of data records by means of sequence numbers and specialized close_notify alert messages to be sent when tearing down the SSL connection. Unfortunately, this is not enough when the last record containing application data together with the closure alert are deleted on purpose, as it happens in the *truncation attacks*. SSLv3/TLSv1 specifications do not indicate what should happen (at the application level) if the close_notify message never arrives at the receiver. Consequently, for applications where it is important to ascertain that the data reached untruncated the other party, it is required to have an additional control at the application level.

In this paper we show (based on practical tests) that some widely-used applications implementing SSLv3 and TLSv1 do not perform further controls on the size of the data to be received, and thus they are vulnerable to truncation attacks. For tests we implemented a specialized MITMSSL tool, used to manipulate the SSL/TLS records exchanged between two communicating parties.

Keywords: security, SSL/TLS, truncation attack, MITM attack.

1 Introduction

The Secure Sockets Layer (SSL) security protocol has been widely implemented and is nowadays the de facto standard(s) for providing secure e-commerce transactions over the Web. SSL was first developed in 1994 by Netscape, when the browser's designers realized that there was no way to guarantee the security of the network through which Web data was transmitted. Consequently, the best way to protect data was to provide encryption and decryption at the connection's endpoints. Since Netscape's designers wanted a unified solution that could

J. Lopez, P. Samarati, and J.L. Ferrer (Eds.): EuroPKI 2007, LNCS 4582, pp. 248–264, 2007.

be used also with non-HTTP applications, SSL was not incorporated into the browser itself but it was located at a level among the reliable TCP transport layer and the application layer above. In theory, application developers would take advantage of the new layer by replacing all the traditional TCP calls (e.g. *send*, *recv*) with the new SSL calls implemented by SSL libraries (e.g. *SSL_write*, *SSL_read* in OpenSSL library [1]). SSL gained remarkable attention and popularity in short time, and it was further improved in version 2, while version 3 [2] was heavily modified in order to fix some serious security drawbacks of SSLv2.

In the same time, as other commercial vendors, such as Microsoft, started to develop their own security protocols operating on top of the transport layer, the Internet Engineering Task Force was asked (by Netscape and Microsoft) to define a standard for an encryption layer protocol as a compromise to stave off incompatible, vendor-specific solutions. The result was the definition of the Transport Layer Security (TLS) protocol, which took into consideration inputs from multiple vendors for its specification. The first version of TLS, TLS 1.0 [3], is based on SSL version 3 (SSLv3), consequently it is sometimes referred as SSL 3.1. TLS 1.1 [4] instead contains mainly improvements to protect the protocol against the cipher block attacks in CBC mode pointed out by Vaudenay [5]. TLS 1.0 and TLS 1.1 are referred further as TLSv1. TLSv1 and SSLv3 have subtle implementation differences [6], the application developers usually notice only very little differences while the end users would not see any difference at all. Nevertheless, TLSv1 and SSLv3 are not interoperable, the most significant difference being that TLSv1 requires certain encryption algorithms that SSLv3 does not. Most (commercial) products support nowadays SSLv2, SSLv3 and TLS 1.0.

All SSL versions, as well as TLSv1, were subject to various security analyses and studies, aimed mainly to identify the weaknesses of the protocol both for what it concerns its design (theoretical attacks), its implementation (practical attacks) as well as its usage, e.g. attacks related to user's behaviour or to the features of the application based on SSLv3/TLSv1 protocols. Examples of such attacks include: the downgrade and the truncation attacks that are due to weaknesses in the protocol specification (in SSLv2 for example) [7]; the Man In The Middle (MITM) attacks that are mostly due to wrong user's behaviour with respect to server's digital certificate used for authentication; the side channel attacks like timing attacks [8] that exploits the information gained from the physical implementation of the SSL protocol running on a system, rather than the theoretical weaknesses in the protocol itself; Vaudenay's attack (further extended in [9]) exploits instead certain characteristics of the application running on top of SSLv3/TLS 1.0, like for example the fact that the same password is going to be sent several times over the SSLv3/TLS 1.0 protected channel.

In this paper we demonstrate (based on practical tests) that a truncation attack can actually be performed against some widely-used applications that support SSLv3/TLS 1.0, and not only against SSLv2, which has already been known to be vulnerable to such an attack. For this purpose we developed a tool named MITMSSL, whose role is to intercept and manipulate (i.e. delete, modify, copy and replace) the SSLv3/TLS 1.0 records exchanged between the client