Network Working Group
Internet-Draft
Intended status: Informational
Expires: January 4, 2013
Y. Pettersen
Opera Software ASA
July 3, 2012

Managing and removing automatic version rollback in TLS Clients
draft-pettersen-tls-version-rollback-removal-00

Abstract

Ever since vendors started deploying TLS 1.0 clients, these clients have had to handle server implementations that do not tolerate the TLS version supported by the client, usually by automatically signaling an older supported version instead. Such version rollbacks represent a potential security hazard, if the older version should become vulnerable to attacks. The same history repeated when TLS Extensions were introduced, as some servers would not negotiate with clients that sent these protocol extensions, forcing clients to reduce protocol functionality in order to maintain interoperability.

This document outlines a procedure to help clients decide when they may use version rollback to maintain interoperability with legacy servers, under what conditions the clients should not allow version rollbacks, such as when the server has indicated support for the TLS Renegotiation Information extension. The intention of this procedure is to limit the use of automatic version rollback to legacy servers and eventually eliminate its use.

Status of this Memo

Copyright Notice

1. Introduction

When vendors of Transport Layer Security (TLS) clients intially developed and released TLS 1.0 [RFC2246] clients, they quickly discovered that not all Secure Sockets Layer (SSL) v3 [RFC6101] servers were willing to accept or complete handshakes with the TLS clients. The reasons for this varied across various server implementations, such as not accepting versions higher than SSL v3, and various errors in the implementation of the handshake, e.g., expecting the RSA Premaster Secret's version field to match the selected version, not the signaled version.

Given the scope of the problem of getting servers fixed, in order to provide a good user experience for their customers, vendors elected instead to restart the connection and signal the older protocol version as the highest supported version in such cases.

This process was repeated when TLS Extensions[RFC6066], TLS 1.1 [RFC4346] and TLS 1.2 [RFC5246] were introduced, as clients had to disable these features to be able to connect with servers that did not tolerate them.

As a consequence, clients are not just vulnerable to a version rollback attack; in the event that a vulnerability in older protocol versions should be discovered, they are intentionally designed to be vulnerable to such attacks by automatically performing a version rollback whenever something goes wrong with the current TLS handshake.

While it would be preferable that clients do not perform version rollbacks, it is presently not practical to forbid it entirely, but there are ways to limit the use of rollbacks, and eventually phase out the usage completely.

This document presents a procedure for selecting when to allow a version rollback and how to implement it, in order to maintain interoperability with legacy servers, as well as when to not allow version rollbacks.

The main factor for deciding not to allow version rollbacks is whether the server supports the TLS Renegotiation Information Extension[RFC5746]. [RFC5746] specifically reminds implementors that servers MUST correctly handle clients that support TLS Extensions and/or new TLS versions than supported by the server. For the most part, server vendors have adhered to this, as (per July 2012) less than 0.14% of servers with Renegotiation Information extension support (70.6%) do not adhere to this requirement, compared to 4.5% among servers that does not support this extension.

1.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

2. Managing Version Rollbacks

When a TLS client initially connects to a TLS server, it exchanges a number messages with the server in order to establish the encrypted connection:

- o Sending the Client Hello, which identifies the client's higest supported version, supported extensions, and cryptographic parameters

- o Receiving the Server Hello, which identifies the server's selected version, supported extensions, cryptographic paramaters

- o Exchange of more messages to negotiate the encryption keys, and other parameters

- o Each sends a Finished message to the other, showing that the negotiation succeeded, after which the secure connection is active

Each step of this negotiation sequence can fail for various reasons, until the Finished messages have been sent and verified. The failures can be indicated with Alert codes or by just shutting down the connection. Frequently, many of these failures are due to incorrect implementation on either end.

This tendency toward implementation issues leading to connection failures have caused most client vendors to adopt a policy of retrying with older versions of the protocol, in case the failure was caused by version-specific problems in the server.

2.1. Version Rollback Sequence

When establishing a TLS connection to a server with unknown capabilities, a client SHOULD use the following sequence, advancing to the next step if the connection attempt fails.

1. If the client supports TLS 1.1 or higher, it SHOULD send a Client Hello indicating this highest version and include all supported extensions. The version of the Record Protocol SHOULD at most be TLS 1.0

2. If step 1 failed, and the server either did not indicate a supported version or this version was TLS 1.0 or below, send a Client Hello indicating TLS 1.0 as the highest version and include all supported extensions. If this fails, the client MAY remove extensions in a separate connection attempt before considering this step to have failed.

3. If step 2 failed, and the server either did not indicate a supported version or this version was SSL v3, send a Client Hello indicating SSL v3 as the highest version, without sending TLS Extensions.

In each step, the client MUST indicate support for the TLS Renegotiation Information Extension, using the TLS_EMPTY_RENEGOTIATION_INFO_SCSV cipher suite value specified by [RFC5746] if TLS Extensions are not sent in the Client Hello.

The client MUST NOT roll back to an older version than the server has indicated, even if the connection handshake failed. That is, if the server indicates support for TLS 1.1, but the connection fails, then the client MUST NOT attempt to connect to the server using TLS 1.0, but allow the connection to fail.

2.2. Version Recovery

Once a connection is established and the client has received the Server Hello, it MUST check the response to determine if the server sends the TLS Renegotiation Information (RI) extension, and then decide how to proceed:

- If the server did not return the RI extension, the client can continue the handshake as normal and MAY continue version rollbacks as described in Section 2.1 if the connection fails.

- If the server did return the RI extension, and the client indicated its highest supported version, with extensions, (first step in Section 2.1) in the Client Hello, the client can continue the handshake as normal but MUST NOT permit version rollbacks, in case the connection fails, but instead allow the connection to fail.

- If the server did return the RI extension, but the client was indicating an older TLS version as its highest supported version, or without TLS Extensions, the client MUST terminate the connection, reestablish it, and send a Client Hello that signals the highest supported version, and includes extensions, and it MUST NOT permit a failure to trigger a new version rollback sequence, but instead end the attempt to establish the connection.

The reason for not allowing version rollbacks if the server supports the RI extension is that such servers MUST accept that clients indicate a higher supported version than they do, and they MUST support or tolerate clients that send TLS Extensions. It must be presumed that, if such a handshake fails, it is because the connection is being subjected to a active version downgrade attack, not that the server has been incorrectly implemented in this respect.

3. IANA Considerations

This document makes no request of IANA.

Note to RFC Editor: this section may be removed on publication as an RFC.

4. Security Considerations

Allowing automatic version rollbacks exposes the TLS connection between the client and server to significant risk if the older version that gets negotiated is vulnerable to an attack that allows the transmitted information to leak.

The use of automatic version rollbacks should be limited to connections to servers that require it for interoperability reasons and be prohibited for any other servers. While it is impractial to discover which servers truly need such consideration, this document specifies the presence of the TLS Renegotiation Information extension as a proxy indication that the server does not require such interoperability considerations.

5. Acknowledgements

6. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2246] Dierks, T. and C. Allen, "The TLS Protocol Version 1.0", RFC 2246, January 1999.

[RFC4346] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.1", RFC 4346, April 2006.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008.

[RFC5746] Rescorla, E., Ray, M., Dispensa, S., and N. Oskov, "Transport Layer Security (TLS) Renegotiation Indication Extension", RFC 5746, February 2010.

[RFC6066] Eastlake, D., "Transport Layer Security (TLS) Extensions: Extension Definitions", RFC 6066, January 2011.

[RFC6101] Freier, A., Karlton, P., and P. Kocher, "The Secure Sockets Layer (SSL) Protocol Version 3.0", RFC 6101, August 2011.

Author's Address

Yngve N. Pettersen
Opera Software ASA

Email: yngve@opera.com