Network Working Group
Internet-Draft
Updates: 2246, 4346, 4347, 5246, 6347
(if approved)
Intended status: Standards Track
Expires: August 16, 2015

B. Moeller
A. Langley
Google

February 12, 2015

TLS Fallback Signaling Cipher Suite Value (SCSV) for Preventing Protocol Downgrade Attacks

draft-ietf-tls-downgrade-scsv-04

Abstract

This document defines a Signaling Cipher Suite Value (SCSV) that prevents protocol downgrade attacks on the Transport Layer Security (TLS) protocol. It updates RFC 2246, RFC 4346, and RFC 5246. Server update considerations are included.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on August 16, 2015.

Copyright Notice

Table of Contents

1. Introduction

To work around interoperability problems with legacy servers, many TLS client implementations do not rely on the TLS protocol version negotiation mechanism alone, but will intentionally reconnect using a downgraded protocol if initial handshake attempts fail. Such clients may fall back to connections in which they announce a version as low as TLS 1.0 (or even its predecessor, SSL 3.0) as the highest supported version.

While such fallback retries can be a useful last resort for connections to actual legacy servers, there's a risk that active attackers could exploit the downgrade strategy to weaken the cryptographic security of connections. Also, handshake errors due to network glitches could similarly be misinterpreted as interaction with a legacy server and result in a protocol downgrade.

All unnecessary protocol downgrades are undesirable (e.g., from TLS 1.2 to TLS 1.1 if both the client and the server actually do support TLS 1.2); they can be particularly harmful when the result is loss of the TLS extension feature by downgrading to SSL 3.0. This document defines a Signaling Cipher Suite Value (SCSV) that can be employed to prevent unintended protocol downgrades between clients and servers that comply with this document, by having the client indicate that the current connection attempt is merely a fallback, and by having

the server return a fatal alert if it detects an inappropriate fallback. (The alert does not necessarily indicate an intentional downgrade attack, since network glitches too could result in inappropriate fallback retries.)

The fallback SCSV defined in this document is not a suitable substitute for proper TLS version negotiation. TLS implementations need to properly handle TLS version negotiation and extensibility mechanisms to avoid the security issues and connection delays associated with fallback retries.

This specification applies to implementations of TLS 1.0 [RFC2246], TLS 1.1 [RFC4346], and TLS 1.2 [RFC5246], and to implementations of DTLS 1.0 [RFC4347] and DTLS 1.2 [RFC6347]. (It is particularly relevant if the TLS implementations also include support for predecessor protocol SSL 3.0 [RFC6101].) It can be applied similarly to later protocol versions.

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 [RFC2119].

2. Protocol values

This document defines a new TLS cipher suite value:

```
TLS_FALLBACK_SCSV          {0x56, 0x00}
```

This is a signaling cipher suite value (SCSV), i.e., it does not actually correspond to a suite of cryptosystems, and it can never be selected by the server in the handshake; rather, its presence in the Client Hello message serves as a backwards-compatible signal from the client to the server.

This document also allocates a new alert value in the TLS Alert Registry [RFC5246]:

```
enum {
  /* ... */
  inappropriate_fallback(86),
  /* ... */
```

```
        (255)
      } AlertDescription;
```

This alert is only generated by servers, as described in Section 3. It is always fatal.

3. Server behavior

This section specifies server behavior when receiving the TLS_FALLBACK_SCSV cipher suite from a client in ClientHello.cipher_suites.

- If TLS_FALLBACK_SCSV appears in ClientHello.cipher_suites and the highest protocol version supported by the server is higher than the version indicated in ClientHello.client_version, the server MUST respond with a fatal inappropriate_fallback alert (unless it responds with a fatal protocol_version alert because the version indicated in ClientHello.client_version is unsupported). The record layer version number for this alert MUST be set to either ClientHello.client_version (as it would for the Server Hello message if the server was continuing the handshake), or to the record layer version number used by the client.

- Otherwise (either TLS_FALLBACK_SCSV does not appear, or it appears and the client's protocol version is at least the highest protocol version supported by the server), the server proceeds with the handshake as usual.

(A protocol version is supported by the server if, in response to appropriate Client Hello messages, the server would use it for ServerHello.server_version. If a particular protocol version is implemented but completely disabled by server settings, it is not considered supported. For example, if the implementation's highest protocol version is TLS 1.2 but the server operator has disabled this version, a TLS 1.1 Client Hello with TLS_FALLBACK_SCSV does not warrant responding with an inappropriate_fallback alert.)

4. Client behavior

The TLS_FALLBACK_SCSV cipher suite value is meant for use by clients that repeat a connection attempt with a downgraded protocol (perform

a "fallback retry") in order to work around interoperability problems with legacy servers.

- If a client sends a ClientHello.client_version containing a lower value than the latest (highest-valued) version supported by the client, it SHOULD include the TLS_FALLBACK_SCSV cipher suite value in ClientHello.cipher_suites; see Section 6 for security considerations for this recommendation. (The client SHOULD put TLS_FALLBACK_SCSV after all cipher suites that it actually intends to negotiate.)

- As an exception to the above, when a client intends to resume a session and sets ClientHello.client_version to the protocol version negotiated for that session, it MUST NOT include TLS_FALLBACK_SCSV in ClientHello.cipher_suites. (In this case, it is assumed that the client already knows the highest protocol version supported by the server: see [RFC5246], Appendix E.1.)

- If a client sets ClientHello.client_version to its highest supported protocol version, it MUST NOT include TLS_FALLBACK_SCSV in ClientHello.cipher_suites.

(A protocol version is supported by the client if the client normally attempts to use it in handshakes. If a particular protocol version is implemented but completely disabled by client settings, it is not considered supported. For example, if the implementation's highest protocol version is TLS 1.2 but the user has disabled this version, a TLS 1.1 handshake is expected and does not warrant sending TLS_FALLBACK_SCSV.)

Fallback retries could be caused by events such as network glitches, and a client including TLS_FALLBACK_SCSV in ClientHello.cipher_suites may receive an inappropriate_fallback alert in response, indicating that the server supports a higher protocol version. Thus, if a client intends to use retries to work around network glitches, it should then retry with the highest version it supports.

If a client keeps track of the highest protocol version apparently supported by a particular server for use in ClientHello.client_version later, then if the client receives an

inappropriate_fallback alert from that server, it MUST clear the memorized highest supported protocol version. (Without the alert, it is a good idea -- but outside of the scope of this document -- for clients to clear that state after a time-out, since the server's highest protocol version could change over time.)

For clients that use client-side TLS False Start [false-start], it is important to note that the TLS_FALLBACK_SCSV mechanism cannot protect the first round of application data sent by the client: refer to the Security Considerations in [false-start], Section 6.

5. Operational Considerations

Updating legacy server clusters to simultaneously add support for newer protocol versions and support for TLS_FALLBACK_SCSV can have complications, if the legacy server implementation is not "version-tolerant" (cannot properly handle Client Hello messages for newer protocol versions): fallback retries required for interoperability with old server nodes might be rejected by updated server nodes.

Updating the server cluster in two consecutive steps makes this safe: first, update the server software but leave the highest supported version unchanged (by disabling newer versions in server settings); then, after all legacy (version-intolerant) implementations have been removed, change server settings to allow new protocol versions.

6. Security Considerations

Section 4 does not require client implementations to send TLS_FALLBACK_SCSV in any particular case, it merely recommends it; behavior can be adapted according to the client's security needs. It is important to remember that omitting TLS_FALLBACK_SCSV enables downgrade attacks, so implementors must take into account whether the protocol version given by ClientHello.client_version still provides an acceptable level of protection. For example, during the initial deployment of a new protocol version (when some interoperability problems may have to be expected), smoothly falling back to the previous protocol version in case of problems may be preferable to potentially not being able to connect at all: so TLS_FALLBACK_SCSV could be omitted for this particular protocol downgrade step.

However, it is strongly recommended to send TLS_FALLBACK_SCSV when downgrading to SSL 3.0 as the CBC cipher suites in SSL 3.0 have weaknesses that cannot be addressed by implementation workarounds like the remaining weaknesses in later (TLS) protocol versions.

7. IANA Considerations

[[TO BE REMOVED: The requested registry allocations require Standards Action, i.e., will only be official with the IESG's Standards Track RFC approval. Since this document is currently an Internet-Draft, IANA so far has in fact not added the cipher suite number and alert number to the respective registries. The values as shown are used in early implementations.

Value	Description	DTLS-OK	Reference
0x56,0x00	TLS_FALLBACK_SCSV	Y	(this document)

http://www.iana.org/assignments/tls-parameters

Value	Description	DTLS-OK	Reference
86	inappropriate_fallback	Y	(this document)

http://www.iana.org/assignments/tls-parameters

]]

IANA has added TLS cipher suite number 0x56,0x00 with name TLS_FALLBACK_SCSV to the TLS Cipher Suite registry, and alert number 86 with name inappropriate_fallback to the TLS Alert registry.

8. References

8.1. 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.

[RFC4347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security", RFC 4347, April 2006.

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

[RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, January 2012.

8.2. Informative References

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

[false-start] Langley, A., Modadugu, N., and B. Moeller, "Transport Layer Security (TLS) False Start", Work in Progress, draft-bmoeller-tls-falsestart-01, November 2014.

Appendix A. Acknowledgements

This specification was inspired by an earlier proposal by Eric Rescorla. We also thank Daniel Kahn Gillmor, Joe Saloway, Brian Smith, Martin Thomson, and others in the TLS Working Group for their feedback and suggestions.

Authors' Addresses

Bodo Moeller
Google Switzerland GmbH
Brandschenkestrasse 110
Zurich 8002
Switzerland

Email: bmoeller@acm.org

Adam Langley
Google Inc.
345 Spear St
San Francisco, CA 94105
USA

Email: agl@google.com