UTA
Internet-Draft
Intended status: Best Current Practice
Expires: August 17, 2014

Y. Sheffer
Porticor
R. Holz
TUM
P. Saint-Andre
&yet

February 13, 2014

Recommendations for Secure Use of TLS and DTLS
draft-sheffer-tls-bcp-02

Abstract

Transport Layer Security (TLS) and Datagram Transport Security Layer (DTLS) are widely used to protect data exchanged over application protocols such as HTTP, SMTP, IMAP, POP, SIP, and XMPP. Over the last few years, several serious attacks on TLS have emerged, including attacks on its most commonly used cipher suites and modes of operation. This document provides recommendations for improving the security of both software implementations and deployed services that use TLS and DTLS.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

Transport Layer Security (TLS) and Datagram Transport Security Layer (DTLS) are widely used to protect data exchanged over application protocols such as HTTP, SMTP, IMAP, POP, SIP, and XMPP. Over the last few years, several serious attacks on TLS have emerged, including attacks on its most commonly used cipher suites and modes of operation. For instance, both AES-CBC and RC4, which together comprise most current usage, have been attacked in the context of TLS. A companion document [I-D.sheffer-uta-tls-attacks] provides detailed information about these attacks.

Because of these attacks, those who implement and deploy TLS and DTLS need updated guidance on how TLS can be used securely. Note that this document provides guidance for deployed services, as well as software implementations. In fact, this document calls for the deployment of algorithms that are widely implemented but not yet widely deployed.

The recommendations herein take into consideration the security of various mechanisms, their technical maturity and interoperability, and their prevalence in implementatios at the time of writing. These recommendations apply to both TLS and DTLS. TLS 1.3, when it is standardized and deployed in the field, should resolve the current vulnerabilities while providing significantly better functionality, and will very likely obsolete the current document.

Community knowledge about the strength of various algorithms and feasible attacks can change quickly, and experience shows that a security BCP is a point-in-time statement. Readers are advised to seek out any errata or updates that apply to this document.

2. Conventions used in this document

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].

3. Recommendations

3.1. Protocol Versions

It is important both to stop using old, less secure versions of SSL/TLS and to start using modern, more secure versions. Therefore:

- Implementations MUST NOT negotiate SSL version 2.

  Rationale: SSLv2 has serious security vulnerabilities [RFC6176].

- Implementations SHOULD NOT negotiate SSL version 3.

  Rationale: SSLv3 [RFC6101] was an improvement over SSLv2 and plugged some significant security holes, but did not support strong cipher suites.

- Implementations MAY negotiate TLS version 1.0 [RFC2246].

  Rationale: TLS 1.0 (published in 1999) includes a way to downgrade the connection to SSLv3 and does not support more modern, strong cipher suites.

- Implementations MAY negotiate TLS version 1.1 [RFC4346].

  Rationale: TLS 1.1 (published in 2006) prevents downgrade attacks to SSL, but does not support certain stronger cipher suites.

- Implementations MUST support, and prefer to negotiate, TLS version 1.2 [RFC5246].

  Rationale: Several stronger cipher suites are available only with TLS 1.2 (published in 2008).

As of the date of this writing, the latest version of TLS is 1.2. When TLS is updated to a newer version, this document will be updated to recommend support for the latest version. If this document is not updated in a timely manner, it can be assumed that support for the latest version of TLS is recommended.

3.2. Fallback to SSL

Some client implementations revert to SSLv3 if the server rejected higher versions of SSL/TLS. This fallback can be forced by a MITM attacker. Moreover, IP scans [[reference?]] show that SSLv3-only servers amount to only about 3% of the current web server population. Therefore, by default clients SHOULD NOT fall back from TLS to SSLv3.

3.3. Cipher Suites

It is important both to stop using old, insecure cipher suites and to start using modern, more secure cipher suites. Therefore:

- Implementations MUST NOT negotiate the NULL cipher suites.

  Rationale: The NULL cipher suites offer no encryption whatsoever and thus are completely insecure.

- Implementations MUST NOT negotiate RC4 cipher suites

  Rationale: The RC4 stream cipher has a variety of cryptographic weaknesses, as documented in [I-D.popov-tls-prohibiting-rc4].

- Implementations MUST NOT negotiate cipher suites offering only so-called "export-level" encryption (including algorithms with 40 bits or 56 bits of security).

  Rationale: These cipher suites are deliberately "dumbed down" and are very easy to break.

- Implementations SHOULD NOT negotiate cipher suites that use algorithms offering less than 128 bits of security (even if they advertise more bits, such as the 168-bit 3DES cipher suites).

  Rationale: Although these cipher suites are not actively subject to breakage, their useful life is short enough that stronger cipher suites are desirable.

- Implementations SHOULD prefer cipher suites that use algorithms with at least 128 (and, if possible, 256) bits of security.

  Rationale: Although the useful life of such cipher suites is unknown, it is probably at least several years for the 128-bit ciphers and "until the next fundamental technology breakthrough" for 256-bit ciphers.

- Implementations MUST support, and SHOULD prefer to negotiate, cipher suites offering forward secrecy, such as those in the "EDH", "DHE", and "ECDHE" families.

  Rationale: Forward secrecy (sometimes called "perfect forward secrecy") prevents the recovery of information that was encrypted with older session keys, thus limiting the amount of time during which attacks can be successful.

Given the foregoing considerations, implementation of the following cipher suites is RECOMMENDED (see [RFC5289] for details):

- TLS_DHE_RSA_WITH_AES_128_GCM_SHA256

- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256

- TLS_DHE_RSA_WITH_AES_256_GCM_SHA384

- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384

We suggest that TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 be preferred in general.

Unfortunately, those cipher suites are supported only in TLS 1.2 since they are authenticated encryption (AEAD) algorithms [RFC5116]. A future version of this document might recommend cipher suites for earlier versions of TLS.

[RFC4492] allows clients and servers to negotiate ECDH parameters (curves). Clients and servers SHOULD prefer verifiably random curves (specifically Brainpool P-256, brainpoolp256r1 [RFC7027]), and fall back to the commonly used NIST P-256 (secp256r1) curve [RFC4492]. In addition, clients SHOULD send an ec_point_formats extension with a single element, "uncompressed".

3.4. Public Key Length

Because Diffie-Hellman keys of 1024 bits are estimated to be roughly equivalent to 80-bit symmetric keys, it is better to use longer keys for the "DH" family of cipher suites. Unfortunately, some existing software cannot handle (or cannot easily handle) key lengths greater than 1024 bits. The most common workaround for these systems is to prefer the "ECDHE" family of cipher suites instead of the "DH" family, then use longer keys. Key lengths of at least 2048 bits are RECOMMENDED, since they are estimated to be roughly equivalent to 112-bit symmetric keys and might be sufficient for at least the next 10 years. In addition to 2048-bit server certificates, the use of SHA-256 fingerprints is RECOMMENDED (see [CAB-Baseline] for more details).

Note: The foregoing recommendations are preliminary and will likely be corrected and enhanced in a future version of this document.

3.5. Compression

Implementations and deployments SHOULD disable TLS-level compression ([RFC5246], Sec. 6.2.2).

3.6. Session Resumption

If TLS session resumption is used, care ought to be taken to do so safely. In particular, the resumption information (either session IDs [RFC5246] or session tickets [RFC5077]) needs to be authenticated and encrypted to prevent modification or eavesdropping by an attacker. For session tickets, a strong cipher suite SHOULD be used when encrypting the ticket (as least as strong as the main TLS cipher suite); ticket keys MUST be changed regularly, e.g. once every week, so as not to negate the effect of forward secrecy. Session ticket validity SHOULD be limited to a reasonable duration (e.g. 1 day), so as not to negate the benefits of forward secrecy.

4. Detailed Guidelines

The following sections provide more detailed information about the recommendations listed above.

4.1. Cipher Suite Negotiation Details

Clients SHOULD include TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 as the first proposal to any server, unless they have prior knowledge that the server cannot respond to a TLS 1.2 client_hello message.

Servers SHOULD prefer this cipher suite (or a similar but stronger one) whenever it is proposed, even if it is not the first proposal.

Both clients and servers SHOULD include the "Supported Elliptic Curves" extension [RFC4492].

Clients are of course free to offer stronger cipher suites, e.g. using AES-256; when they do, the server SHOULD prefer the stronger cipher suite unless there are compelling reasons (e.g., seriously degraded performance) to choose otherwise.

Note that other profiles of TLS 1.2 exist that use different cipher suites. For example, [RFC6460] defines a profile that uses the TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 and TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 cipher suites.

This document is not an application profile standard, in the sense of Sec. 9 of [RFC5246]. As a result, clients and servers are still required to support the TLS mandatory cipher suite, TLS_RSA_WITH_AES_128_CBC_SHA.

4.2. Alternative Cipher Suites

Elliptic Curves Cryptography is not universally deployed for several reasons, including its complexity compared to modular arithmetic and longstanding IPR concerns. On the other hand, there are two related issues hindering effective use of modular Diffie-Hellman cipher suites in TLS:

- There are no protocol mechanisms to negotiate the DH groups or parameter lengths supported by client and server.

- There are widely deployed client implementations that reject received DH parameters, if they are longer than 1024 bits.

We note that with DHE and ECDHE cipher suites, the TLS master key only depends on the Diffie Hellman parameters and not on the strength the the RSA certificate; moreover, 1024 bits DH parameters are generally considered insufficient at this time.

Because of the above, we recommend using (in priority order):

1. Elliptic Curve DHE with negotiated parameters [RFC5289]

2. TLS_DHE_RSA_WITH_AES_128_GCM_SHA256 [RFC5288], with 2048-bit Diffie-Hellman parameters

3. The same cipher suite, with 1024-bit parameters.

With modular ephemeral DH, deployers SHOULD carefully evaluate interoperability vs. security considerations when configuring their TLS endpoints.

5. IANA Considerations

This document requests no actions of IANA.

6. Security Considerations

6.1. AES-GCM

Please refer to [RFC5246], Sec. 11 for general security considerations when using TLS 1.2, and to [RFC5288], Sec. 6 for security considerations that apply specifically to AES-GCM when used with TLS.

6.2. Forward Secrecy

Forward secrecy (also often called Perfect Forward Secrecy or "PFS") is a defense against an attacker who records encrypted conversations where the session keys are only encrypted with the communicating parties’ long-term keys. Should the attacker be able to obtain these long-term keys at some point later in the future, he will be able to decrypt the session keys and thus the entire conversation. In the context of TLS and DTLS, such compromise of long-term keys is not entirely implausible. It can happen, for example, due to:

- A client or server being attacked by some other attack vector, and the private key retrieved.

- A long-term key retrieved from a device that has been sold or otherwise decommissioned without prior wiping.

- A long-term key used on a device as a default key [Heninger2012].

- A key generated by a Trusted Third Party like a CA, and later retrieved from it either by extortion or compromise [Soghoian2011].

- A cryptographic break-through, or the use of asymmetric keys with insufficient length [Kleinjung2010].

PFS ensures in such cases that the session keys cannot be determined even by an attacker who obtains the long-term keys some time after the conversation. It also protects against an attacker who is in possession of the long-term keys, but remains passive during the conversation.

PFS is generally achieved by using the Diffie-Hellman scheme to derive session keys. The Diffie-Hellman scheme has both parties maintain private secrets and send parameters over the network as modular powers over certain cyclic groups. The properties of the so-called Discrete Logarithm Problem (DLP) allow to derive the session keys without an eavesdropper being able to do so. There is currently no known attack against DLP if sufficiently large parameters are chosen.

Unfortunately, many TLS/DTLS cipher suites were defined that do not enable PFS, e.g. TLS_RSA_WITH_AES_256_CBC_SHA256. We thus advocate strict use of PFS-only ciphers.

7. Acknowledgements

We would like to thank Stephen Farrell, Simon Josefsson, Yoav Nir, Kenny Paterson, Patrick Pelletier, and Rich Salz for their review. Thanks to Brian Smith whose "browser cipher suites" page is a great resource. Finally, thanks to all others who commented on the TLS and other lists and are not mentioned here by name.

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.

[RFC4492] Blake-Wilson, S., Bolyard, N., Gupta, V., Hawk, C., and B. Moeller, "Elliptic Curve Cryptography (ECC) Cipher Suites for Transport Layer Security (TLS)", RFC 4492, May 2006.

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

[RFC5288] Salowey, J., Choudhury, A., and D. McGrew, "AES Galois Counter Mode (GCM) Cipher Suites for TLS", RFC 5288, August 2008.

[RFC5289] Rescorla, E., "TLS Elliptic Curve Cipher Suites with SHA-256/384 and AES Galois Counter Mode (GCM)", RFC 5289, August 2008.

[RFC6176] Turner, S. and T. Polk, "Prohibiting Secure Sockets Layer (SSL) Version 2.0", RFC 6176, March 2011.

[RFC7027] Merkle, J. and M. Lochter, "Elliptic Curve Cryptography (ECC) Brainpool Curves for Transport Layer Security (TLS)", RFC 7027, October 2013.

8.2. Informative References

[CAB-Baseline] "Baseline Requirements for the Issuance and Management of Publicly-Trusted Certificates Version 1.1.6", 2013, <https://www.cabforum.org/documents.html>.

[Heninger2012] Heninger, N., Durumeric, Z., Wustrow, E., and J. Halderman, "Mining Your Ps and Qs: Detection of Widespread Weak Keys in Network Devices", Usenix Security Symposium 2012, 2012.

[I-D.popov-tls-prohibiting-rc4] Popov, A., "Prohibiting RC4 Cipher Suites", draft-popov-tls-prohibiting-rc4-01 (work in progress), October 2013.

[I-D.sheffer-uta-tls-attacks] Sheffer, Y., Holz, R., and P. Saint-Andre, "Summarizing Current Attacks on TLS and DTLS", draft-sheffer-uta-tls-attacks-00 (work in progress), February 2014.

[Kleinjung2010] Kleinjung, T., "Factorization of a 768-Bit RSA Modulus", CRYPTO 10, 2010.

[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.

[RFC5077] Salowey, J., Zhou, H., Eronen, P., and H. Tschofenig, "Transport Layer Security (TLS) Session Resumption without Server-Side State", RFC 5077, January 2008.

[RFC5116] McGrew, D., "An Interface and Algorithms for Authenticated Encryption", RFC 5116, January 2008.

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

[RFC6460] Salter, M. and R. Housley, "Suite B Profile for Transport Layer Security (TLS)", RFC 6460, January 2012.

[Soghoian2011] Soghoian, C. and S. Stamm, "Certified lies: Detecting and defeating government interception attacks against SSL.", Proc. 15th Int. Conf. Financial Cryptography and Data Security , 2011.

Appendix A. Appendix: Change Log

Note to RFC Editor: please remove this section before publication.

A.1. -02

- Reorganized the content to focus on recommendations.

- Moved description of attacks to a separate document (draft-sheffer-uta-tls-attacks).

- Strengthened recommendations regarding session resumption.

A.2. -01

- Clarified our motivation in the introduction.

- Added a section justifying the need for PFS.

- Added recommendations for RSA and DH parameter lengths. Moved from DHE to ECDHE, with a discussion on whether/when DHE is appropriate.

- Recommendation to avoid fallback to SSLv3.

- Initial information about browser support - more still needed!

- More clarity on compression.

- Client can offer stronger cipher suites.

- Discussion of the regular TLS mandatory cipher suite.

A.3. -00

- Initial version.

Authors' Addresses

Yaron Sheffer
Porticor
29 HaHarash St.
Hod HaSharon 4501303
Israel

Email: yaronf.ietf@gmail.com

Ralph Holz
Technische Universitaet Muenchen
Boltzmannstr. 3
Garching 85748
Germany

Email: holz@net.in.tum.de

Peter Saint-Andre
&yet

Email: ietf@stpeter.im