TLS Working Group
Internet-Draft
Intended status: Standards Track
Expires: September 22, 2013

P. Gutmann
University of Auckland

March 21, 2013

Encrypt-then-MAC for TLS
draft-gutmann-tls-encrypt-then-mac-01.txt

Abstract

This document describes a means of negotiating the use of the encrypt-then-MAC security mechanism in place of TLS' existing MAC-then-encrypt one, which has been the subject of a number of security vulnerabilities over a period of many years.

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 September 22, 2013.

Table of Contents

1. Introduction

[TLS] uses a MAC-then-encrypt construction that was regarded as secure at the time the original SSL protocol was specified in the mid-1990s, but that is no longer regarded as secure [EncryptThenMAC] [EncryptThenAuth]. This construction, as used in TLS, has been the subject of numerous security vulnerabilities and attacks stretching over a period of many years. This document specifies a means of switching to the more secure encrypt-then-MAC construction as part of the TLS handshake, replacing the current MAC-then-encrypt construction.

1.1. 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].

2. Negotiating Encrypt-then-MAC

The use of encrypt-then-MAC is negotiated via TLS extensions as defined in [TLS]. On connecting, the client includes the encrypt_then_MAC extension in its client_hello if it wishes to use encrypt-then-MAC rather than the default MAC-then-encrypt. If the server is capable of meeting this requirement, it responds with an encrypt_then_MAC in its server_hello. The "extension_type" value for this extension is [TBD] and the "extension_data" field of this extension SHALL be empty.

2.1. Rationale

The use of TLS extensions to negotiate an overall switch is preferable to defining new ciphersuites because the latter would

result in a Cartesian explosion of suites, potentially requiring duplicating every single existing suite with a new one that uses encrypt-then-MAC. In contrast the approach presented here requires just a single new extension type with a corresponding minimal-length extension sent by client and server.

Another possibility for introducing encrypt-then-MAC would be to make it part of TLS 1.3, however this would require the implementation and deployment of all of TLS 1.2 just to support a trivial code change in the order of encryption and MAC'ing. In contrast deploying encrypt-then-MAC via the TLS extension mechanism required changing less than a dozen lines of code in one implementation (not including the handling for the new extension type, which was a further 50 or so lines of code).

The use of extensions precludes use with SSL 3.0, but then it's likely that anything still using this nearly two decades-old protocol will be vulnerable to any number of other attacks anyway, so there seems little point in bending over backwards to accomodate SSL 3.0.

3. Applying Encrypt-then-MAC

Once the use of encrypt-then-MAC has been negotiated, processing of TLS packets switches from the standard:

```
encrypt( data || MAC || pad )
```

to the new:

```
encrypt( data || pad ) || MAC
```

with the MAC covering the entire packet up to the start of the MAC value. In [TLS] notation the MAC calculation is:

```
MAC(MAC_write_key, seq_num +
    TLSCompressed.type +
    TLSCompressed.version +
    TLSCompressed.length +
    ENC(content + padding + padding_length));
```

for TLS 1.0 without the explicit IV and:

```
MAC(MAC_write_key, seq_num +
    TLSCompressed.type +
    TLSCompressed.version +
```

```
    TLSCompressed.length +
    IV +
    ENC(content + padding + padding_length));
```

for TLS 1.1 and greater with explicit IV. The final MAC value is then appended to the encrypted data and padding. Note that this calculation is identical to the existing one with the exception that the MAC calculation is run over the payload ciphertext rather than the plaintext.

In [TLS] notation the overall packet is then:

```
struct {
       ContentType type;
       ProtocolVersion version;
       uint16 length;
       GenericStream/BlockCipher fragment;
       opaque MAC;
       } TLSCiphertext;
```

Note that this is identical to the existing TLS layout with the single exception being that the MAC value is moved outside the encrypted data.

Decryption reverses this processing. The MAC SHALL be evaluated before any further processing such as decryption is performed, and if the MAC verification fails then processing SHALL terminate immediately. This eliminates any timing channels that may be available through the use of manipulated packet data.

> [Implementation note: There is currently a test server available for interop testing at https://eid.vx4.net:443/. This uses the "extension_type" value 0x10, which is the first currently unassigned TLS extension value. This server has been tested successfully with several different implementations].

4. Security Considerations

This document defines an improved security mechanism encrypt-then-MAC to replace the current MAC-then-encrypt one. This is regarded as more secure than the current mechanism [EncryptThenMAC] [EncryptThenAuth], and should mitigate or eliminate a number of attacks on the current mechanism, provided that the instructions on MAC processing given in Section 3 are applied.

5. IANA Considerations

This document defines a new extension for TLS.

6. Acknowledgements

The author would like to thank the members of the TLS mailing list for their feedback on this document.

7. References

7.1. Normative References

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

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

[3] Blake-Wilson, S., Nystrom, M., Hopwood, D., Mikkelsen, J., and T. Wright, "Transport Layer Security (TLS) Extensions", RFC 4366, April 2006.

7.2. Informative References

[1] Bellare, M. and C. Namprempre, "Authenticated Encryption: Relations among notions and analysis of the generic composition paradigm", Springer-Verlag LNCS 1976, December 2000.

[2] Krawczyk, H., "The Order of Encryption and Authentication for Protecting Communications (or: How Secure Is SSL?)", Springer-Verlag LNCS 2139, August 2001.

Author's Address

Peter Gutmann
University of Auckland
Department of Computer Science
University of Auckland
New Zealand

Email: pgut001@cs.auckland.ac.nz