Internet Engineering Task Force (IETF)
Request for Comments: 6358
Category: Experimental
ISSN: 2070-1721

P. Hoffman
VPN Consortium

January 2012

Additional Master Secret Inputs for TLS

Abstract

This document describes a mechanism for using additional master secret inputs with Transport Layer Security (TLS) and Datagram TLS (DTLS).

Status of This Memo

This document is not an Internet Standards Track specification; it is published for examination, experimental implementation, and evaluation.

This document defines an Experimental Protocol for the Internet community. This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Not all documents approved by the IESG are a candidate for any level of Internet Standard; see Section 2 of RFC 5741.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at http://www.rfc-editor.org/info/rfc6358.

1. Introduction

Some TLS 1.2 [RFC5246] and DTLS 1.2 [RFC6347] extensions want to mix particular data into the calculation of the master secret. This mixing creates a cryptographic binding of the added material directly into the secret that is used to protect the TLS session. For example, some systems want to be sure that there is sufficient randomness in the TLS master secret, and this can be accomplished by adding it directly to the master secret calculations.

This document describes a framework for TLS and DTLS extensions to meet these requirements. In an extension that uses this framework, a client and server provide data in the handshake using normal TLS extensions, and then this data is combined with the ClientHello and ServerHello random values during the derivation of the master_secret.

Extensions that specify data to be added to the master secret are called "extensions with master secret input". An extension with master secret input must specify the additional input that comes from the client and/or the server. Note that the term "and/or" is used here because the definition of the extension might cause input to the master secret to come from only one of the participants.

Note that extensions that do not specify that they are extensions with master secret input cannot be extensions with master secret input. That is, every extension that does not call itself an extension with master secret input is treated just like a normal extension. Also note that this document only describes a framework; if an extension uses this framework, and a client and server both implement the extension, no signaling about the use of master secret input is needed: that comes as part of the extension definition itself.

Use of one or more of these extensions changes the way that the master secret is calculated in TLS and DTLS. That is, if the handshake has no extensions, or only extensions that are not

extensions with master secret input, the master secret calculation is unchanged.

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. Master Secret Calculation Modifications for TLS and DTLS

When an extension with master secret input is present in the handshake, the additional master secret input values MUST be mixed into the pseudorandom function (PRF) calculation along with the client and server random values during the computation of the master secret. For the calculation of the master secret, the extensions MUST be sorted by extension type order. Note that TLS 1.2 specifies that there can only be one extension per type, and the extensions can appear in mixed order.

Each extension with master secret input adds its own specified input, called "additional_ms_input_1" for the extension with master secret input that has the lowest type number, "additional_ms_input_2" for the extension with master secret input with the second lowest type number, and so on.

The calculation of the master secret becomes:

```
master_secret = PRF(pre_master_secret, "master secret",
                    ClientHello.random +
                    ClientHello.additional_ms_input_1 +
                    ClientHello.additional_ms_input_2 +
                    . . .
                    ClientHello.additional_ms_input_N +
                    ServerHello.random +
                    ServerHello.additional_ms_input_1 +
                    ServerHello.additional_ms_input_2 +
                    . . .
                    ServerHello.additional_ms_input_N +
                    )[0..47];
```

Using the specified order of the additional_ms_input_n fields in the master_secret is required for interoperability. Otherwise, a server and a client would not know how to unambiguously calculate the same master_secret.

3. Security Considerations

This modification to TLS and DTLS increases the amount of data that an attacker can inject into the master secret calculation. This potentially would allow an attacker who had partially compromised the inputs to the master secret calculation greater scope for influencing the output. Hash-based PRFs like the one used in TLS master secret calculations are designed to be fairly indifferent to the input size.

The additional master secret input may have no entropy; in fact, it might be completely predictable to an attacker. TLS is designed to function correctly even when the PRF used in the master secret calculation has a great deal of predictable material because the PRF is used to generate distinct keying material for each connection. Thus, even in the face of completely predictable additional master secret input values, no harm is done to the resulting PRF output. When there is entropy in these values, that entropy is reflected in the PRF output.

4. Acknowledgments

Much of the text in this document is derived from text written by Eric Rescorla, Margaret Salter, and Jerry Solinas.

5. Normative References

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

[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.

Author’s Address

Paul Hoffman
VPN Consortium

EMail: paul.hoffman@vpnc.org