Internet Engineering Task Force (IETF)
Request for Comments: 5705
Category: Standards Track
ISSN: 2070-1721

E. Rescorla
RTFM, Inc.

March 2010

Keying Material Exporters for Transport Layer Security (TLS)

Abstract

A number of protocols wish to leverage Transport Layer Security (TLS) to perform key establishment but then use some of the keying material for their own purposes. This document describes a general mechanism for allowing that.

Status of This Memo

This is an Internet Standards Track document.

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). Further information on Internet Standards is available in 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/5705.

Table of Contents

1. Introduction

Note: The mechanism described in this document was previously known as "TLS Extractors" but was changed to avoid a name conflict with the use of the term "Extractor" in the cryptographic community.

A number of protocols wish to leverage Transport Layer Security (TLS) [RFC5246] or Datagram TLS (DTLS) [RFC4347] to perform key establishment but then use some of the keying material for their own purposes. A typical example is DTLS-SRTP [DTLS-SRTP], a key management scheme for the Secure Real-time Transport Protocol (SRTP) that uses DTLS to perform a key exchange and negotiate the SRTP [RFC3711] protection suite and then uses the DTLS master_secret to generate the SRTP keys.

These applications imply a need to be able to export keying material (later called Exported Keying Material or EKM) from TLS/DTLS to an application or protocol residing at an upper layer, and to securely agree on the upper-layer context where the keying material will be used. The mechanism for exporting the keying material has the following requirements:

- Both client and server need to be able to export the same EKM value.

- EKM values should be indistinguishable from random data to attackers who don't know the master_secret.

- It should be possible to export multiple EKM values from the same TLS/DTLS association.

- Knowing one EKM value should not reveal any useful information about the master_secret or about other EKM values.

The mechanism described in this document is intended to fulfill these requirements. This mechanism is compatible with all versions of TLS.

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. Binding to Application Contexts

In addition to using an exporter to obtain keying material, an application using the keying material has to securely establish the upper-layer context where the keying material will be used. The details of this context depend on the application, but it could include things such as algorithms and parameters that will be used with the keys, identifier(s) for the endpoint(s) who will use the keys, identifier(s) for the session(s) where the keys will be used, and the lifetime(s) for the context and/or keys. At a minimum, there should be some mechanism for signaling that an exporter will be used.

This specification does not mandate a single mechanism for agreeing on such context; instead, there are several possibilities that can be used (and can complement each other). For example:

- Information about the upper-layer context can be included in the optional data after the exporter label (see Section 4).

- Information about the upper-layer context can be exchanged in TLS extensions included in the ClientHello and ServerHello messages. This approach is used in [DTLS-SRTP]. The handshake messages are protected by the Finished messages, so once the handshake completes, the peers will have the same view of the information. Extensions also allow a limited form of negotiation: for example, the TLS client could propose several alternatives for some context parameters, and the TLS server could select one of them.

- The upper-layer protocol can include its own handshake, which can be protected using the keys exported by TLS.

No matter how the context is agreed, it is required that it has one part that indicates which application will use the exported keys. This part is the disambiguating label string (see Section 4).

It is important to note that just embedding TLS messages in the upper-layer protocol may not automatically secure all the important context information, since the upper-layer messages are not covered by TLS Finished messages.

4. Exporter Definition

The output of the exporter is intended to be used in a single scope, which is associated with the TLS session, the label, and the context value.

The exporter takes three input values:

- a disambiguating label string,
- a per-association context value provided by the application using the exporter, and
- a length value.

If no context is provided, it then computes:

```
PRF(SecurityParameters.master_secret, label,
    SecurityParameters.client_random +
    SecurityParameters.server_random
    )[length]
```

If context is provided, it computes:

```
PRF(SecurityParameters.master_secret, label,
    SecurityParameters.client_random +
    SecurityParameters.server_random +
    context_value_length + context_value
    )[length]
```

Where PRF is the TLS Pseudorandom Function in use for the session. The output is a pseudorandom bit string of length bytes generated from the master_secret. (This construction allows for interoperability with older exporter-type constructions which do not use context values, e.g., [RFC5281]).

Labels here have the same definition as in TLS, i.e., an ASCII string with no terminating NULL. Label values beginning with "EXPERIMENTAL" MAY be used for private use without registration. All other label

values MUST be registered via Specification Required as described by RFC 5226 [RFC5226]. Note that exporter labels have the potential to collide with existing PRF labels. In order to prevent this, labels SHOULD begin with "EXPORTER". This is not a MUST because there are existing uses that have labels which do not begin with this prefix.

The context value allows the application using the exporter to mix its own data with the TLS PRF for the exporter output. One example of where this might be useful is an authentication setting where the client credentials are valid for more than one identity; the context value could then be used to mix the expected identity into the keying material, thus preventing substitution attacks. The context value length is encoded as an unsigned, 16-bit quantity (uint16; see [RFC5246], Section 4.4) representing the length of the context value. The context MAY be zero length. Because the context value is mixed with the master_secret via the PRF, it is safe to mix confidential information into the exporter, provided that the master_secret will not be known to the attacker.

5. Security Considerations

The prime security requirement for exporter outputs is that they be independent. More formally, after a particular TLS session, if an adversary is allowed to choose multiple (label, context value) pairs and is given the output of the PRF for those values, the attacker is still unable to distinguish between the output of the PRF for a (label, context value) pair (different from the ones that it submitted) and a random value of the same length. In particular, there may be settings, such as the one described in Section 4, where the attacker can control the context value; such an attacker MUST NOT be able to predict the output of the exporter. Similarly, an attacker who does not know the master secret should not be able to distinguish valid exporter outputs from random values. The current set of TLS PRFs is believed to meet this objective, provided the master secret is randomly generated.

Because an exporter produces the same value if applied twice with the same label to the same master_secret, it is critical that two EKM values generated with the same label not be used for two different purposes -- hence, the requirement for IANA registration. However, because exporters depend on the TLS PRF, it is not a threat to the use of an EKM value generated from one label to reveal an EKM value generated from another label.

With certain TLS cipher suites, the TLS master secret is not necessarily unique to a single TLS session. In particular, with RSA key exchange, a malicious party acting as TLS server in one session and as TLS client in another session can cause those two sessions to

have the same TLS master secret (though the sessions must be established simultaneously to get adequate control of the Random values). Applications using the EKM need to consider this in how they use the EKM; in some cases, requiring the use of other cipher suites (such as those using a Diffie-Hellman key exchange) may be advisable.

Designing a secure mechanism that uses exporters is not necessarily straightforward. This document only provides the exporter mechanism, but the problem of agreeing on the surrounding context and the meaning of the information passed to and from the exporter remains. Any new uses of the exporter mechanism should be subject to careful review.

6. IANA Considerations

IANA has created a TLS Exporter Label registry for this purpose. The initial contents of the registry are given below:

Value	Reference	Note
client finished	[RFC5246]	(1)
server finished	[RFC5246]	(1)
master secret	[RFC5246]	(1)
key expansion	[RFC5246]	(1)
client EAP encryption	[RFC5216]	
ttls keying material	[RFC5281]	
ttls challenge	[RFC5281]	

Note: (1) These entries are reserved and MUST NOT be used for the purpose described in RFC 5705, in order to avoid confusion with similar, but distinct, use in RFC 5246.

Future values are allocated via the RFC 5226 Specification Required policy. The label is a string consisting of printable ASCII characters. IANA MUST also verify that one label is not a prefix of any other label. For example, labels "key" or "master secretary" are forbidden.

7. Acknowledgments

Thanks to Pasi Eronen for valuable comments and for the contents of the IANA section and Section 3. Thanks to David McGrew for helpful discussion of the security considerations and to Vijay Gurbani and Alfred Hoenes for editorial comments.

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.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.

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

8.2. Informative References

[DTLS-SRTP] McGrew, D. and E. Rescorla, "Datagram Transport Layer Security (DTLS) Extension to Establish Keys for Secure Real-time Transport Protocol (SRTP)", Work in Progress, February 2009.

[RFC3711] Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, "The Secure Real-time Transport Protocol (SRTP)", RFC 3711, March 2004.

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

[RFC5216] Simon, D., Aboba, B., and R. Hurst, "The EAP-TLS Authentication Protocol", RFC 5216, March 2008.

[RFC5281] Funk, P. and S. Blake-Wilson, "Extensible Authentication Protocol Tunneled Transport Layer Security Authenticated Protocol Version 0 (EAP-TTLSv0)", RFC 5281, August 2008.

Author's Address

Eric Rescorla
RTFM, Inc.
2064 Edgewood Drive
Palo Alto, CA 94303
USA

EMail: ekr@rtfm.com