Network Working Group
Internet-Draft
Intended status: Informational
Expires: April 27, 2010

J. Solinas
National Security Agency
P. Hoffman
VPN Consortium
October 24, 2009

Additional PRF Inputs for TLS
draft-solinas-tls-additional-prf-input-01

Status of this Memo

Copyright Notice

Abstract

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

1. Introduction

TLS [RFC5246] and DTLS [RFC4347] use a 32-byte "Random" value consisting of a 32-bit time value and 28 randomly generated bytes:

```
struct {
    uint32   gmt_unix_time;
    opaque   random_bytes[28];
} Random;
```

The client and server each contribute a Random value which is then mixed with secret keying material to produce the final per-association keying material.

In a number of United States Government applications, it is desirable to have some material that is contributed both by client and server, has an arbitrary length, is possibly structured, and is mixed into the eventual keying material. These requirements are incompatible with the current Random mechanism, which supports a short, fixed-length value.

This document describes a mechanism that meets these requirements. It provides a means for a client and server to exchange data early in the handshake; this data is then concatenated to the ClientHello and ServerHello Randoms during the derivation of the master_secret. This additional data may be encoded with information that is useful to some implementations.

The mechanism described here has several desirable features which may prove useful beyond the requirements of the US Government applications. It is easy to implement, and using it adds little to the overall computation of the TLS handshake. Further, interoperability is preserved between clients that implement the extension and servers that do not, and between clients that do not implement the extension and servers that do.

1.1. Example Usages

Several recommendations for key derivation call for the concatenation of additional information fields into the key derivation function when producing a shared key. For example, the specification of Alternative 1 in NIST Special Publication 800-56A ([SP800-56A]) calls for the inclusion of "OtherInfo" fields. In this case, OtherInfo contains the concatenation of a variety of mandatory and optional sub-fields. The extension described in this document provides a simple mechanism both for supplying the public information to the peer and for the inclusion of the information in the key derivation. Many other application environments require or allow similar fields.

Implementors may want to provide a mechanism for relaying identity and version information similar to the "Vendor ID Payload" used in ISAKMP [RFC2408].

1.2. 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].

2. The AdditionalPRFInput Extension

The additional PRF input is carried in a new TLS extension called "AdditionalPRFInput". The extension is basically a concatenated list of items that will be added into the PRF.

```
struct {
    uint16 AdditionalPRFInputType;
    opaque AdditionalPRFInputValue<0..2^16-4>;
} AdditionalPRFItem;

struct {
   AdditionalPRFItem item_list<0..2^16>
} AdditionalPRFInput;
```

The AdditionalPRFInputType is a value assigned by IANA. The AdditionalPRFInputValue is a byte-string which is generated in an implementation-dependent fashion based on the AdditionalPRFInputType.

2.1. Negotiating the AdditionalPRFInput Extension

The client requests support for the additional PRF input feature by sending an additional_prf_input extension in its ClientHello. The "extension_data" field contains an AdditionalPRFInput.

When a server that does not recognize the additional_prf_input extension receives that extension, [RFC4366] requires that the server will ignore it. A server that recognizes the extension MAY ignore the extension; in such a case, the server MUST continue with the exchange as if it had not received the extension. If the server does not recognize one or more of the AdditionalPRFInputType values, the server MUST continue with the exchange as if it had not received the extension.

If the server wishes to use the additional PRF input feature, it MUST send its own additional_prf_input extension with a non-null AdditionalPRFInput. The AdditionalPRFInput from the server MUST contain the same structure as was received from the client, with each AdditionalPRFInputType included, in the same order as was sent from the client.

Because RFC 4366 does not permit servers to request extensions that the client did not offer, the client might not offer the additional_prf_input extension even if the server requires it for operation. In this case, the server would need to generate a fatal "handshake_failure" alert to stop the connection. Similarly, because there is no way to mark extensions as critical, the server might ignore the additional_prf_input extension even though the client requires it for operation. In this case, the client would need to generate a fatal "handshake_failure" alert to stop the connection. In either case, the peer will not know why the handshake failed (this is true for all TLS extensions that one side requires but cannot communicate that requirement to the other side).

2.2. PRF Modifications

When the additional PRF input feature is in use, the additional PRF input values are mixed into the PRF along with the client and server random values during the conversion of the pre_master_secret to the master_secret. Thus, the PRF becomes:

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

If the additional_prf_input extension is agreed to, the above derivation for the PRF MUST be used.

Because new extensions may not be introduced in resumed handshakes, mixing in the additional PRF inputs during the session resumption PRF invocation would simply involve mixing in the same material twice.

Therefore, the additional PRF inputs are only used when the pre_master_secret is converted into the master_secret.

3. Defined Types

A registry for types of additional PRF input will be maintained by IANA. Types are added to that registry only based on published RFCs. The value in the registry are used in the AdditionalPRFInputType field.

Two types of additional PRF input are defined in this document: AdditionalRandom and OtherInfo.

3.1. AdditionalRandom Type

The AdditionalRandom type, whose AdditionalPRFInputType value is {TBD1}, is used to add additional random values from the client and/or the server to the PRF. The size and the contents of the AdditionalPRFInputValue are undefined (other than it must be between 0 and 2^16-4 bytes). The size of the AdditionalPRFInputValue provided by the client does not indicate anything about the expected size of the AdditionalPRFInputValue from the server.

3.2. OtherInfo Type

The OtherInfo type, whose AdditionalPRFInputType value is {TBD2}, is used to add additional information from the client and/or the server to the PRF. The size and the contents of the AdditionalPRFInputValue are undefined (other than it must be between 0 and 2^16-4 bytes). This type can be used for meeting the requirements in [SP800-56A] and similar documents.

4. Security Considerations

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

The additional PRF 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 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 PRF 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.

It is anticipated that applications may want to have access to the additional PRF input values and that they may contain data that is meaningful at a higher layer. Because the values are covered by the TLS Finished message, they are integrity-protected by TLS. However, the application must independently provide any confidentiality necessary for those values.

5. IANA Considerations

This document defines an extension to TLS, in accordance with RFC 4366:

```
enum { additional_prf_input (TBD) } ExtensionType;
```

IANA will set up a new registry, "Additional PRF Input Types" which will refer to this document. Entries in that registry are added only based on published RFCs. The entries in that registry have three fields: type name, type value, and reference. The type name is free text. The type value is an integer between 0 and 65536. The reference is the RFC that defines the type.

The two initial values for the registry are:

Type name	Type value	Reference
AdditionalRandom	{TBD1}	[This document]
OtherInfo	{TBD2}	[This document]

6. Acknowledgements

This work was supported by the US Department of Defense.

7. References

7.1. Normative References

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

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

[RFC4366] Blake-Wilson, S., Nystrom, M., Hopwood, D., Mikkelsen, J., and T. Wright, "Transport Layer Security (TLS) Extensions", RFC 4366, April 2006.

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

7.2. Informative References

[RFC2408] Maughan, D., Schneider, M., and M. Schertler, "Internet Security Association and Key Management Protocol (ISAKMP)", RFC 2408, November 1998.

[SP800-56A] National Institute of Standards and Technology, U.S. Department of Commerce, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography (Revised)", NIST SP 800-56A, March 2007.

Appendix A. Version History

[[RFC Editor: this entire section is to be removed before final publication.]]

A.1. Changes from draft-solinas-tls-additional-prf-input-00 to -01

Removed "The mechanism also allows the use of random nonce values that are longer than the fixed size of 224 bits" from section 1.1. Also removed the example of TLS servers arranging for their clients to provide authentication data early in the protocol because it was confusing.

Completely changed the structure of the extension in section 2 from being a single blob to being a concatenated list of typed blobs.

In section 2.1, added "If the server does not recognize one or more of the AdditionalPRFInputType values, the server MUST continue with the exchange as if it had not received the extension." Also added "The AdditionalPRFInput from the server MUST contain the same structure as was received from the client, with each AdditionalPRFInputType included, in the same order as was sent from the client.".

In section 2.2, change the PRF calculation to use additional_prf_input_item.

All of section 3 is new.

The registry in the IANA Considerations is new.

Authors' Addresses

Jerome A. Solinas
National Security Agency

Email: jasolin@orion.ncsc.mil

Paul Hoffman
VPN Consortium

Email: paul.hoffman@vpnc.org