Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: July 31, 2011

R. Seggelmann
M. Tuexen
Muenster Univ. of Applied Sciences
M. Williams

January 27, 2011

Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS) Heartbeat Extension
draft-ietf-tls-dtls-heartbeat-01.txt

Abstract

This document describes the Heartbeat Extension for the Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS) protocol.

The Heartbeat Extension provides a new protocol for TLS/DTLS allowing the usage of keep-alive functionality without performing a renegotiation and a basis for path maximum transmission unit (PMTU) discovery for DTLS.

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 July 31, 2011.

Table of Contents

1. Introduction

1.1. Overview

This document describes the Heartbeat Extension for the Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS) protocols, as defined in [RFC5246] and [RFC4347].

DTLS is designed to secure traffic running on top of unreliable transport protocols. Usually such protocols have no session management. The only mechanism available at the DTLS layer to figure out if a peer is still alive is performing a costly renegotiation. If the application uses unidirectional traffic there is no other way. Furthermore, DTLS needs to perform path maximum transmission unit (PMTU) discovery but has no specific message type to realize it without affecting user message transfer.

TLS is based on reliable protocols but there is not necessarily a feature available to keep the connection alive without continuous data transfer.

The Heartbeat Extension as described in this document overcomes these limitations. The user can use the new HeartbeatRequest message which has to be answered by the peer with a HeartbeartResponse immediately. To perform PMTU discovery, HeartbeatRequest messages containing padding can be used as described in [RFC4821] for the Stream Control Transmission Protocol (SCTP) using the padding chunk (PAD-chunk) defined in [RFC4820].

1.2. Conventions

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. Heartbeat Hello Extension

The support of Heartbeats is indicated with Hello Extensions. A peer can not only indicate that its implementation supports Heartbeats, it can also choose whether it is willing to receive and respond or only to send them. This decision can be changed with every renegotiation. HeartbeatRequests MUST NOT be sent to a peer denying acceptance.

```
enum {
   peer_allowed_to_send(1),
   peer_not_allowed_to_send(2),
   (255)
} HeartbeatMode;

struct {
   HeartbeatMode mode;
} HeartbeatExtension;
```

3. Heartbeat Protocol

The Heartbeat protocol is a new protocol on top of the Record Layer. The protocol itself consists of two message types: HeartbeatRequest and HeartbeatResponse.

```
enum {
   heartbeat_request(1),
   heartbeat_response(2),
   (255)
} HeartbeatMessageType;
```

Like the ChangeCipherSpec message, a HeartbeatRequest message can arrive at any time during the lifetime of a connection. Whenever a HeartbeatRequest message is received, it has to be answered with a corresponding HeartbeatResponse message immediately.

However, a HeartbeatRequest message SHOULD NOT be sent during handshakes.

There MUST NOT be more than one HeartbeatRequest message in flight at a time.

When using DTLS, HeartbeatRequest messages MUST be retransmitted using the simple timeout and retransmission scheme DTLS uses for flights. In particular, after a number of retransmissions without receiving a corresponding HeartbeatResponse message having the expected payload the DTLS connection SHOULD be terminated. The threshold used for this SHOULD be the same as for DTLS handshake messages. Please note, that after the timer supervising a HeartbeatRequest messages expires, this message is no longer considered in flight. Therefore the HeartbeatRequest message is eligible for retransmission.

When using TLS, HeartbeatRequest messages only need to be sent once. The transport layer will handle retransmissions. If no corresponding HeartbeatResponse message has been received after a user configured

amount of time, the TLS connection SHOULD be terminated.

4. Heartbeat Request and Response Messages

The Heartbeat protocol messages consist of their type and an arbitrary payload and padding.

```
struct {
   HeartbeatMessageType type;
   uint16 payload_length;
   opaque payload[HeartbeatMessage.payload_length];
   opaque padding[padding_length];
} HeartbeatMessage;
```

The length of a HeartbeatMessage in total MUST NOT exceed 2^14 or max_fragment_length when negotiated as defined in [RFC6066].

type The message type, either heartbeat_request or heartbeat_response.

payload_length The length of the payload.

payload The payload consists of arbitrary content.

padding The padding is additional arbitrary content which MUST be ignored by the receiver. The padding_length is TLSPlaintext.length - payload_length - 3 with TLS and DTLSPlaintext.length - payload_length - 3 with DTLS.

When a HeartbeatRequest message is received, a corresponding HeartbeatResponse message MUST be sent carrying an exact copy of the payload of the HeartbeatRequest. The padding of the received HeartbeatRequest message MUST be ignored. It MUST NOT be included in the HeartbeatResponse message, i.e. the padding field of the HeartbeatResponse message MUST have a length of zero.

If a received HeartbeatResponse message does not contain the expected payload the message MUST be discarded silently. If it does contain the expected payload the retransmission timer MUST be stopped.

If payload_length is either shorter than expected and thus indicates padding in a HeartbeatResponse or exceeds the actual message length in any message type, an illegal parameter alert MUST be sent in response.

5. IANA Considerations

[NOTE to RFC-Editor:

"RFCXXXX" is to be replaced by the RFC number you assign this document.

]

IANA needs to assign the heartbeat content type (value TBD) from the TLS ContentType Registry as specified in [RFC5246]. The reference should be RFCXXXX.

IANA needs to maintain a new registry for Heartbeat Message Types. Initially IANA needs to assign the heartbeat_request (suggested value 1) and the heartbeat_response (suggested value 2) message type. The message types are numbers in the range from 0 to 255 (decimal). This registry uses the Specification Required policy as described in [RFC5226]. The reference should be RFCXXXX.

IANA needs to assign the heartbeat extension type (value TBD) from the TLS Extension Type Registry as specified in [RFC5246]. The reference should be RFCXXXX.

IANA needs to maintain a new registry for Heartbeat Modes. Initially IANA needs to assign the peer_allowed_to_send (suggested value 1) and the peer_not_allowed_to_send (suggested value 2) modes. The modes are numbers in the range from 0 to 255 (decimal). This registry uses the Specification Required policy as described in [RFC5226]. The reference should be RFCXXXX.

6. Security Considerations

This document does not add any additional security considerations in addition to the ones given in [RFC4347] and [RFC5246].

7. Acknowledgments

The authors wish to thank Pasi Eronen, Adam Langley, Eric Rescorla, Peter Saint-Andre, and Juho Vaehae-Herttua for their invaluable 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.

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

[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.

[RFC6066] Eastlake, D., "Transport Layer Security (TLS) Extensions: Extension Definitions", RFC 6066, January 2011.

8.2. Informative References

[RFC4820] Tuexen, M., Stewart, R., and P. Lei, "Padding Chunk and Parameter for the Stream Control Transmission Protocol (SCTP)", RFC 4820, March 2007.

[RFC4821] Mathis, M. and J. Heffner, "Packetization Layer Path MTU Discovery", RFC 4821, March 2007.

Authors' Addresses

Robin Seggelmann
Muenster University of Applied Sciences
Stegerwaldstr. 39
48565 Steinfurt
DE

Email: seggelmann@fh-muenster.de

Michael Tuexen
Muenster University of Applied Sciences
Stegerwaldstr. 39
48565 Steinfurt
DE

Email: tuexen@fh-muenster.de

Michael Williams

Email: michael.glenn.williams@gmail.com