Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 26, 2016

P. Jones
Cisco

February 23, 2016

DTLS Tunnel between Media Distribution Device and Key Management Function to Facilitate Key Exchange
draft-jones-perc-dtls-tunnel-00

Abstract

This document defines a DTLS tunneling protocol for use in multimedia conferences that enables a Media Distribution Device (MDD) to facilitate key exchange between an endpoint in a conference and the Key Management Function (KMF) responsible for key distribution. The protocol is designed to ensure that key material used for end-to-end encryption and authentication is inaccessible to the MDD, while key material used for hop-by-hop encryption and authentication is accessible to the MDD.

Table of Contents

1. Introduction

An objective of the work in the Privacy-Enhanced RTP Conferencing (PERC) working group is to ensure that endpoints in a multimedia conference have access to the end-to-end (E2E) key material used to encrypt and authenticate Real-time Transport Protocol (RTP) [RFC3550] packets, while the Media Distribution Device (MDD) does not. At the same time, the MDD needs access to key material used for hop-by-hop (HBH) encryption and authentication.

The procedures in this document depend on two important media security specifications, namely DTLS-SRTP [RFC5764] and [I-D.ietf-avtcore-srtp-ekt].

DTLS-SRTP [RFC5764] defines a set of procedures for establishing encryption and authentication keys between two entities (e.g., an endpoint and the MDD). [I-D.ietf-avtcore-srtp-ekt] defines a DTLS [RFC6347] extension that build on DTLS-SRTP to allow an entity to transmit a key encrypting key (the "EKT key") to its peer. The EKT key is used to securely convey an SRTP [RFC3711] master key to the peer via an SRTP packet. Together, these procedures would meet the needs of PERC, but care has to be taken to ensure that the MDD does not gain access to the E2E media encryption and authentication key material.

To prevent the MDD from gaining access to the E2E key material, this specification defines a set of procedures for tunneling the DTLS signaling from the endpoint through the MDD to the Key Management Function (KMF). To accomplish this, a DTLS association is first established between the MDD and KMF ("tunnel"). DTLS packets received from the endpoint are encapsulated inside the tunnel as data to be sent to the KMF. Likewise, DTLS messages received inside the tunnel are extracted and forwarded to the endpoint. In effect, the DTLS association for the DTLS-SRTP procedures is established between the endpoint and the KMF, with the MDD simply forwarding packets between the two entities.

Following the existing DTLS-SRTP procedures, the endpoint and KMF will arrive at a selected cipher and key material, which are used for HBH encryption and authentication by both the endpoint and the MDD. However, since the MDD would not have direct access to this information, the KMF will share this information with the MDD via the tunneling protocol defined in this document.

The EKT procedures are used to convey the an EKT key that is shared among all participants in a conference. It is the responsibility of the KMF to send the EKT key for the conference to the endpoint via the DTLS association established between the endpoint and the KMF. The endpoint can then securely transmit its SRTP master keys to other endpoints via SRTP following the procedures in [I-D.ietf-avtcore-srtp-ekt].

By establishing this DTLS tunnel between the MDD and KMF and implementing the protocol defined in this document, it is possible for the MDD to facilitate the establishment of a secure DTLS association between an endpoint and the KMF in order for the endpoint to receive E2E key material and to derive the HBH key material. At the same time, the KMF can securely provide the HBH key material to the MDD.

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 RFC 2119 [RFC2119] when they appear in ALL CAPS. These words may also appear in this document in lower case as plain English words, absent their normative meanings.

3. Tunneling Concept

A DTLS association (tunnel) established between the MDD and the KMF. This tunnel is used to relay DTLS messages between the endpoint and KMF, as depicted in Figure 1:

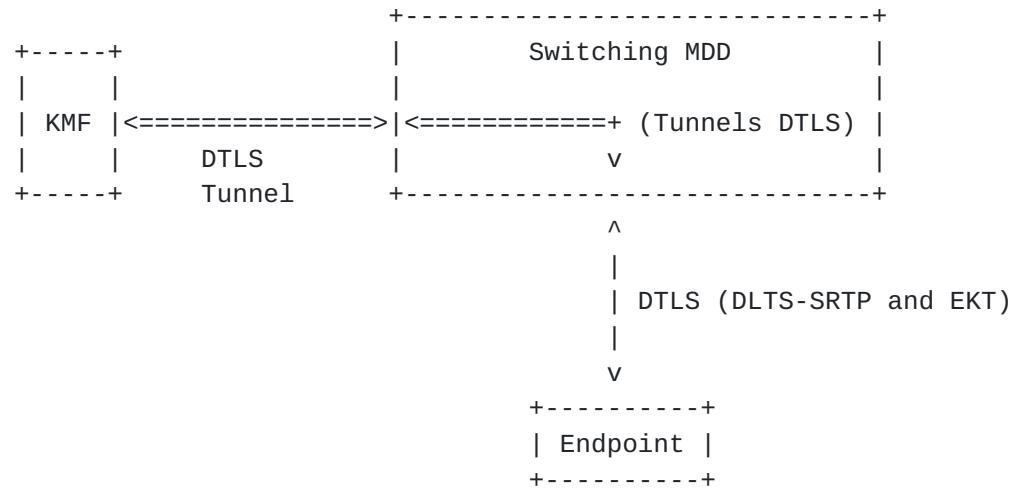

Figure 1: DTLS Tunnel to KMF

The three entities involved in this communication flow are the endpoint, the MDD, and the KMF. The behavior of each entity is described in Section 5.

The KMF is a logical function whose location is not dictated by this document. The KMF might be co-resident with an enterprise key management server, reside in one of the endpoints participating in the conference, or exist elsewhere. What is important is that the KMF is not co-resident with the MDD, as otherwise the MDD will be able to gain access to the E2E key material.

4. Example Message Flows

This section provides some example message flows to help clarify the procedures described later in this document.

Figure 2 shows the establishment of the DTLS tunnel between the MDD and the KMF. The MDD might establish a single tunnel for all communication between itself and the KMF, a single tunnel for each conference, or one tunnel per endpoint. Regardless of how many tunnels the MDD chooses to establish, they are each established in the same way.

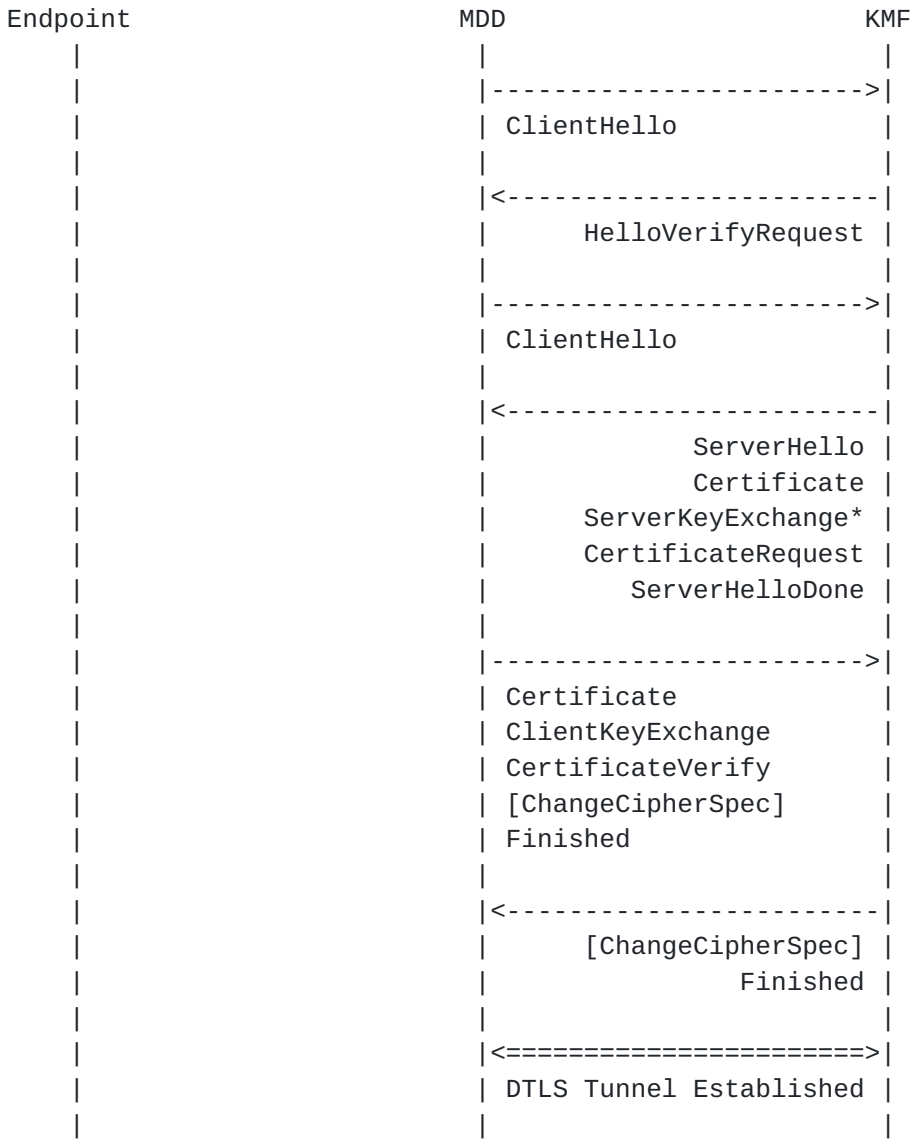

Figure 2: Establishing a DTLS Tunnel

Note that the ServerKeyExchange(*) message is transmitted as required per [RFC5246].

The above flow is almost identical to Figure 1 of [RFC6347], with the only significant change being that, since both client and server certificates must be exchanged, those messages are present and non-optional.

Once the tunnel is established, it is possible for the MDD to tunnel DTLS messages between the endpoint and the KMF. Figure 3 shows a message flow wherein the endpoint uses DTLS-SRTP to establish the HBH cipher and key material, the KMF provides the MDD with the HBH cipher and key material, and the KMF sends the E2E key material to the

endpoint. The tunneled messages are shown with the name of the tunneling protocol message used within parentheses. Tunneled DTLS messages are always carried within the data structure "dtls_message", but the message type is shown in the figure to illustrate which message is sent at which point in the exchange.

Figure 3: Sample DTLS-SRTP and EKT Exchange via a Tunnel

Note that the ServerKeyExchange(*) message is transmitted as required per [RFC5246].

Each of these tunneled messages on the right-hand side of Figure 3 is a message of type "DTLSTunnelMessage" (see Section 6). Each message contains the following information:

* Protocol version
* Association ID
* Conference ID
* Message type (Empty, SupportedProfiles, or SRTPKeyInformation)
* Type-specific content
* DTLS message being tunneled

Of particular interest are the "SupportedProfiles" and "SRTPKeyInformation" messages.

The "SupportedProfiles" message allows the MDD to tell the KMF which protection profiles it uses. The KMF will need to select a common profile supported by both the endpoint and the MDD to ensure that hop-by-hop operations can be successfully performed.

The "SRTPKeyInformation" message contains the KMF-selected cipher and derived key material for those hop-by-hop operations. The derivation of the hop-by-hop key material is performed independently by both the endpoint and the KMF per [RFC5764]. The MDD would extract this information when the message is received and use it for hop-by-hop encryption and authentication operations.

The end-to-end key material is provided by the KMF to the endpoint via the "ekt_key" message as per [I-D.ietf-avtcore-srtp-ekt]. While the EKT message passes through the MDD, it is encrypted and, therefore, inaccessible to the MDD. The endpoint does not send an ekt_key message to the KMF, since only the KMF provides an EKT Key for use in the conference.

5. Tunneling Procedures

The following sub-sections explain in detail the expected behavior of the endpoint, the media distribution device (MDD), and the key management function (KMF).

It is important to note that the tunneling protocol described in this document is not an extension to TLS (@!RFC5246) or DTLS (@!RFC6347). Rather, it is a protocol that carries endpoint- or MDD-generated DTLS messages as data inside of the DTLS tunnel established between the MDD and KMF.

5.1. Endpoint Procedures

The endpoint's role is actually quite simple: it follows the procedures outlined in [RFC5764] without any changes to the procedures defined in that specification in order to establish the keys used for HBH encryption and authentication. Additionally, it uses the procedures defined in [I-D.ietf-avtcore-srtp-ekt] to receive an EKT key to facilitate securing media end-to-end.

The endpoint initiates signaling to establish the DTLS association and expects the KMF to act as the DTLS server. The endpoint MUST verify the KMF's server certificate. The endpoint MUST also provide its certificate to the MDD for verification as a part of the DTLS handshake.

The endpoint exchanges EKT [I-D.ietf-avtcore-srtp-ekt] messages over the DTLS association between itself and the KMF in order to receive the EKT key. The EKT key is used by the endpoint to securely transmit the SRTP master key used for end-to-end media encryption and authentication.

Since the DTLS association is established between the endpoint and the KMF, no entity along the path, including the MDD, will have access to the key material used for E2E encryption and authentication.

5.2. Media Distribution Device Tunneling Procedures

The MDD, acting as a client, establishes a DTLS association between itself and the KMF, acting as a server, for the purpose of facilitating key exchange between an endpoint and the KMF. To differentiate this DTLS association from the one initiated by the endpoint, this association is called a "tunnel". A tunnel may be established when the first endpoint attempts to establish a DTLS association with the KMF, or the tunnel may be established in advance and independent of communication with an endpoint.

A tunnel allows the MDD to relay DTLS messages for any number of endpoints. The MDD cannot see the plaintext contents of the encrypted exchanges between the KMF and an endpoint, but the protocol does enable the KMF to provide the HBH key material to the MDD for each of the individual DTLS associations.

The MDD may establish a single DTLS tunnel to the KMF or it may establish more than one. However, the MDD MUST ensure that all DTLS messages received by the endpoint for the same DTLS association are transmitted over the same tunnel.

When a DTLS message is received by the MDD from an endpoint, it blindly forwards that message to the KMF encapsulated in a "DTLSTunnelMessage" using the message type "Empty" (see Section 6) in all cases except the initial message for each association (as explained below). To uniquely identify a distinct endpoint-originated DTLS association, the MDD assigns a tunnel-unique "association identifier" for the association and includes a "conference identifier" known to both the MDD and the KMF.

The association identifier is necessary since multiple DTLS messages from multiple endpoints might be relayed over the same tunnel. By uniquely assigning an association identifier, the MDD can determine which message received from the KMF needs to be forwarded to which endpoint.

The conference identifier is necessary to allow the KMF to provide the endpoint with the correct E2E key material for the conference the endpoint is attempting to join. It is important to note that merely receiving the conference identifier is not an indication of authorization. Through some means defined outside the scope of this document, it is expected that the KMF will know for which conferences the endpoint is authorized to receive E2E key material.

> Editor's Note: If we enhance EKT so that the endpoint can convey a conference identifier or other information (e.g., a participant ID in the form of a UUID assigned to the endpoint for the conference) that allows the KMF to associate an endpoint and a particular conference, we could relieve the MDD of having to provide a conference ID as a part of the tunneling protocol. Modifying EKT to enable the endpoint to convey this information should be preferred.

All messages for a given DTLS association MUST be sent via the same tunnel and MUST include the same association identifier. The MDD MUST forward all messages received from either the endpoint or the KMF to ensure proper communication between those two entities.

When forwarding the first message received for a new endpoint-originated DTLS association (the "ClientHello + use_srtp + ekt"), the MDD relays the message inside a message of type "SupportedProfiles". This allows the MDD to advertise to the KMF which SRTP protection profiles it supports for HBH operations.

When the MDD receives a message from the KMF of type "SRTPKeyInformation", it extracts the cipher and key material conveyed in that message in order to perform HBH encryption and authentication for RTP/RTCP packets sent to and from the endpoint. Since the HBH cipher and key material will be different for each

endpoint, the MDD uses the association identifier to ensure the key material is associated with the correct endpoint.

5.3. Key Management Function Tunneling Procedures

The KMF MUST be prepared to establish one or more tunnels (DTLS associations) with the MDD for the purpose of relaying DTLS messages between an endpoint and the KMF. The KMF does not initiate a tunnel. Rather, the KMF acts as a server and the MDD acts as a client to establish a tunnel.

When the MDD relays a DTLS message from an endpoint via a tunnel, the MDD will include an association identifier that is unique per endpoint-originated DTLS association relayed via that tunnel. The association identifier remains constant for the life of the DTLS association. Since the same association identifier value might be used on different tunnels between the MDD and KMF, the KMF identifies each endpoint-originated distinct DTLS association by the association identifier and the tunnel over which the DTLS association was established. The KMF MUST use the same association identifier in messages it sends to the endpoint and MUST send all messages for a given DTLS association via the same tunnel. This is to ensure that the MDD can properly relay messages to the correct endpoint.

The KMF extracts tunneled DTLS messages and acts on those messages as if the endpoint had established the DTLS association directly with the KMF. The KMF MUST use a certificate expected by the endpoint. How the endpoint learns of the KMF's certificate or certificate fingerprint is outside the scope of this document.

The endpoint MUST provide a certificate to the KMF for validation. How the KMF is able to determine that a certificate belongs to a particular endpoint is outside the scope of this document.

When sending a message to the endpoint, the KMF encapsulates the message in the DTLSTunnelMessage.dtls_message field of the tunnel protocol. Messages are normally tunneled using the message type "Empty", except when the KMF provides cipher and key material for HBH encryption and authentication (explained below).

The KMF acts as the server in the DTLS-SRTP exchanges with the endpoint, so the KMF will dictate to the endpoint which cipher to employ for HBH operations. The selected cipher is conveyed in the ExtendedServerHello message (per [RFC5764]) to the endpoint, which is merely tunneled through the MDD and otherwise ignored by the MDD.

Once the SRTP master key and salt values for HBH encryption and authentication are derived by the KMF, those values and the selected

cipher are conveyed to the MDD when the KMF transmits the Finished message to the endpoint. The Finished message is encapsulated inside the tunnel in a message of type "SRTPKeyInformation".

After sending the Finished message, the KMF will send an ekt_key message to the endpoint containing the EKT key used in the conference.

6. Tunneling Protocol

The protocol syntax for the DTLS tunnel established between the MDD and KMF is shown below. The syntax is borrowed from [RFC5246].

```
enum {
    empty(0),
    supported_profiles(1),
    srtp_key_information(2),
    (255)
} KMFMessageType;

struct {
    uint8 major;
    uint8 minor;
} ProtocolVersion;

struct {
    ProtocolVersion version;    /* Defined as {0x01, 0x00}  */
    opaque association_id[16];  /* MDD-defined              */
    opaque conference_id[16];   /* Conference identifier    */
    KMFMessageType type;        /* Types defined above      */
    select(KMFMessageType) {
        case empty: Empty;
                /* Used for most tunneled messages          */
        case supported_profiles: SupportedProfiles;
                /* MDD->KMF only; supported profiles        */
        case srtp_key_information: SRTPKeyInformation;
                /* KMF->MDD only; HBH cipher and key info   */
    } body;
    opaque dtls_message<0..2^16-1>;
                /* Encapsulated DTLS message                */
} DTLSTunnelMessage;

struct { } Empty;

/* Much of the following is borrowed from RFC 5764 */

uint8 SRTPProtectionProfile[2];
```

```
SRTPProtectionProfile SRTPProtectionProfiles<2..2^16-1>;

struct {
    SRTPProtectionProfiles SRTPProtectionProfiles;
    opaque srtp_mki<0..255>;
} SupportedProfiles;

struct {
    uint8 protection_profile[2];
    opaque client_write_SRTP_master_key<1..2^16-1>;
    opaque server_write_SRTP_master_key<1..2^16-1>;
    opaque client_write_SRTP_master_salt<1..2^16-1>;
    opaque server_write_SRTP_master_salt<1..2^16-1>;
} SRTPKeyInformation;
```

7. IANA Considerations

There are no IANA considerations for this document.

8. Security Considerations

[TBD]

9. Acknowledgments

The author would like to thank David Benham for reviewing this document and providing constructive comments.

10. Normative References

[I-D.ietf-avtcore-srtp-ekt]
Mattsson, J., McGrew, D., and D. Wing, "Encrypted Key Transport for Secure RTP", draft-ietf-avtcore-srtp-ekt-03 (work in progress), October 2014.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, DOI 10.17487/RFC3550, July 2003, <http://www.rfc-editor.org/info/rfc3550>.

[RFC3711] Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K. Norrman, "The Secure Real-time Transport Protocol (SRTP)", RFC 3711, DOI 10.17487/RFC3711, March 2004, <http://www.rfc-editor.org/info/rfc3711>.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <http://www.rfc-editor.org/info/rfc5246>.

[RFC5764] McGrew, D. and E. Rescorla, "Datagram Transport Layer Security (DTLS) Extension to Establish Keys for the Secure Real-time Transport Protocol (SRTP)", RFC 5764, DOI 10.17487/RFC5764, May 2010, <http://www.rfc-editor.org/info/rfc5764>.

[RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, DOI 10.17487/RFC6347, January 2012, <http://www.rfc-editor.org/info/rfc6347>.

Author's Address

Paul Jones
Cisco
7025 Kit Creek Rd.
Research Trianle Park, North Carolina 27709
USA

Phone: +1 919 476 2048
Email: paulej@packetizer.com