Internet Engineering Task Force (IETF)
Request for Comments: 5763
Category: Standards Track
ISSN: 2070-1721

J. Fischl
Skype, Inc.
H. Tschofenig
Nokia Siemens Networks
E. Rescorla
RTFM, Inc.

May 2010

Framework for Establishing a Secure Real-time Transport Protocol (SRTP) Security Context Using Datagram Transport Layer Security (DTLS)

Abstract

This document specifies how to use the Session Initiation Protocol (SIP) to establish a Secure Real-time Transport Protocol (SRTP) security context using the Datagram Transport Layer Security (DTLS) protocol. It describes a mechanism of transporting a fingerprint attribute in the Session Description Protocol (SDP) that identifies the key that will be presented during the DTLS handshake. The key exchange travels along the media path as opposed to the signaling path. The SIP Identity mechanism can be used to protect the integrity of the fingerprint attribute from modification by intermediate proxies.

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/rfc5763.

Table of Contents

1. Introduction

The Session Initiation Protocol (SIP) [RFC3261] and the Session Description Protocol (SDP) [RFC4566] are used to set up multimedia sessions or calls. SDP is also used to set up TCP [RFC4145] and additionally TCP/TLS connections for usage with media sessions [RFC4572]. The Real-time Transport Protocol (RTP) [RFC3550] is used to transmit real-time media on top of UDP and TCP [RFC4571]. Datagram TLS [RFC4347] was introduced to allow TLS functionality to be applied to datagram transport protocols, such as UDP and DCCP. This document provides guidelines on how to establish SRTP [RFC3711] security over UDP using an extension to DTLS (see [RFC5764]).

The goal of this work is to provide a key negotiation technique that allows encrypted communication between devices with no prior relationships. It also does not require the devices to trust every call signaling element that was involved in routing or session setup. This approach does not require any extra effort by end users and does not require deployment of certificates that are signed by a well-known certificate authority to all devices.

The media is transported over a mutually authenticated DTLS session where both sides have certificates. It is very important to note that certificates are being used purely as a carrier for the public keys of the peers. This is required because DTLS does not have a mode for carrying bare keys, but it is purely an issue of formatting. The certificates can be self-signed and completely self-generated. All major TLS stacks have the capability to generate such certificates on demand. However, third-party certificates MAY also be used if the peers have them (thus reducing the need to trust intermediaries). The certificate fingerprints are sent in SDP over SIP as part of the offer/answer exchange.

The fingerprint mechanism allows one side of the connection to verify that the certificate presented in the DTLS handshake matches the certificate used by the party in the signaling. However, this requires some form of integrity protection on the signaling. S/MIME signatures, as described in RFC 3261, or SIP Identity, as described in [RFC4474], provide the highest level of security because they are not susceptible to modification by malicious intermediaries. However, even hop-by-hop security, such as provided by SIPS, offers some protection against modification by attackers who are not in control of on-path signaling elements. Because DTLS-SRTP only requires message integrity and not confidentiality for the signaling, the number of elements that must have credentials and be trusted is significantly reduced. In particular, if RFC 4474 is used, only the Authentication Service need have a certificate and be trusted. Intermediate elements cannot undetectably modify the message and

therefore cannot mount a man-in-the-middle (MITM) attack. By comparison, because SDESCRIPTIONS [RFC4568] requires confidentiality for the signaling, all intermediate elements must be trusted.

This approach differs from previous attempts to secure media traffic where the authentication and key exchange protocol (e.g., Multimedia Internet KEYing (MIKEY) [RFC3830]) is piggybacked in the signaling message exchange. With DTLS-SRTP, establishing the protection of the media traffic between the endpoints is done by the media endpoints with only a cryptographic binding of the media keying to the SIP/SDP communication. It allows RTP and SIP to be used in the usual manner when there is no encrypted media.

In SIP, typically the caller sends an offer and the callee may subsequently send one-way media back to the caller before a SIP answer is received by the caller. The approach in this specification, where the media key negotiation is decoupled from the SIP signaling, allows the early media to be set up before the SIP answer is received while preserving the important security property of allowing the media sender to choose some of the keying material for the media. This also allows the media sessions to be changed, rekeyed, and otherwise modified after the initial SIP signaling without any additional SIP signaling.

Design decisions that influence the applicability of this specification are discussed in Section 3.

2. Overview

Endpoints wishing to set up an RTP media session do so by exchanging offers and answers in SDP messages over SIP. In a typical use case, two endpoints would negotiate to transmit audio data over RTP using the UDP protocol.

Figure 1 shows a typical message exchange in the SIP trapezoid.

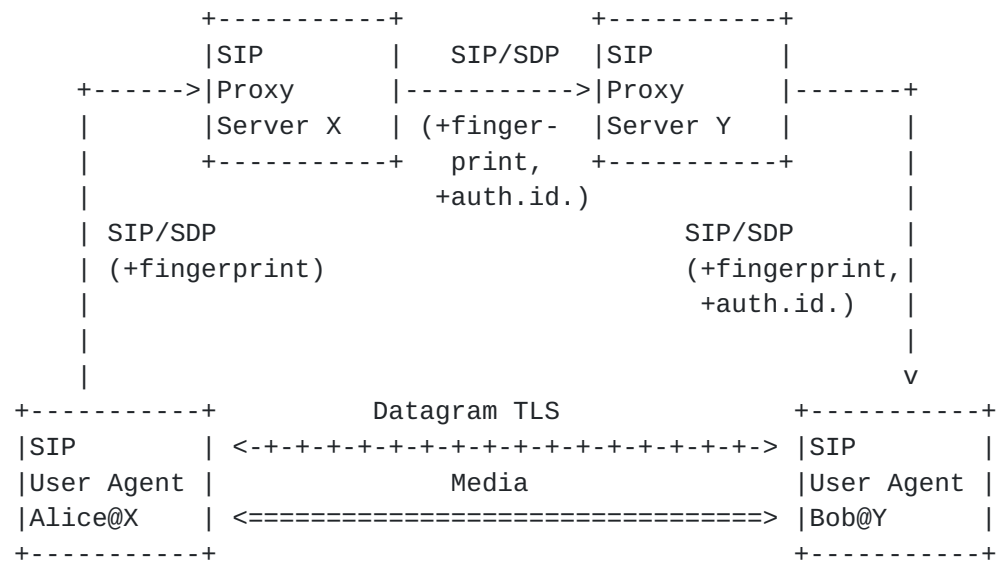


Figure 1: DTLS Usage in the SIP Trapezoid

Consider Alice wanting to set up an encrypted audio session with Bob. Both Bob and Alice could use public-key-based authentication in order to establish a confidentiality protected channel using DTLS.

Since providing mutual authentication between two arbitrary endpoints on the Internet using public-key-based cryptography tends to be problematic, we consider more deployment-friendly alternatives. This document uses one approach and several others are discussed in Section 8.

Alice sends an SDP offer to Bob over SIP. If Alice uses only self-signed certificates for the communication with Bob, a fingerprint is included in the SDP offer/answer exchange. This fingerprint binds the DTLS key exchange in the media plane to the signaling plane.

The fingerprint alone protects against active attacks on the media but not active attacks on the signaling. In order to prevent active attacks on the signaling, "Enhancements for Authenticated Identity Management in the Session Initiation Protocol (SIP)" [RFC4474] may be used. When Bob receives the offer, the peers establish some number of DTLS connections (depending on the number of media sessions) with mutual DTLS authentication (i.e., both sides provide certificates). At this point, Bob can verify that Alice's credentials offered in TLS match the fingerprint in the SDP offer, and Bob can begin sending

media to Alice. Once Bob accepts Alice's offer and sends an SDP answer to Alice, Alice can begin sending confidential media to Bob over the appropriate streams. Alice and Bob will verify that the fingerprints from the certificates received over the DTLS handshakes match with the fingerprints received in the SDP of the SIP signaling. This provides the security property that Alice knows that the media traffic is going to Bob and vice versa without necessarily requiring global Public Key Infrastructure (PKI) certificates for Alice and Bob. (See Section 8 for detailed security analysis.)

3. Motivation

Although there is already prior work in this area (e.g., Security Descriptions for SDP [RFC4568], Key Management Extensions [RFC4567] combined with MIKEY [RFC3830] for authentication and key exchange), this specification is motivated as follows:

- TLS will be used to offer security for connection-oriented media. The design of TLS is well-known and implementations are widely available.

- This approach deals with forking and early media without requiring support for Provisional Response ACKnowledgement (PRACK) [RFC3262] while preserving the important security property of allowing the offerer to choose keying material for encrypting the media.

- The establishment of security protection for the media path is also provided along the media path and not over the signaling path. In many deployment scenarios, the signaling and media traffic travel along a different path through the network.

- When RFC 4474 is used, this solution works even when the SIP proxies downstream of the authentication service are not trusted. There is no need to reveal keys in the SIP signaling or in the SDP message exchange, as is done in SDESCRIPTIONS [RFC4568]. Retargeting of a dialog-forming request (changing the value of the Request-URI), the User Agent (UA) that receives it (the User Agent Server, UAS) can have a different identity from that in the To header field. When RFC 4916 is used, then it is possible to supply its identity to the peer UA by means of a request in the reverse direction, and for that identity to be signed by an Authentication Service.

- In this method, synchronization source (SSRC) collisions do not result in any extra SIP signaling.

- Many SIP endpoints already implement TLS. The changes to existing SIP and RTP usage are minimal even when DTLS-SRTP [RFC5764] is used.

4. Terminology

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].

DTLS/TLS uses the term "session" to refer to a long-lived set of keying material that spans associations. In this document, consistent with SIP/SDP usage, we use it to refer to a multimedia session and use the term "TLS session" to refer to the TLS construct. We use the term "association" to refer to a particular DTLS cipher suite and keying material set that is associated with a single host/port quartet. The same DTLS/TLS session can be used to establish the keying material for multiple associations. For consistency with other SIP/SDP usage, we use the term "connection" when what's being referred to is a multimedia stream that is not specifically DTLS/TLS.

In this document, the term "Mutual DTLS" indicates that both the DTLS client and server present certificates even if one or both certificates are self-signed.

5. Establishing a Secure Channel

The two endpoints in the exchange present their identities as part of the DTLS handshake procedure using certificates. This document uses certificates in the same style as described in "Connection-Oriented Media Transport over the Transport Layer Security (TLS) Protocol in the Session Description Protocol (SDP)" [RFC4572].

If self-signed certificates are used, the content of the subjectAltName attribute inside the certificate MAY use the uniform resource identifier (URI) of the user. This is useful for debugging purposes only and is not required to bind the certificate to one of the communication endpoints. The integrity of the certificate is ensured through the fingerprint attribute in the SDP. The subjectAltName is not an important component of the certificate verification.

The generation of public/private key pairs is relatively expensive. Endpoints are not required to generate certificates for each session.

The offer/answer model, defined in [RFC3264], is used by protocols like the Session Initiation Protocol (SIP) [RFC3261] to set up multimedia sessions. In addition to the usual contents of an SDP

[RFC4566] message, each media description ("m=" line and associated parameters) will also contain several attributes as specified in [RFC5764], [RFC4145], and [RFC4572].

When an endpoint wishes to set up a secure media session with another endpoint, it sends an offer in a SIP message to the other endpoint. This offer includes, as part of the SDP payload, the fingerprint of the certificate that the endpoint wants to use. The endpoint SHOULD send the SIP message containing the offer to the offerer's SIP proxy over an integrity protected channel. The proxy SHOULD add an Identity header field according to the procedures outlined in [RFC4474]. The SIP message containing the offer SHOULD be sent to the offerer's SIP proxy over an integrity protected channel. When the far endpoint receives the SIP message, it can verify the identity of the sender using the Identity header field. Since the Identity header field is a digital signature across several SIP header fields, in addition to the body of the SIP message, the receiver can also be certain that the message has not been tampered with after the digital signature was applied and added to the SIP message.

The far endpoint (answerer) may now establish a DTLS association with the offerer. Alternately, it can indicate in its answer that the offerer is to initiate the TLS association. In either case, mutual DTLS certificate-based authentication will be used. After completing the DTLS handshake, information about the authenticated identities, including the certificates, are made available to the endpoint application. The answerer is then able to verify that the offerer's certificate used for authentication in the DTLS handshake can be associated to the certificate fingerprint contained in the offer in the SDP. At this point, the answerer may indicate to the end user that the media is secured. The offerer may only tentatively accept the answerer's certificate since it may not yet have the answerer's certificate fingerprint.

When the answerer accepts the offer, it provides an answer back to the offerer containing the answerer's certificate fingerprint. At this point, the offerer can accept or reject the peer's certificate and the offerer can indicate to the end user that the media is secured.

Note that the entire authentication and key exchange for securing the media traffic is handled in the media path through DTLS. The signaling path is only used to verify the peers' certificate fingerprints.

The offer and answer MUST conform to the following requirements.

- The endpoint MUST use the setup attribute defined in [RFC4145]. The endpoint that is the offerer MUST use the setup attribute value of setup:actpass and be prepared to receive a client_hello before it receives the answer. The answerer MUST use either a setup attribute value of setup:active or setup:passive. Note that if the answerer uses setup:passive, then the DTLS handshake will not begin until the answerer is received, which adds additional latency. setup:active allows the answer and the DTLS handshake to occur in parallel. Thus, setup:active is RECOMMENDED. Whichever party is active MUST initiate a DTLS handshake by sending a ClientHello over each flow (host/port quartet).

- The endpoint MUST NOT use the connection attribute defined in [RFC4145].

- The endpoint MUST use the certificate fingerprint attribute as specified in [RFC4572].

- The certificate presented during the DTLS handshake MUST match the fingerprint exchanged via the signaling path in the SDP. The security properties of this mechanism are described in Section 8.

- If the fingerprint does not match the hashed certificate, then the endpoint MUST tear down the media session immediately. Note that it is permissible to wait until the other side's fingerprint has been received before establishing the connection; however, this may have undesirable latency effects.

6. Miscellaneous Considerations

6.1. Anonymous Calls

The use of DTLS-SRTP does not provide anonymous calling; however, it also does not prevent it. However, if care is not taken when anonymous calling features, such as those described in [RFC3325] or [RFC5767] are used, DTLS-SRTP may allow deanonymizing an otherwise anonymous call. When anonymous calls are being made, the following procedures SHOULD be used to prevent deanonymization.

When making anonymous calls, a new self-signed certificate SHOULD be used for each call so that the calls cannot be correlated as to being from the same caller. In situations where some degree of correlation is acceptable, the same certificate SHOULD be used for a number of calls in order to enable continuity of authentication; see Section 8.4.

Additionally, note that in networks that deploy [RFC3325], RFC 3325 requires that the Privacy header field value defined in [RFC3323] needs to be set to 'id'. This is used in conjunction with the SIP identity mechanism to ensure that the identity of the user is not asserted when enabling anonymous calls. Furthermore, the content of the subjectAltName attribute inside the certificate MUST NOT contain information that either allows correlation or identification of the user that wishes to place an anonymous call. Note that following this recommendation is not sufficient to provide anonymization.

6.2. Early Media

If an offer is received by an endpoint that wishes to provide early media, it MUST take the setup:active role and can immediately establish a DTLS association with the other endpoint and begin sending media. The setup:passive endpoint may not yet have validated the fingerprint of the active endpoint's certificate. The security aspects of media handling in this situation are discussed in Section 8.

6.3. Forking

In SIP, it is possible for a request to fork to multiple endpoints. Each forked request can result in a different answer. Assuming that the requester provided an offer, each of the answerers will provide a unique answer. Each answerer will form a DTLS association with the offerer. The offerer can then securely correlate the SDP answer received in the SIP message by comparing the fingerprint in the answer to the hashed certificate for each DTLS association.

6.4. Delayed Offer Calls

An endpoint may send a SIP INVITE request with no offer in it. When this occurs, the receiver(s) of the INVITE will provide the offer in the response and the originator will provide the answer in the subsequent ACK request or in the PRACK request [RFC3262], if both endpoints support reliable provisional responses. In any event, the active endpoint still establishes the DTLS association with the passive endpoint as negotiated in the offer/answer exchange.

6.5. Multiple Associations

When there are multiple flows (e.g., multiple media streams, non-multiplexed RTP and RTCP, etc.) the active side MAY perform the DTLS handshakes in any order. Appendix B of [RFC5764] provides some guidance on the performance of parallel DTLS handshakes. Note that if the answerer ends up being active, it may only initiate handshakes on some subset of the potential streams (e.g., if audio and video are

offered but it only wishes to do audio). If the offerer ends up being active, the complete answer will be received before the offerer begins initiating handshakes.

6.6. Session Modification

Once an answer is provided to the offerer, either endpoint MAY request a session modification that MAY include an updated offer. This session modification can be carried in either an INVITE or UPDATE request. The peers can reuse the existing associations if they are compatible (i.e., they have the same key fingerprints and transport parameters), or establish a new one following the same rules are for initial exchanges, tearing down the existing association as soon as the offer/answer exchange is completed. Note that if the active/passive status of the endpoints changes, a new connection MUST be established.

6.7. Middlebox Interaction

There are a number of potentially bad interactions between DTLS-SRTP and middleboxes, as documented in [MMUSIC-MEDIA], which also provides recommendations for avoiding such problems.

6.7.1. ICE Interaction

Interactive Connectivity Establishment (ICE), as specified in [RFC5245], provides a methodology of allowing participants in multimedia sessions to verify mutual connectivity. When ICE is being used, the ICE connectivity checks are performed before the DTLS handshake begins. Note that if aggressive nomination mode is used, multiple candidate pairs may be marked valid before ICE finally converges on a single candidate pair. Implementations MUST treat all ICE candidate pairs associated with a single component as part of the same DTLS association. Thus, there will be only one DTLS handshake even if there are multiple valid candidate pairs. Note that this may mean adjusting the endpoint IP addresses if the selected candidate pair shifts, just as if the DTLS packets were an ordinary media stream.

Note that Simple Traversal of the UDP Protocol through NAT (STUN) packets are sent directly over UDP, not over DTLS. [RFC5764] describes how to demultiplex STUN packets from DTLS packets and SRTP packets.

6.7.2. Latching Control without ICE

If ICE is not being used, then there is potential for a bad interaction with Session Border Controllers (SBCs) via "latching", as described in [MMUSIC-MEDIA]. In order to avoid this issue, if ICE is not being used and the DTLS handshake has not completed upon receiving the other side's SDP, then the passive side MUST do a single unauthenticated STUN [RFC5389] connectivity check in order to open up the appropriate pinhole. All implementations MUST be prepared to answer this request during the handshake period even if they do not otherwise do ICE. However, the active side MUST proceed with the DTLS handshake as appropriate even if no such STUN check is received and the passive MUST NOT wait for a STUN answer before sending its ServerHello.

6.8. Rekeying

As with TLS, DTLS endpoints can rekey at any time by redoing the DTLS handshake. While the rekey is under way, the endpoints continue to use the previously established keying material for usage with DTLS. Once the new session keys are established, the session can switch to using these and abandon the old keys. This ensures that latency is not introduced during the rekeying process.

Further considerations regarding rekeying in case the SRTP security context is established with DTLS can be found in Section 3.7 of [RFC5764].

6.9. Conference Servers and Shared Encryptions Contexts

It has been proposed that conference servers might use the same encryption context for all of the participants in a conference. The advantage of this approach is that the conference server only needs to encrypt the output for all speakers instead of once per participant.

This shared encryption context approach is not possible under this specification because each DTLS handshake establishes fresh keys that are not completely under the control of either side. However, it is argued that the effort to encrypt each RTP packet is small compared to the other tasks performed by the conference server such as the codec processing.

Future extensions, such as [SRTP-EKT] or [KEY-TRANSPORT], could be used to provide this functionality in concert with the mechanisms described in this specification.

6.10. Media over SRTP

Because DTLS's data transfer protocol is generic, it is less highly optimized for use with RTP than is SRTP [RFC3711], which has been specifically tuned for that purpose. DTLS-SRTP [RFC5764] has been defined to provide for the negotiation of SRTP transport using a DTLS connection, thus allowing the performance benefits of SRTP with the easy key management of DTLS. The ability to reuse existing SRTP software and hardware implementations may in some environments provide another important motivation for using DTLS-SRTP instead of RTP over DTLS. Implementations of this specification MUST support DTLS-SRTP [RFC5764].

6.11. Best Effort Encryption

[RFC5479] describes a requirement for best-effort encryption where SRTP is used and where both endpoints support it and key negotiation succeeds, otherwise RTP is used.

[MMUSIC-SDP] describes a mechanism that can signal both RTP and SRTP as an alternative. This allows an offerer to express a preference for SRTP, but RTP is the default and will be understood by endpoints that do not understand SRTP or this key exchange mechanism. Implementations of this document MUST support [MMUSIC-SDP].

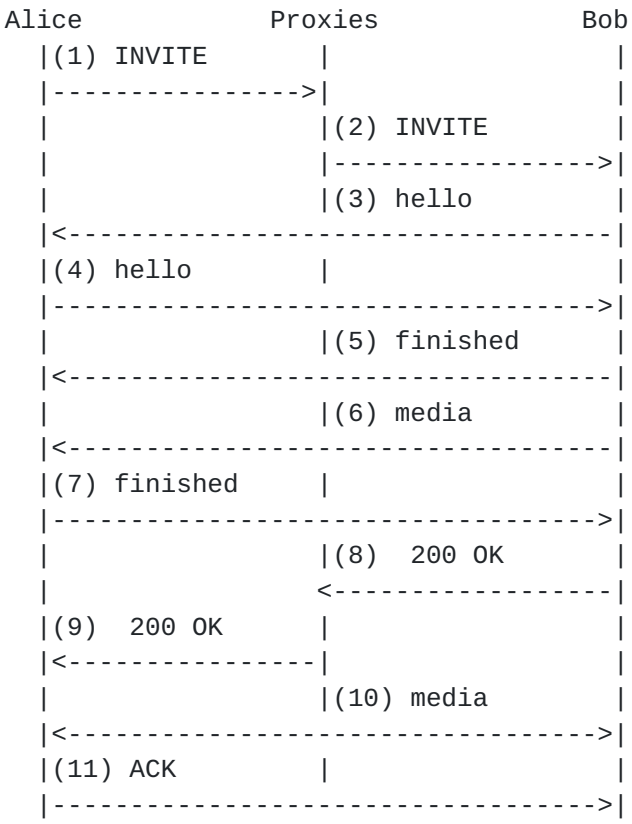
7. Example Message Flow

Prior to establishing the session, both Alice and Bob generate self-signed certificates that are used for a single session or, more likely, reused for multiple sessions. In this example, Alice calls Bob. In this example, we assume that Alice and Bob share the same proxy.

7.1. Basic Message Flow with Early Media and SIP Identity

This example shows the SIP message flows where Alice acts as the passive endpoint and Bob acts as the active endpoint; meaning that as soon as Bob receives the INVITE from Alice, with DTLS specified in the "m=" line of the offer, Bob will begin to negotiate a DTLS association with Alice for both RTP and RTCP streams. Early media (RTP and RTCP) starts to flow from Bob to Alice as soon as Bob sends the DTLS finished message to Alice. Bi-directional media (RTP and RTCP) can flow after Alice receives the SIP 200 response and once Alice has sent the DTLS finished message.

The SIP signaling from Alice to her proxy is transported over TLS to ensure an integrity protected channel between Alice and her identity service. Transport between proxies should also be protected somehow, especially if SIP Identity is not in use.



Message (1): INVITE Alice -> Proxy

This shows the initial INVITE from Alice to Bob carried over the TLS transport protocol to ensure an integrity protected channel between Alice and her proxy that acts as Alice's identity service. Alice has requested to be either the active or passive endpoint by specifying a=setup:actpass in the SDP. Bob chooses to act as the DTLS client and will initiate the session. Also note that there is a fingerprint attribute in the SDP. This is computed from Alice's self-signed certificate.

This offer includes a default "m=" line offering RTP in case the answerer does not support SRTP. However, the potential configuration utilizing a transport of SRTP is preferred. See [MMUSIC-SDP] for more details on the details of SDP capability negotiation.

```
INVITE sip:bob@example.com SIP/2.0
To: <sip:bob@example.com>
From: "Alice"<sip:alice@example.com>;tag=843c7b0b
Via: SIP/2.0/TLS ua1.example.com;branch=z9hG4bK-0e53sadfkasldkfj
Contact: <sip:alice@ua1.example.com>
Call-ID: 6076913b1c39c212@REVMTEpG
CSeq: 1 INVITE
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, UPDATE
Max-Forwards: 70
Content-Type: application/sdp
Content-Length: xxxx
Supported: from-change

v=0
o=- 1181923068 1181923196 IN IP4 ua1.example.com
s=example1
c=IN IP4 ua1.example.com
a=setup:actpass
a=fingerprint: SHA-1 \
  4A:AD:B9:B1:3F:82:18:3B:54:02:12:DF:3E:5D:49:6B:19:E5:7C:AB
t=0 0
m=audio 6056 RTP/AVP 0
a=sendrecv
a=tcap:1 UDP/TLS/RTP/SAVP RTP/AVP
a=pcfg:1 t=1
```

Message (2): INVITE Proxy -> Bob

This shows the INVITE being relayed to Bob from Alice (and Bob's) proxy. Note that Alice's proxy has inserted an Identity and Identity-Info header. This example only shows one element for both proxies for the purposes of simplification. Bob verifies the identity provided with the INVITE.

```
INVITE sip:bob@ua2.example.com SIP/2.0
To: <sip:bob@example.com>
From: "Alice"<sip:alice@example.com>;tag=843c7b0b
Via: SIP/2.0/TLS proxy.example.com;branch=z9hG4bK-0e53sadfkasldk
Via: SIP/2.0/TLS ua1.example.com;branch=z9hG4bK-0e53sadfkasldkfj
Record-Route: <sip:proxy.example.com;lr>
Contact: <sip:alice@ua1.example.com>
Call-ID: 6076913b1c39c212@REVMTEpG
CSeq: 1 INVITE
Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, UPDATE
Max-Forwards: 69
Identity: CyI4+nAkHrH3ntmaxgr01TMxTmtjP7MASwliNRdupRI1vpkXRvZXx1ja9k
          3W+v1PDsy32MaqZi0M5WfEkXxbgTnPYW0jIoK8HMyY1VT7egt0kk4XrKFC
          HYWGCl0nB2sNsM9CG4hq+YJZTMaSROoMUBhikVIjnQ8ykeD6UXNOyfI=
Identity-Info: https://example.com/cert
Content-Type: application/sdp
Content-Length: xxxx
Supported: from-change

v=0
o=- 1181923068 1181923196 IN IP4 ua1.example.com
s=example1
c=IN IP4 ua1.example.com
a=setup:actpass
a=fingerprint: SHA-1 \
  4A:AD:B9:B1:3F:82:18:3B:54:02:12:DF:3E:5D:49:6B:19:E5:7C:AB
t=0 0
m=audio 6056 RTP/AVP 0
a=sendrecv
a=tcap:1 UDP/TLS/RTP/SAVP RTP/AVP
a=pcfg:1 t=1
```

Message (3): ClientHello Bob -> Alice

Assuming that Alice's identity is valid, Line 3 shows Bob sending a DTLS ClientHello(s) directly to Alice. In this case, two DTLS ClientHello messages would be sent to Alice: one to ua1.example.com:6056 for RTP and another to port 6057 for RTCP, but only one arrow is drawn for compactness of the figure.

Message (4): ServerHello+Certificate Alice -> Bob

Alice sends back a ServerHello, Certificate, and ServerHelloDone for both RTP and RTCP associations. Note that the same certificate is used for both the RTP and RTCP associations. If RTP/RTCP multiplexing [RFC5761] were being used only a single association would be required.

Message (5): Certificate Bob -> Alice

Bob sends a Certificate, ClientKeyExchange, CertificateVerify, change_cipher_spec, and Finished for both RTP and RTCP associations. Again note that Bob uses the same server certificate for both associations.

Message (6): Early Media Bob -> Alice

At this point, Bob can begin sending early media (RTP and RTCP) to Alice. Note that Alice can't yet trust the media since the fingerprint has not yet been received. This lack of trusted, secure media is indicated to Alice via the UA user interface.

Message (7): Finished Alice -> Bob

After Message 7 is received by Bob, Alice sends change_cipher_spec and Finished.

Message (8): 200 OK Bob -> Alice

When Bob answers the call, Bob sends a 200 OK SIP message that contains the fingerprint for Bob's certificate. Bob signals the actual transport protocol configuration of SRTP over DTLS in the acfg parameter.

```
SIP/2.0 200 OK
To: <sip:bob@example.com>;tag=6418913922105372816
From: "Alice" <sip:alice@example.com>;tag=843c7b0b
Via: SIP/2.0/TLS proxy.example.com:5061;branch=z9hG4bK-0e53sadfkasldk
Via: SIP/2.0/TLS ua1.example.com;branch=z9hG4bK-0e53sadfkasldkfj
Record-Route: <sip:proxy.example.com;lr>
Call-ID: 6076913b1c39c212@REVMTEpG
CSeq: 1 INVITE
Contact: <sip:bob@ua2.example.com>
Content-Type: application/sdp
Content-Length: xxxx
Supported: from-change
```

```
v=0
o=- 6418913922105372816 2105372818 IN IP4 ua2.example.com
s=example2
c=IN IP4 ua2.example.com
a=setup:active
a=fingerprint: SHA-1 \
  FF:FF:FF:B1:3F:82:18:3B:54:02:12:DF:3E:5D:49:6B:19:E5:7C:AB
t=0 0
m=audio 12000 UDP/TLS/RTP/SAVP 0
a=acfg:1 t=1
```

Message (9): 200 OK Proxy -> Alice

Alice receives the message from her proxy and validates the certificate presented in Message 7. The endpoint now shows Alice that the call as secured.

Message (10): RTP+RTCP Alice -> Bob

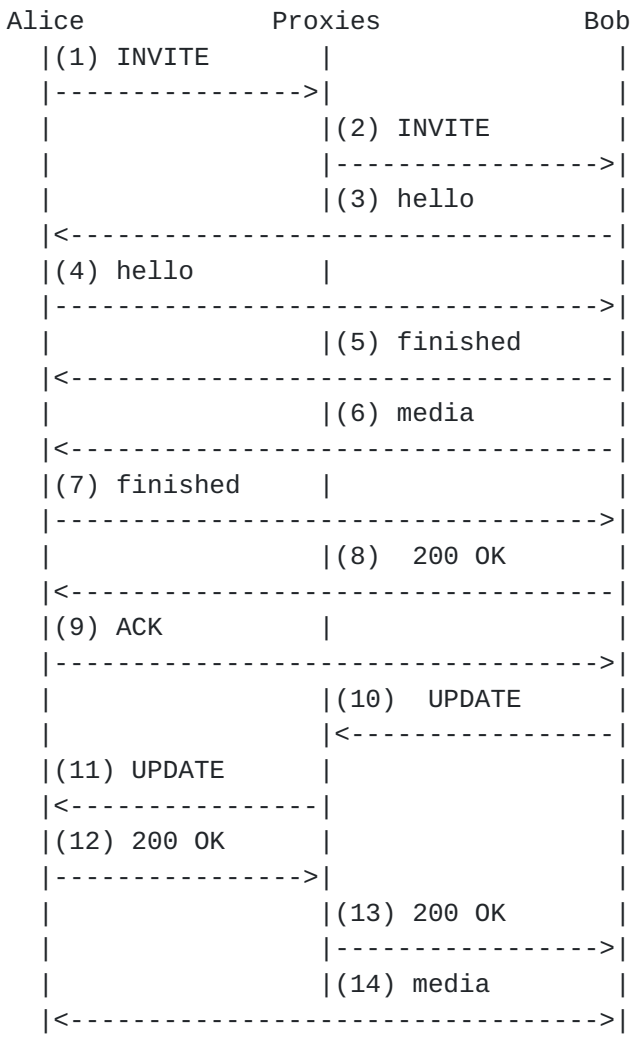
At this point, Alice can also start sending RTP and RTCP to Bob.

Message (11): ACK Alice -> Bob

Finally, Alice sends the SIP ACK to Bob.

7.2. Basic Message Flow with Connected Identity (RFC 4916)

The previous example did not show the use of RFC 4916 for connected identity. The following example does:



The first 9 messages of this example are the same as before. However, Messages 10-13, performing the RFC 4916 UPDATE, are new.

Message (10): UPDATE Bob -> Proxy

Bob sends an RFC 4916 UPDATE towards Alice. This update contains his fingerprint. Bob's UPDATE contains the same session information that he provided in his 200 OK (Message 8). Note that in principle an UPDATE here can be used to modify session parameters. However, in this case it's being used solely to confirm the fingerprint.

```
UPDATE sip:alice@ua1.example.com SIP/2.0
Via: SIP/2.0/TLS ua2.example.com;branch=z9hG4bK-0e53sadfkasldkfj
To: "Alice" <sip:alice@example.com>;tag=843c7b0b
From <sip:bob@example.com>;tag=6418913922105372816
Route: <sip:proxy.example.com;lr>
Call-ID: 6076913b1c39c212@REVMTEpG
CSeq: 2 UPDATE
Contact: <sip:ua2.example.com>
Content-Type: application/sdp
Content-Length: xxxx
Supported: from-change
Max-Forwards: 70

v=0
o=- 6418913922105372816 2105372818 IN IP4 ua2.example.com
s=example2
c=IN IP4 ua2.example.com
a=setup:active
a=fingerprint: SHA-1 \
  FF:FF:FF:B1:3F:82:18:3B:54:02:12:DF:3E:5D:49:6B:19:E5:7C:AB
t=0 0
m=audio 12000 UDP/TLS/RTP/SAVP 0
a=acfg:1 t=1
```

Message (11): UPDATE Proxy -> Alice

This shows the UPDATE being relayed to Alice from Bob (and Alice's proxy). Note that Bob's proxy has inserted an Identity and Identity-Info header. As above, we only show one element for both proxies for purposes of simplification. Alice verifies the identity provided. (Note: the actual identity signatures here are incorrect and provided merely as examples.)

```
UPDATE sip:alice@ua1.example.com SIP/2.0
Via: SIP/2.0/TLS proxy.example.com;branch=z9hG4bK-0e53sadfkasldkfj
Via: SIP/2.0/TLS ua2.example.com;branch=z9hG4bK-0e53sadfkasldkfj
To: "Alice" <sip:alice@example.com>;tag=843c7b0b
From <sip:bob@example.com>;tag=6418913922105372816
Call-ID: 6076913b1c39c212@REVMTEpG
CSeq: 2 UPDATE
Contact: <sip:bob@ua2.example.com>
Content-Type: application/sdp
Content-Length: xxxx
Supported: from-change
Max-Forwards: 69
Identity: CyI4+nAkHrH3ntmaxgr01TMxTmtjP7MASwliNRdupRI1vpkXRvZXx1ja9k
          3W+v1PDsy32MaqZi0M5WfEkXxbgTnPYW0jIoK8HMyY1VT7egt0kk4XrKFC
          HYWGCl0nB2sNsM9CG4hq+YJZTMaSROoMUBhikVIjnQ8ykeD6UXNOyfI=
Identity-Info: https://example.com/cert

v=0
o=- 6418913922105372816 2105372818 IN IP4 ua2.example.com
s=example2
c=IN IP4 ua2.example.com
a=setup:active
a=fingerprint: SHA-1 \
  FF:FF:FF:B1:3F:82:18:3B:54:02:12:DF:3E:5D:49:6B:19:E5:7C:AB
t=0 0
m=audio 12000 UDP/TLS/RTP/SAVP 0
a=acfg:1 t=1
```

Message (12): 200 OK Alice -> Bob

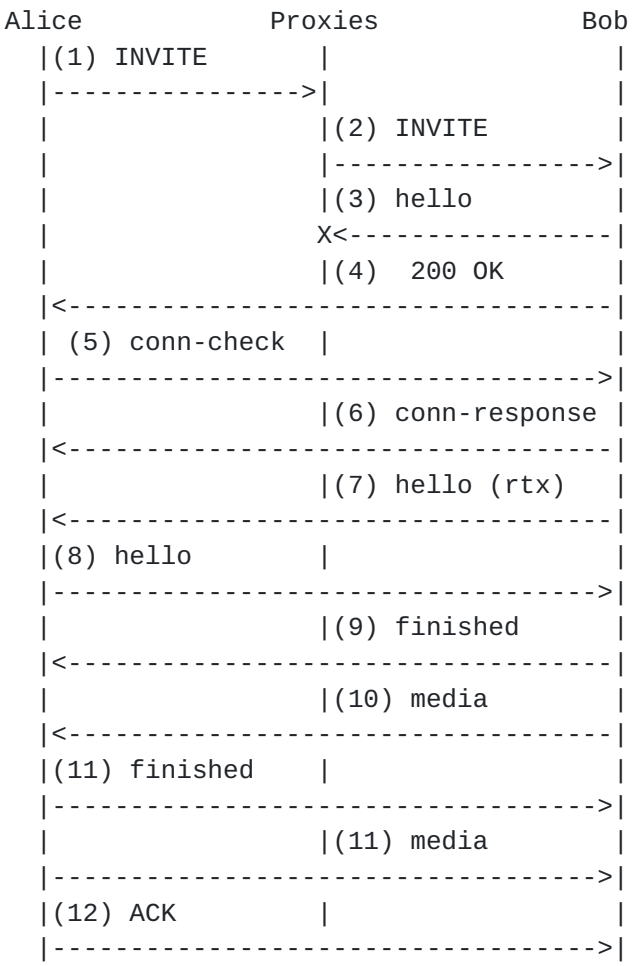
This shows Alice's 200 OK response to Bob's UPDATE. Because Bob has merely sent the same session parameters he sent in his 200 OK, Alice can simply replay her view of the session parameters as well.

```
SIP/2.0 200 OK
To: "Alice" <sip:alice@example.com>;tag=843c7b0b
From <sip:bob@example.com>;tag=6418913922105372816
Via: SIP/2.0/TLS proxy.example.com;branch=z9hG4bK-0e53sadfkasldkfj
Via: SIP/2.0/TLS ua2.example.com;branch=z9hG4bK-0e53sadfkasldkfj
Call-ID: 6076913b1c39c212@REVMTEpG
CSeq: 2 UPDATE
Contact: <sip:bob@ua2.example.com>
Content-Type: application/sdp
Content-Length: xxxx
Supported: from-change

v=0
o=- 1181923068 1181923196 IN IP4 ua2.example.com
s=example1
c=IN IP4 ua2.example.com
a=setup:actpass
a=fingerprint: SHA-1 \
  4A:AD:B9:B1:3F:82:18:3B:54:02:12:DF:3E:5D:49:6B:19:E5:7C:AB
t=0 0
m=audio 6056 RTP/AVP 0
a=sendrecv
a=tcap:1 UDP/TLS/RTP/SAVP RTP/AVP
a=pcfg:1 t=1
```

7.3. Basic Message Flow with STUN Check for NAT Case

In the previous examples, the DTLS handshake has already completed by the time Alice receives Bob's 200 OK (8). Therefore, no STUN check is sent. However, if Alice had a NAT, then Bob's ClientHello might get blocked by that NAT, in which case Alice would send the STUN check described in Section 6.7.1 upon receiving the 200 OK, as shown below:



The messages here are the same as in the first example (for simplicity this example omits an UPDATE), with the following three new messages:

Message (5): STUN connectivity-check Alice -> Bob

Section 6.7.1 describes an approach to avoid an SBC interaction issue where the endpoints do not support ICE. Alice (the passive endpoint) sends a STUN connectivity check to Bob. This opens a pinhole in Alice's NAT/firewall.

Message (6): STUN connectivity-check response Bob -> Alice

Bob (the active endpoint) sends a response to the STUN connectivity check (Message 3) to Alice. This tells Alice that her connectivity check has succeeded and she can stop the retransmit state machine.

Message (7): Hello (retransmit) Bob -> Alice

Bob retransmits his DTLS ClientHello, which now passes through the pinhole created in Alice's firewall. At this point, the DTLS handshake proceeds as before.

8. Security Considerations

DTLS or TLS media signaled with SIP requires a way to ensure that the communicating peers' certificates are correct.

The standard TLS/DTLS strategy for authenticating the communicating parties is to give the server (and optionally the client) a PKIX [RFC5280] certificate. The client then verifies the certificate and checks that the name in the certificate matches the server's domain name. This works because there are a relatively small number of servers with well-defined names; a situation that does not usually occur in the VoIP context.

The design described in this document is intended to leverage the authenticity of the signaling channel (while not requiring confidentiality). As long as each side of the connection can verify the integrity of the SDP received from the other side, then the DTLS handshake cannot be hijacked via a man-in-the-middle attack. This integrity protection is easily provided by the caller to the callee (see Alice to Bob in Section 7) via the SIP Identity [RFC4474] mechanism. Other mechanisms, such as the S/MIME mechanism described in RFC 3261, or perhaps future mechanisms yet to be defined could also serve this purpose.

While this mechanism can still be used without such integrity mechanisms, the security provided is limited to defense against passive attack by intermediaries. An active attack on the signaling plus an active attack on the media plane can allow an attacker to attack the connection (R-SIG-MEDIA in the notation of [RFC5479]).

8.1. Responder Identity

SIP Identity does not support signatures in responses. Ideally, Alice would want to know that Bob's SDP had not been tampered with and who it was from so that Alice's User Agent could indicate to Alice that there was a secure phone call to Bob. [RFC4916] defines an approach for a UA to supply its identity to its peer UA, and for this identity to be signed by an authentication service. For example, using this approach, Bob sends an answer, then immediately follows up with an UPDATE that includes the fingerprint and uses the SIP Identity mechanism to assert that the message is from Bob@example.com. The downside of this approach is that it requires

the extra round trip of the UPDATE. However, it is simple and secure even when not all of the proxies are trusted. In this example, Bob only needs to trust his proxy. Offerers SHOULD support this mechanism and answerers SHOULD use it.

In some cases, answerers will not send an UPDATE and in many calls, some media will be sent before the UPDATE is received. In these cases, no integrity is provided for the fingerprint from Bob to Alice. In this approach, an attacker that was on the signaling path could tamper with the fingerprint and insert themselves as a man-in-the-middle on the media. Alice would know that she had a secure call with someone, but would not know if it was with Bob or a man-in-the-middle. Bob would know that an attack was happening. The fact that one side can detect this attack means that in most cases where Alice and Bob both wish for the communications to be encrypted, there is not a problem. Keep in mind that in any of the possible approaches, Bob could always reveal the media that was received to anyone. We are making the assumption that Bob also wants secure communications. In this do nothing case, Bob knows the media has not been tampered with or intercepted by a third party and that it is from Alice@example.com. Alice knows that she is talking to someone and that whoever that is has probably checked that the media is not being intercepted or tampered with. This approach is certainly less than ideal but very usable for many situations.

8.2. SIPS

If SIP Identity is not used, but the signaling is protected by SIPS, the security guarantees are weaker. Some security is still provided as long as all proxies are trusted. This provides integrity for the fingerprint in a chain-of-trust security model. Note, however, that if the proxies are not trusted, then the level of security provided is limited.

8.3. S/MIME

RFC 3261 [RFC3261] defines an S/MIME security mechanism for SIP that could be used to sign that the fingerprint was from Bob. This would be secure.

8.4. Continuity of Authentication

One desirable property of a secure media system is to provide continuity of authentication: being able to ensure cryptographically that you are talking to the same person as before. With DTLS, continuity of authentication is achieved by having each side use the same public key/self-signed certificate for each connection (at least with a given peer entity). It then becomes possible to cache the

credential (or its hash) and verify that it is unchanged. Thus, once a single secure connection has been established, an implementation can establish a future secure channel even in the face of future insecure signaling.

In order to enable continuity of authentication, implementations SHOULD attempt to keep a constant long-term key. Verifying implementations SHOULD maintain a cache of the key used for each peer identity and alert the user if that key changes.

8.5. Short Authentication String

An alternative available to Alice and Bob is to use human speech to verify each other's identity and then to verify each other's fingerprints also using human speech. Assuming that it is difficult to impersonate another's speech and seamlessly modify the audio contents of a call, this approach is relatively safe. It would not be effective if other forms of communication were being used such as video or instant messaging. DTLS supports this mode of operation. The minimal secure fingerprint length is around 64 bits.

ZRTP [AVT-ZRTP] includes Short Authentication String (SAS) mode in which a unique per-connection bitstring is generated as part of the cryptographic handshake. The SAS can be as short as 25 bits and so is somewhat easier to read. DTLS does not natively support this mode. Based on the level of deployment interest, a TLS extension [RFC5246] could provide support for it. Note that SAS schemes only work well when the endpoints recognize each other's voices, which is not true in many settings (e.g., call centers).

8.6. Limits of Identity Assertions

When RFC 4474 is used to bind the media keying material to the SIP signaling, the assurances about the provenance and security of the media are only as good as those for the signaling. There are two important cases to note here:

- RFC 4474 assumes that the proxy with the certificate "example.com" controls the namespace "example.com". Therefore, the RFC 4474 authentication service that is authoritative for a given namespace can control which user is assigned each name. Thus, the authentication service can take an address formerly assigned to Alice and transfer it to Bob. This is an intentional design feature of RFC 4474 and a direct consequence of the SIP namespace architecture.

- When phone number URIs (e.g., 'sip:+17005551008@chicago.example.com' or 'sip:+17005551008@chicago.example.com;user=phone') are used, there is no structural reason to trust that the domain name is authoritative for a given phone number, although individual proxies and UAs may have private arrangements that allow them to trust other domains. This is a structural issue in that Public Switched Telephone Network (PSTN) elements are trusted to assert their phone number correctly and that there is no real concept of a given entity being authoritative for some number space.

In both of these cases, the assurances that DTLS-SRTP provides in terms of data origin integrity and confidentiality are necessarily no better than SIP provides for signaling integrity when RFC 4474 is used. Implementors should therefore take care not to indicate misleading peer identity information in the user interface. That is, if the peer's identity is sip:+17005551008@chicago.example.com, it is not sufficient to display that the identity of the peer as +17005551008, unless there is some policy that states that the domain "chicago.example.com" is trusted to assert the E.164 numbers it is asserting. In cases where the UA can determine that the peer identity is clearly an E.164 number, it may be less confusing to simply identify the call as encrypted but to an unknown peer.

In addition, some middleboxes (back-to-back user agents (B2BUAs) and Session Border Controllers) are known to modify portions of the SIP message that are included in the RFC 4474 signature computation, thus breaking the signature. This sort of man-in-the-middle operation is precisely the sort of message modification that RFC 4474 is intended to detect. In cases where the middlebox is itself permitted to generate valid RFC 4474 signatures (e.g., it is within the same administrative domain as the RFC 4474 authentication service), then it may generate a new signature on the modified message. Alternately, the middlebox may be able to sign with some other identity that it is permitted to assert. Otherwise, the recipient cannot rely on the RFC 4474 Identity assertion and the UA MUST NOT indicate to the user that a secure call has been established to the claimed identity. Implementations that are configured to only establish secure calls SHOULD terminate the call in this case.

If SIP Identity or an equivalent mechanism is not used, then only protection against attackers who cannot actively change the signaling is provided. While this is still superior to previous mechanisms, the security provided is inferior to that provided if integrity is provided for the signaling.

8.7. Third-Party Certificates

This specification does not depend on the certificates being held by endpoints being independently verifiable (e.g., being issued by a trusted third party). However, there is no limitation on such certificates being used. Aside from the difficulty of obtaining such certificates, it is not clear what identities those certificates would contain -- RFC 3261 specifies a convention for S/MIME certificates that could also be used here, but that has seen only minimal deployment. However, in closed or semi-closed contexts where such a convention can be established, third-party certificates can reduce the reliance on trusting even proxies in the endpoint's domains.

8.8. Perfect Forward Secrecy

One concern about the use of a long-term key is that compromise of that key may lead to compromise of past communications. In order to prevent this attack, DTLS supports modes with Perfect Forward Secrecy using Diffie-Hellman and Elliptic-Curve Diffie-Hellman cipher suites. When these modes are in use, the system is secure against such attacks. Note that compromise of a long-term key may still lead to future active attacks. If this is a concern, a backup authentication channel, such as manual fingerprint establishment or a short authentication string, should be used.

9. Acknowledgments

Cullen Jennings contributed substantial text and comments to this document. This document benefited from discussions with Francois Audet, Nagendra Modadugu, and Dan Wing. Thanks also for useful comments by Flemming Andreasen, Jonathan Rosenberg, Rohan Mahy, David McGrew, Miguel Garcia, Steffen Fries, Brian Stucker, Robert Gilman, David Oran, and Peter Schneider.

We would like to thank Thomas Belling, Guenther Horn, Steffen Fries, Brian Stucker, Francois Audet, Dan Wing, Jari Arkko, and Vesa Lehtovirta for their input regarding traversal of SBCs.

10. References

10.1. Normative References

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

[RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, June 2002.

[RFC3264] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model with Session Description Protocol (SDP)", RFC 3264, June 2002.

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, May 2008.

[RFC3323] Peterson, J., "A Privacy Mechanism for the Session Initiation Protocol (SIP)", RFC 3323, November 2002.

[RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, July 2003.

[RFC4145] Yon, D. and G. Camarillo, "TCP-Based Media Transport in the Session Description Protocol (SDP)", RFC 4145, September 2005.

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

[RFC4474] Peterson, J. and C. Jennings, "Enhancements for Authenticated Identity Management in the Session Initiation Protocol (SIP)", RFC 4474, August 2006.

[RFC4566] Handley, M., Jacobson, V., and C. Perkins, "SDP: Session Description Protocol", RFC 4566, July 2006.

[RFC4572] Lennox, J., "Connection-Oriented Media Transport over the Transport Layer Security (TLS) Protocol in the Session Description Protocol (SDP)", RFC 4572, July 2006.

[RFC5389] Rosenberg, J., Mahy, R., Matthews, P., and D. Wing, "Session Traversal Utilities for NAT (STUN)", RFC 5389, October 2008.

10.2. Informative References

[RFC4571] Lazzaro, J., "Framing Real-time Transport Protocol (RTP) and RTP Control Protocol (RTCP) Packets over Connection-Oriented Transport", RFC 4571, July 2006.

[RFC3325] Jennings, C., Peterson, J., and M. Watson, "Private Extensions to the Session Initiation Protocol (SIP) for Asserted Identity within Trusted Networks", RFC 3325, November 2002.

[RFC5245] Rosenberg, J., "Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal for Offer/Answer Protocols", RFC 5245, April 2010.

[RFC4567] Arkko, J., Lindholm, F., Naslund, M., Norrman, K., and E. Carrara, "Key Management Extensions for Session Description Protocol (SDP) and Real Time Streaming Protocol (RTSP)", RFC 4567, July 2006.

[RFC4568] Andreasen, F., Baugher, M., and D. Wing, "Session Description Protocol (SDP) Security Descriptions for Media Streams", RFC 4568, July 2006.

[AVT-ZRTP] Zimmermann, P., Johnston, A., and J. Callas, "ZRTP: Media Path Key Agreement for Secure RTP", Work in Progress, March 2009.

[SRTP-EKT] McGrew, D., Andreasen, F., and L. Dondeti, "Encrypted Key Transport for Secure RTP", Work in Progress, March 2009.

[RFC5764] McGrew, D. and E. Rescorla, "Datagram Transport Layer Security (DTLS) Extension to Establish Keys for Secure Real-time Transport Protocol (SRTP)", RFC 5764, May 2010.

[RFC5479] Wing, D., Fries, S., Tschofenig, H., and F. Audet, "Requirements and Analysis of Media Security Management Protocols", RFC 5479, March 2009.

[MMUSIC-SDP]
Andreasen, F., "SDP Capability Negotiation", Work in Progress, February 2010.

[RFC5761] Perkins, C. and M. Westerlund, "Multiplexing RTP Data and Control Packets on a Single Port", RFC 5761, April 2010.

[RFC3262] Rosenberg, J. and H. Schulzrinne, "Reliability of Provisional Responses in Session Initiation Protocol (SIP)", RFC 3262, June 2002.

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

[RFC4916] Elwell, J., "Connected Identity in the Session Initiation Protocol (SIP)", RFC 4916, June 2007.

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

[RFC3830] Arkko, J., Carrara, E., Lindholm, F., Naslund, M., and K. Norrman, "MIKEY: Multimedia Internet KEYing", RFC 3830, August 2004.

[SIPPING-SRTP]
Wing, D., Audet, F., Fries, S., Tschofenig, H., and A. Johnston, "Secure Media Recording and Transcoding with the Session Initiation Protocol", Work in Progress, October 2008.

[KEY-TRANSPORT]
Wing, D., "DTLS-SRTP Key Transport (KTR)", Work in Progress, March 2009.

[MMUSIC-MEDIA]
Stucker, B. and H. Tschofenig, "Analysis of Middlebox Interactions for Signaling Protocol Communication along the Media Path", Work in Progress, March 2009.

[RFC5767] Munakata, M., Schubert, S., and T. Ohba, "User-Agent-Driven Privacy Mechanism for SIP", RFC 5767, April 2010.

Appendix A. Requirements Analysis

[RFC5479] describes security requirements for media keying. This section evaluates this proposal with respect to each requirement.

A.1. Forking and Retargeting (R-FORK-RETARGET, R-BEST-SECURE, R-DISTINCT)

In this document, the SDP offer (in the INVITE) is simply an advertisement of the capability to do security. This advertisement does not depend on the identity of the communicating peer, so forking and retargeting work when all the endpoints will do SRTP. When a mix of SRTP and non-SRTP endpoints are present, we use the SDP capabilities mechanism currently being defined [MMUSIC-SDP] to transparently negotiate security where possible. Because DTLS establishes a new key for each session, only the entity with which the call is finally established gets the media encryption keys (R3).

A.2. Distinct Cryptographic Contexts (R-DISTINCT)

DTLS performs a new DTLS handshake with each endpoint, which establishes distinct keys and cryptographic contexts for each endpoint.

A.3. Reusage of a Security Context (R-REUSE)

DTLS allows sessions to be resumed with the 'TLS session resumption' functionality. This feature can be used to lower the amount of cryptographic computation that needs to be done when two peers re-initiate the communication. See [RFC5764] for more on session resumption in this context.

A.4. Clipping (R-AVOID-CLIPPING)

Because the key establishment occurs in the media plane, media need not be clipped before the receipt of the SDP answer. Note, however, that only confidentiality is provided until the offerer receives the answer: the answerer knows that they are not sending data to an attacker but the offerer cannot know that they are receiving data from the answerer.

A.5. Passive Attacks on the Media Path (R-PASS-MEDIA)

The public key algorithms used by DTLS cipher suites, such as RSA, Diffie-Hellman, and Elliptic Curve Diffie-Hellman, are secure against passive attacks.

A.6. Passive Attacks on the Signaling Path (R-PASS-SIG)

DTLS provides protection against passive attacks by adversaries on the signaling path since only a fingerprint is exchanged using SIP signaling.

A.7. (R-SIG-MEDIA, R-ACT-ACT)

An attacker who controls the media channel but not the signaling channel can perform a MITM attack on the DTLS handshake but this will change the certificates that will cause the fingerprint check to fail. Thus, any successful attack requires that the attacker modify the signaling messages to replace the fingerprints.

If RFC 4474 Identity or an equivalent mechanism is used, an attacker who controls the signaling channel at any point between the proxies performing the Identity signatures cannot modify the fingerprints without invalidating the signature. Thus, even an attacker who controls both signaling and media paths cannot successfully attack the media traffic. Note that the channel between the UA and the authentication service MUST be secured and the authentication service MUST verify the UA's identity in order for this mechanism to be secure.

Note that an attacker who controls the authentication service can impersonate the UA using that authentication service. This is an intended feature of SIP Identity -- the authentication service owns the namespace and therefore defines which user has which identity.

A.8. Binding to Identifiers (R-ID-BINDING)

When an end-to-end mechanism such as SIP-Identity [RFC4474] and SIP-Connected-Identity [RFC4916] or S/MIME are used, they bind the endpoint's certificate fingerprints to the From: address in the signaling. The fingerprint is covered by the Identity signature. When other mechanisms (e.g., SIPS) are used, then the binding is correspondingly weaker.

A.9. Perfect Forward Secrecy (R-PFS)

DTLS supports Diffie-Hellman and Elliptic Curve Diffie-Hellman cipher suites that provide PFS.

A.10. Algorithm Negotiation (R-COMPUTE)

DTLS negotiates cipher suites before performing significant cryptographic computation and therefore supports algorithm negotiation and multiple cipher suites without additional computational expense.

A.11. RTP Validity Check (R-RTP-VALID)

DTLS packets do not pass the RTP validity check. The first byte of a DTLS packet is the content type and all current DTLS content types have the first two bits set to zero, resulting in a version of zero; thus, failing the first validity check. DTLS packets can also be distinguished from STUN packets. See [RFC5764] for details on demultiplexing.

A.12. Third-Party Certificates (R-CERTS, R-EXISTING)

Third-party certificates are not required because signaling (e.g., [RFC4474]) is used to authenticate the certificates used by DTLS. However, if the parties share an authentication infrastructure that is compatible with TLS (third-party certificates or shared keys) it can be used.

A.13. FIPS 140-2 (R-FIPS)

TLS implementations already may be FIPS 140-2 approved and the algorithms used here are consistent with the approval of DTLS and DTLS-SRTP.

A.14. Linkage between Keying Exchange and SIP Signaling (R-ASSOC)

The signaling exchange is linked to the key management exchange using the fingerprints carried in SIP and the certificates are exchanged in DTLS.

A.15. Denial-of-Service Vulnerability (R-DOS)

DTLS offers some degree of Denial-of-Service (DoS) protection as a built-in feature (see Section 4.2.1 of [RFC4347]).

A.16. Crypto-Agility (R-AGILITY)

DTLS allows cipher suites to be negotiated and hence new algorithms can be incrementally deployed. Work on replacing the fixed MD5/SHA-1 key derivation function is ongoing.

A.17. Downgrading Protection (R-DOWNGRADE)

DTLS provides protection against downgrading attacks since the selection of the offered cipher suites is confirmed in a later stage of the handshake. This protection is efficient unless an adversary is able to break a cipher suite in real-time. RFC 4474 is able to prevent an active attacker on the signaling path from downgrading the call from SRTP to RTP.

A.18. Media Security Negotiation (R-NEGOTIATE)

DTLS allows a User Agent to negotiate media security parameters for each individual session.

A.19. Signaling Protocol Independence (R-OTHER-SIGNALING)

The DTLS-SRTP framework does not rely on SIP; every protocol that is capable of exchanging a fingerprint and the media description can be secured.

A.20. Media Recording (R-RECORDING)

An extension, see [SIPPING-SRTP], has been specified to support media recording that does not require intermediaries to act as an MITM.

When media recording is done by intermediaries, then they need to act as an MITM.

A.21. Interworking with Intermediaries (R-TRANSCODER)

In order to interface with any intermediary that transcodes the media, the transcoder must have access to the keying material and be treated as an endpoint for the purposes of this document.

A.22. PSTN Gateway Termination (R-PSTN)

The DTLS-SRTP framework allows the media security to terminate at a PSTN gateway. This does not provide end-to-end security, but is consistent with the security goals of this framework because the gateway is authorized to speak for the PSTN namespace.

A.23. R-ALLOW-RTP

DTLS-SRTP allows RTP media to be received by the calling party until SRTP has been negotiated with the answerer, after which SRTP is preferred over RTP.

A.24. R-HERFP

The Heterogeneous Error Response Forking Problem (HERFP) is not applicable to DTLS-SRTP since the key exchange protocol will be executed along the media path and hence error messages are communicated along this path and proxies do not need to progress them.

Authors' Addresses

Jason Fischl
Skype, Inc.
2145 Hamilton Ave.
San Jose, CA 95135
USA

Phone: +1-415-692-1760
EMail: jason.fischl@skype.net

Hannes Tschofenig
Nokia Siemens Networks
Linnoitustie 6
Espoo, 02600
Finland

Phone: +358 (50) 4871445
EMail: Hannes.Tschofenig@gmx.net
URI: http://www.tschofenig.priv.at

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

EMail: ekr@rtfm.com