Internet Engineering Task Force
MMUSIC Working Group
INTERNET-DRAFT
Expires: October 2002

J. Arkko
E. Carrara
F. Lindholm
M. Naslund
K. Norrman
Ericsson

April, 2002

Key Management Extensions for SDP and RTSP
<draft-ietf-mmusic-kmgmt-ext-04.txt>

Status of this memo

Abstract

This document defines general extensions for SDP and RTSP to carry the security information needed by a key management protocol, in order to secure the media. These extensions are presented as a framework, to be used by one or more key management protocols. As such, its use is meaningful only when it is completed by the key management protocol in use.

General guidelines are also given on how the framework should be used together with SIP and RTSP.

TABLE OF CONTENTS

1. Introduction

There has recently been work to define a security framework for the protection of real-time applications running over RTP, [SRTP]. However, a security protocol needs a key management infrastructure to exchange keys and security parameters, managing and refreshing keys, etc.

A key management protocol is executed prior to the security protocol execution. The key management protocol's main goal is to, in a secure and reliable way, establish a so called security association for the security protocol. This includes one or several cryptographic keys and a set of necessary parameters for the security protocol, e.g., cipher and authentication algorithm to be used. The key management protocol has similarities with, e.g., SIP [SIP] and RTSP [RTSP] in the sense that it negotiates necessary information in order to be able to setup the session.

The focus in the following sections is to describe SDP attribute extensions and RTSP header extensions to support key management, and a possible integration within SIP and RTSP. A framework is therefore described in the following. Such a framework will need to be completed by one or more key management protocols, to describe how the framework is used, e.g. which is the data to be carried in the extensions.

Some of the motivations to create a framework with the possibility to include the key management in the session establishment are:

* Just as the codec information is a description of how to encode and decode the audio (or video) stream, the key management data is a description of how to encrypt and decrypt the data.

* The possibility to negotiate the security for the entire multimedia session at the same time.

* The knowledge of the media at the session establishment makes it easy to tie the key management to the multimedia sessions.

* This approach may be more efficient than setting up the security later, as that approach might force extra roundtrips, possibly also a separate set-up for each stream, hence implying more delay to the actual setup of the media session.

Currently in SDP [SDPnew], one field exists to transport keys, i.e. the "key=" field. However, this is not enough for a key management protocol. The approach here is to use and extend the SDP description to transport the key management offer/answer and also to associate it with the media sessions. SIP uses the offer/answer model [OAM] whereby extensions to SDP will be enough. An extra RTSP header is also defined.

1.1. Notational 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 RFC-2119.

2. Extensions to SDP and RTSP

This section describes common attributes that are to be included in an SDP description or in an RTSP header when an integrated key management protocol is used. The attribute values MUST follow the general SDP or RTSP guideline.

For the SDP description, the key management attributes may be defined at session level (i.e. before the media descriptor lines) and/or at media level. If the key management attributes are defined at media level, they will only apply to that specific media. If the key management attributes are defined at both session and media level, the media level definition overrides the session level definition for that specific media.

The following SDP attribute is defined:

key-mgmt:<name> <opaque-data>

<name> is the name of the key management protocol and the opaque-data is a field to transport the key management protocol data. The key management protocol data contains the necessary information to establish the security protocol, e.g., keys and cryptographic parameters. All parameters and keys are protected by the key management. Note that if the key management protocol fails, e.g., the receiver does not accept any of the proposed security parameters, or simply does not understand the key management protocol, the security setup will fail. Consequently, it is impossible to establish a secure session. This is very similar to the normal SIP/SDP behavior: if the sender supports codecs which are not supported by the receiver, it will be problematic to set up a session.

2.1. SDP Extensions

This section provides an Augmented Backus-Naur Form (ABNF) grammar (as used in [SDPnew]) for the key management extensions to SDP.

Note that the new definitions are compliant with the definition of an attribute field, i.e.

```
attribute      = (att-field ":" att-value) | att-field
```

One new attribute for SDP is defined:

```
key-mgmt  = "key-mgmt:" prtcl-name keymgmt-data

prtcl-name     = non-ws-string
                 ; e.g. "MIKEY"

keymgmt-data   = byte-string
```

where non-ws-string and byte-string are as defined in SDP [SDPnew].

2.2. RTSP Extensions

To support the needed attribute described, the following RTSP header is defined:

KeyMgmt _ "keymgmt" ":" "prot" "=" token ";" "data" "=" quoted-string

token and quoted-string are as defined in the RTSP specification [RTSP].

The KeyMgmt header should be possible to use in both request and response messages of the following methods:

* DESCRIBE

* ANNOUNCE
* SETUP

3. Usage with SIP and RTSP

This section gives recommendations of how/when to include the defined key management attribute when SIP and/or RTSP are used together with SDP.

Some general requirements are set on a key management protocol (and

its API) when used within SIP and RTSP:

* It MUST be possible to execute the key management protocol in at most one roundtrip in case the answerer accepts the offer.

* It MUST be possible, using the key management API, to receive a valid offer/answer and whether the provided offer was accepted or not.

Today, the MIKEY protocol [MIKEY] has adopted the key management extensions to work together with SIP and RTSP. Other protocols may use the described attribute and header, e.g. Kerberos [KERB].

3.1. General SDP processing

When an SDP message is created, the following procedure should be applied:

* The identifier of the key management protocol used (e.g. MIKEY or Kerberos) is put in the prtcl-name field.

* The keymgmt-data field is created by the data received from the key management protocol API. The data may e.g. be a MIKEY message or Kerberos ticket.

A received SDP message that contains the key management attributes SHOULD process these attributes in the following manner:

* Detect the key management protocol used by checking the prtcl-name field in the key management attribute.

* Extract the key management data from the keymgmt-data field and call the key management protocol with the extracted data. Note that depending on key management protocol, some extra parameters might of course be requested, such as the source/destination network address/port(s) for the specified media.

* Depending on the outcome of the key management processing (i.e. whether it was accepted or not), the processing can proceed

according to normal processing (e.g. according to the offer/answer

model, see also Section 3.2).

If more than one key management protocol are supported, multiple instance of the key management attribute MAY be included in the initial offer, each transporting a different key management data. However, the offerer is RECOMMENDED to include only one of the protocols for a specific media. If the answerer cannot support the proposed protocol, it rejects the offer. Placing multiple key management offers in a single message would have the disadvantage that the message expands and the computational workload for the offerer will increase drastically. It might be acceptable to use a trial and error approach if the number of key management protocols supported are few. The possibility to support multiple key management protocols may introduce bidding down attacks. It is therefore important that the local policy considers this (e.g., only allows protocols that from a security point of view are equivalent, to be negotiated).

What can be done to increase the likelihood for a successful setup is to use a capability discovery mechanism (e.g., used in SIP). In this case, the key management protocols supported are expressed at session level without any data (i.e., a list of only the key-mgmt:<name> part is used).

```
v=0
o=alice 2891092738 2891092738 IN IP4 lost.somewhere.com
c=IN IP4 lost.somewhere.com
a=key-mgmt:mikey
a=key-mgmt:coolxchg
m=audio 0 RTP/SAVP 98
a=rtpmap:98 AMR/8000
m=video 0 RTP/SAVP 31 34
a=rtpmap:31 H261/90000
a=rtpmap:34 H263/90000
```

3.2. SIP usage

The offerer SHOULD include the key management data within an offer that contains the media description it should apply to. The answerer MUST check with the key management protocol if the attribute values are valid, and then obtain from the key management the data to include in the answer. If the offer is not accepted, the answerer returns a notification message and the offerer may go out with a new (different) offer, depending on the local security policy.

Re-keying can be handled as a new offer, i.e. a re-INVITE should be sent with the new proposed parameters. The answerer treats this as a new offer where the key management is the issue of change.

3.3. RTSP usage

RTSP does not use the offer/answer model, as SIP does. This causes some problems as it is not possible (without abusing RTSP) to send back an answer to the server (as the server will in most cases be the one initiating the security parameter exchange). To solve this, a new header has been introduced (Section 2.2). This also assumes that the key management also have some kind of binding to the media, so that the response to the server will be processed as required.

The processing of a key management header in RTSP should be done analogous of the SDP message processing. The initial key management message from a server should be sent to the client using SDP. When responding to this, the client uses the new RTSP header to send back an answer (included in the SETUP message). If the server retrieves a SETUP message in which it expects a key management message, but none is included, a 403 Forbidden is returned to the client.

The server may provide re-keying/updating facilities by sending a new key management message in an ANNOUNCE messages. The ANNOUNCE message contains an SDP message including the key management parameters. The response message is put in the new RTSP header in the response from the client to the server. Note that the ANNOUNCE messages MUST be supported if this feature are to be used.

3.4. Example scenarios

Example 1 (SIP)

A SIP call is taking place between Alice and Bob. Alice sends an Invite message consisting of the following offer:

```
v=0
o=alice 2891092738 2891092738 IN IP4 lost.somewhere.com
s=Cool stuff
e=alice@w-land.org
t=0 0
c=IN IP4 lost.somewhere.com
a=key-mgmt:mikey uiSDF9sdhs727ghsd/dhsoKkdOokdo7eWsnDSJD...
m=audio 49000 RTP/SAVP 98
a=rtpmap:98 AMR/8000
m=video 52230 RTP/SAVP 31
```

```
a=rtpmap:31 H261/90000
```

i.e. Alice proposes to set up one audio stream and one video stream that run over SRTP. To set up the security parameters for SRTP, she uses MIKEY. Note that MIKEY is negotiating the crypto suite for both streams (as it is placed at the session level).

Bob accepts the offer and sends an answer back to Alice:

```
v=0
o=bob 2891092897 2891092897 IN IP4 found.somewhere.com
s=Cool stuff
e=bob@null.org
t=0 0
c=IN IP4 found.somewhere.com
a=key-mgmt:mikey skaoqDeMkdwRW278HjKVB...
m=audio 49030 RTP/SAVP 98
a=rtpmap:98 AMR/8000
m=video 52230 RTP/SAVP 31
a=rtpmap:31 H261/90000
```

Example 2 (SDP)

This example shows how Alice would have done in the previous example if she wished to protect only the audio stream.

```
v=0
o=alice 2891092738 2891092738 IN IP4 lost.somewhere.com
s=Cool stuff
e=alice@w-land.org
t=0 0
c=IN IP4 lost.somewhere.com
m=audio 49000 RTP/SAVP 98
a=rtpmap:98 AMR/8000
a=key-mgmt:mikey uiSDF9sdhs727ghsd/dhsoKkdOokdo7eWsnDSJD...
m=video 52230 RTP/AVP 31
a=rtpmap:31 H261/90000
```

Note that even if the key management attribute is specified at

session level, the video part will not be affected by this (as a security profile is not used).

Example 3 (RTSP)

A client wants to set up a streaming session and requests a media description from the streaming server.

```
DESCRIBE rtsp://server.example.com/fizzle/foo RTSP/1.0
CSeq: 312
Accept: application/sdp
From: user@client.com
```

The server sends back an OK message including a SDP description.

```
RTSP/1.0 200 OK
CSeq: 312
Date: 23 Jan 1997 15:35:06 GMT
Content-Type: application/sdp
```



```
v=0
o=actionmovie 2891092738 2891092738 IN IP4 movie.somewhere.com
s=Action Movie
e=action@movie.somewhere.com
t=0 0
c=IN IP4 movie.somewhere.com
a=key-mgmt:mikey uiSDF9sdhs727ghsd/dhsoKkdOokdo7eWsnDSJD...
m=audio 0 RTP/SAVP 98
a=rtpmap:98 AMR/8000
control:rtsp://movie.somewhere.com/action/audio
m=video 0 RTP/SAVP 31
a=rtpmap:31 H261/90000
control:rtsp://movie.somewhere.com/action/video
```

The client is now ready to setup the sessions. It includes the key management data in the first message going back to the server (i.e. the SETUP message).

```
SETUP rtsp://movie.somewhere.com/action/audio RTSP/1.0
CSeq: 313
Transport: RTP/SAVP/UDP;unicast;client_port=3056-3057
```

```
keymgmt: prot=mikey; data="skaoqDeMkdwRW278HjKVB..."
```

The server processes the request including checking the validity of the key management header.

```
RTSP/1.0 200 OK
CSeq: 313
Session: 12345678
Transport: RTP/SAVP/UDP;unicast;client_port=3056-3057;
                      server_port=5000-5001
```

The RTSP then proceeds as usual (with e.g. a SETUP message for the video followed by a PLAY message).

4. Adding a Key management protocol

This framework can not be used with all key management protocols. The key management protocol needs to comply with the requirements described in Section 3. To be able to use a key management protocol with this framework, the following needs to be specified:

* the key management protocol name that should be used in the protocol name fields in both SDP and RTSP (e.g. "mikey" for MIKEY).

* the information the key management needs from SDP and RTSP (Section 3 gives a guideline of what SDP and RTSP needs from the key management). The exact API is implementation specific, but it SHOULD at least support to exchange the specified information.

The encoding of the data MUST be specified for each key management protocol and comply with the SDP and RTSP definitions. For most protocols, base64 encoding will be most appropriate.

5. Security Considerations

The nature of this document is to allow SDP and RTSP to support security of the media sessions. It is therefore not the intention of this document to describe possible security solution or to define possible security problems. The defined SDP and RTSP extensions are

not believed to introduce any new security risks to SDP and RTSP.

Note that the purpose of the key management fields is to provide information to secure the media streams. Under the assumption that the key management schemes are secure, the SDP can be passed along unprotected without affecting the key management, and the media streams will still be secure even if some attackers gained knowledge of the SDP contents.

However, if the SDP messages are not sent authenticated between the parties, it is possible for an active attacker to change attributes without being detected. As the key management protocol may (indirect) rely on some of the session information from SDP (e.g., address information), an attack on SDP may give indirect consequences on the key management. In general, it is therefore a good thing, not only to try to secure the session, but also to secure the session setup.

6. IANA Considerations

New attribute fields for SDP (see Section 2.1) and RTSP header are registered (see Section 2.2).

7. Conclusions

A security solution for real-time applications needs a key management infrastructure. Integrating the key management scheme with the session establishment protocol could be done efficiently in most of the scenarios. This draft proposes a framework that integrates a key management protocol (e.g., MIKEY) into SIP and RTSP, and which can be accompanied by different key management protocols. A set of new attributes and headers has been defined in SDP and RTSP to support this.

8. Acknowledgments

Thanks to: Rolf Blom, Magnus Westerlund, and the rest involved in the MMUSIC WG and the MSEC WG.

A special thanks to Joerg Ott and Colin Perkins.

9. Author's Addresses

Jari Arkko
Ericsson
02420 Jorvas Phone: +358 40 5079256
Finland Email: jari.arkko@ericsson.com

Elisabetta Carrara
Ericsson Research
SE-16480 Stockholm Phone: +46 8 50877040
Sweden EMail: elisabetta.carrara@era.ericsson.se

Fredrik Lindholm
Ericsson Research
SE-16480 Stockholm Phone: +46 8 58531705
Sweden EMail: fredrik.lindholm@era.ericsson.se

Mats Naslund
Ericsson Research
SE-16480 Stockholm Phone: +46 8 58533739
Sweden EMail: mats.naslund@era.ericsson.se

Karl Norrman
Ericsson Research
SE-16480 Stockholm Phone: +46 8 4044502
Sweden EMail: karl.norrman@era.ericsson.se

10. References

10.1. Normative References

[OAM] Rosenberg, J. and Schulzrinne, H., "An Offer/Answer Model with SDP", Internet Draft, IETF, Work in progress (MMUSIC).

[RTSP] Schulzrinne, H., Rao, A., and Lanphier, R., "Real Time Streaming Protocol (RTSP)", IETF, RFC 2326.

[SDPnew] Handley, M., Jacobson, V., and Perkins, C., "SDP: Session Description Protocol", Internet Draft, IETF, Work in progress (MMUSIC).

[SIP] Handley, M., Schulzrinne, H., Schooler, E., and Rosenberg, J., "SIP: Session Initiation Protocol", IETF, RFC 2543.

10.2. Informative References

[KERB] Kohl, J., Neuman, C., "The Kerberos Network Authentication Service (V5)", IETF, RFC 1510.

[MIKEY] Arkko, J., Carrara, E., Lindholm, F., Naslund, M., and Norrman, K., "MIKEY: Multimedia Internet KEYing", Internet Draft, IETF, Work in progress (MSEC).

[SRTP] Baugher, M., Blom, R., Carrara, E., McGrew, D., Naslund, M, Norrman, K., and Oran, D., "The Secure Real Time Transport Protocol", Internet Draft, IETF, Work in Progress (AVT).

This Internet-Draft expires in October 2002.