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

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

January, 2002

Key Management Extensions for SDP and RTSP
<draft-ietf-mmusic-kmgmt-ext-01.txt>

Status of this memo

Abstract

This document defines extensions for SDP and RTSP to carry the security information needed by a key management protocol, in order to secure the media stream. Indications are also given on how it 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.

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.

Some of the motivations 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 [SDP], 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. All 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 extensions were defined in a way to:

* give a minimal impact on current SDP implementations, i.e. only minimal modifications MUST be built into an existing SDP stack in order to support the key management.

* make it easy to use another key management protocol, or a new version, without having to redefine the attributes or add new ones.

The following set of attributes have been identified as necessary to support:

* key management protocol identifier, to indicate the key management protocol used ('keymgmt-prtcl')

* key management raw data field, to transport the key management protocol data ('keymgmt-data')

* in the case of SDP, an extra (optional) authentication attribute to be able to tie the key management data to the surrounding media definitions ('key-extra-auth').

2.1. SDP Extensions

This section provides an Augmented Backus-Naur Form (ABNF) grammar

(as used in [SDP]) 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
```

Three new attributes are defined, keymgmt-prtcl, keymgmt-data, and key-extra-auth.

```
keymgmt-prtcl  = "keymgmt-prot:" prtcl-name

prtcl-name     = 1*(safe)
                 ; e.g. "MIKEY"

keymgmt-data   = "keymgmt-data:" byte-string

key-extra-auth = "keymgmt-auth:" byte-string
```

where safe and byte-string are as defined in SDP [SDP].

2.2. RTSP Extensions

To support the three different attributes described, the following RTSP header is defined:

```
KeyMgmt    = "KeyMgmt" ":" [stream-url] protocol data

stream-url = "url" "=" url ";"

protocol   = "Prot" "=" token ";"

data       = "Data" "=" quoted-string
```

url, token and quoted-string are as defined in the RTSP specification [RTSP]. The url indicates the stream URL, which the parameters correspond to.

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

* DESCRIBE
* ANNOUNCE
* SETUP
* PLAY
* RECORD
* SET_PARAMETER
* GET_PARAMETER
* OPTIONS

3. Usage with SIP and RTSP

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

Some general requirements MUST be set on the key management protocol if it has to be suitable to work together with 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.

* The protocol MUST return, to SDP/RTSP, a valid answer whether the provided offer was accepted or not.

* There MUST be a possibility for the key management protocol to tie the media sessions to the negotiated parameters (i.e. an interface between the key management and the SDP and RTSP implementation MUST exist).

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

3.1. 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 should 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.

3.2. 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).

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 either the SETUP or the PLAY message).

The server may provide re-keying facilities by sending a new key management message in a SET_PARAMETER or ANNOUNCE messages. In the latter, the RTSP header is not used if the ANNOUNCE message includes a SDP description where the data can be provided in. The response message is then put in the new RTSP header in the response message from the client to the server. Note that the SET PARAMETER and the ANNOUNCE messages are not mandatory to support for the servers.

3.3. 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=keymgmt-prot:MIKEY
a=keymgmt-data: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=keymgmt-prot:MIKEY
a=keymgmt-data: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=keymgmt-prot:MIKEY
a=keymgmt-data:uiSDF9sdhs727ghsd/dhsoKkdOokdo7eWsnDSJD...
m=video 52230 RTP/AVP 31
a=rtpmap:31 H261/90000
```

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. As the server

```
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=keymgmt-prot:MIKEY
a=keymgmt-data: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 is then proceeded as usual (with e.g. a SETUP message for the video followed by a PLAY message).

4. 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.

The 'key-extra-auth' attribute may be (optionally) used to guarantee an authenticated binding between the session(s) and the security parameters, e.g. authenticating both the key management lines and (parts of) the surrounding SDP description. Each key management specifies the coverage of such 'key-extra-auth' attribute. A denial-of-service attack may be open if such authenticated binding is not provided. Note however, the 'key-extra-auth' cannot work when NATs are present.

Note that the purpose of the key management fields is to secure the media streams themselves. Under the assumption that the key management schemes are secure, the SDP payloads can be passed along unprotected, and the media streams will still be secure even if some attackers gained knowledge of the SDP contents. There may however, be other reasons to protect the SDP payloads such as preventing attackers from gaining any information regarding the session or the used equipment.

5. IANA Considerations

Three new attributes fields for SDP (see Section 2.1) and one new RTSP header are registered (see Section 2.2).

6. 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. A set of new attributes and headers have been defined in SDP and RTSP so that it will be possible to integrate a key management protocol (e.g. MIKEY) into SIP and RTSP.

7. Acknowledgments

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

8. 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

9. References

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

[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)", RFC 2326, April 1998.

[SDP] Handley, M., and Jacobson, V., "Session Description Protocol (SDP)", IETF, RFC2327

[SIP] Handley, M., Schulzrinne, H., Schooler, E., and Rosenberg, J., "SIP: Session Initiation Protocol", IETF, RFC2543.

[SRTP] 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 July 2002.