

Internet Engineering Task Force
MMUSIC Working Group
INTERNET-DRAFT
Expires: April 2004

J. Arkko
E. Carrara
F. Lindholm
M. Naslund
K. Norrman
Ericsson
October 2003

**Key Management Extensions for Session Description
Protocol (SDP) and Real Time Streaming Protocol (RTSP)**
<[draft-ietf-mmusic-kmgmt-ext-09.txt](#)>

Status of this memo

This document is an Internet-Draft and is in full conformance with all provisions of [Section 10 of RFC2026](#).

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or cite them other than as "work in progress".

The list of current Internet-Drafts can be accessed at <http://www.ietf.org/ietf/lid-abstracts.txt>

The list of Internet-Draft Shadow Directories can be accessed at <http://www.ietf.org/shadow.html>

Copyright Notice

Copyright (C) The Internet Society (2003). All Rights Reserved.

Abstract

This document defines general extensions for SDP and RTSP to carry messages as specified 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.....	2
1.1. Notational Conventions.....	3
2. Extensions to SDP and RTSP.....	4
2.1. SDP Extensions.....	4
2.2. RTSP Extensions.....	5
3. Usage with SIP and RTSP.....	5
3.1. General SDP processing.....	6
3.2. SIP usage.....	7
3.3. RTSP usage.....	8
3.4. Bidding-down attack prevention.....	9
3.5. Example scenarios.....	10
4. Adding further Key management protocols.....	13
5. Security Considerations.....	13
6. IANA Considerations.....	14
6.1. SDP Attribute Registration.....	14
6.2. RTSP Header Registration.....	15
6.3. Protocol Identifier Registration.....	15
7. Acknowledgments.....	16
8. Author's Addresses.....	16
9. References.....	17
9.1. Normative References.....	17
9.2. Informative References.....	17

[1. Introduction](#)

[Editor remark] All instances of RFC xxxx should be replaced with the RFC number of this document, when published. Furthermore, all instances of RFC yyyy should be replaced with the RFC number of the MIKEY (Multimedia Internet KEYing) document [[MIKEY](#)], when published.

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 solution to exchange keys and security parameters, manage and refresh 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 security association for the security

protocol. This includes one or more cryptographic keys and the 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 a new SDP attribute and RTSP header extension to support key management, and the integration within SIP and RTSP. A framework is therefore described in the following. This framework is 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.
- * The possibility to negotiate keying material end-to-end without applying end-to-end protection of the SDP (instead, hop-by-hop security mechanisms can be used which may be useful if intermediate proxies needs access to the SDP).

Currently in SDP [[SDPnew](#)], one field exists to transport keys, i.e. the "k=" field. However, this is not enough for a key management protocol as there are many more parameters that need to be transported. 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. However, RTSP [[RTSP](#)] does not use the offer/answer model with SDP, so a new header is introduced to convey key management data.

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 [[RFC2119](#)].

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 guidelines (see [[SDPnew](#)] and [[RTSP](#)]).

For both SDP and RTSP, the general method of adding the key management protocol is to introduce new attributes, one identifier to identify the specific key management protocol, and one data field where the key management protocol data is placed. 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 protocol.

The key management data SHALL be base64 encoded and comply with the base64 grammar as defined in [[SDPnew](#)]. The key management protocol identifier, KMID, is defined as below (where ALPHA and DIGIT are as defined in [[RFC2234](#)]).

KMID = 1*(ALPHA/DIGIT)

Values for the identifier, KMID, are registered and defined in accordance to [Section 6](#). Note that the KMID will be case sensitive and it is therefore RECOMMENDED that values registered are lower case letters.

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-id keymgmt-data

prtcl-id = KMID
; e.g. "mikey"

keymgmt-data = text

where KMID is as previously defined, "text" is as defined in SDP [[SDPnew](#)]. Prtcl-id refers to the set of values defined for KMID in

[Section 6](#). "text" is consistent with the requirement of base64-

encoded data, and KMID is consistent with the standard definition of non-ws-string.

The attribute may be used at session level, media level, or at both levels. An attribute defined at media level overrides an attribute defined at session level. [Section 3](#) describes in detail how the attributes are used and how the SDP is handled in different usage scenarios.

[2.2.](#) RTSP Extensions

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

```
KeyMgmt = "keymgmt" ":" 1#key-mgmt-spec
```

```
key-mgmt-spec = "prot" "=" KMID ";" "data" "=" quoted-string
```

where KMID is as previously defined and "quoted-string" as defined in the RTSP specification [[RTSP](#)]. "quoted-string" is consistent with the requirement of base64-encoded data, and KMID is consistent with the standard definition of token.

The KeyMgmt header should be possible to use in the messages described in the table below.

Method	Direction	Requirement
DESCRIBE	C->S	optional
SETUP	C->S	required
ANNOUNCE	C->S, S->C	optional (required: if re-key is supported)

Note: [Section 3](#) describes in detail how the RTSP extensions are used.

[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.

When a key management protocol is integrated with SIP/SDP and RTSP, the following requirements are placed on the key management:

- * 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 from the SIP/SDP and RTSP application, using the key management API, to receive key management data, and information of whether a message is 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](#)], however this is subject to future standardization.

3.1. General SDP processing

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

- * The identifier of the key management protocol used MUST be placed in the prtcl-id field. The protocol identifier values are specified by IANA (see [Section 6](#)).
- * The list of protocol identifiers is provided by the SDP application to (each) key management protocol, as defined in [Section 3.4](#). (to defeat bidding-down attacks).
- * The keymgmt-data field MUST be created as follows. The key management protocol MUST be used to create the key management message. This message SHALL be base64 encoded [[RFC3548](#)] by the SDP application and then encapsulated in the keymgmt-data attribute. The data may e.g. be a MIKEY message (see [[MIKEY](#)], Section 7).

A received SDP message that contains the key management attributes SHALL be processed in the following manner:

- * The key management protocol is identified according to the prtcl-id field. The protocol identifier values are specified by IANA ([Section 6](#)).
- * The key management data from the keymgmt-data field MUST be extracted, base64 decoded to reconstruct the original message, and then passed to the key management protocol.
- * The list of protocol identifiers is provided by the SDP application to the key management protocol, as defined in [Section 3.4](#). Note that depending on key management protocol, some extra parameters might also be requested by the specific API, such as the source/destination network address/port(s) for the specified media (however, this will be implementation specific depending on the actual API). The extra parameters that a key management protocol might need (other than the ones defined here) SHOULD be documented, describing their use, as well as the interaction of that key management protocol with SDP and RTSP.
- * If the key management processing is successful, then the answerer sends back the answer. Otherwise, if the key management rejects the offer, an error is sent back ("606 Not Acceptable") and the

session is aborted. See [Section 3.2](#) for further details.

Note that the key management attribute MAY be repeated more than once (e.g., one at session level and one at media level). Consequently, the process is repeated for each key management attribute detected. However, in case of failure of the key management (on either session or media level), the session setup SHALL be aborted (see also [Section 3.2](#) and [Section 3.3](#) for more details).

In the Offer/Answer case, and in general when there is an answer, if more than one key management protocol is supported, multiple instances of the key management attribute MAY be included in the initial offer, each transporting a different key management data, thus indicating supported alternatives.

If the sender includes more than one key management protocol attribute at session level (analogous for the media level), these SHOULD be listed in order of preference (the first being the preferred). The receiver selects the key management protocol it wishes to use and includes only that attribute in the answer. If the receiver does not support any of the sender's suggested key management protocols, the receiver returns an error message (see [section 3.2](#) and [section 3.3](#)), whereby the sender MUST abort the current setup procedure.

Note that the placement of multiple key management offers in a single message has the disadvantage that the message expands and the computational workload for the offerer will increase drastically. Unless the guidelines of [Section 3.4](#) are followed, multiple lines may open up for bidding-down attacks.

The following Sections describe the specific use with SIP and RTSP respectively. There are of course other cases where SDP is used, such as SAP and HTTP. If SDP is transported in an Offer-Answer model fashion, then the guidelines in [Section 3.2](#) can be used. However, for one-way SDP distribution (i.e. without back channel), the above guidelines can be used though with certain restrictions. First, the key management protocol MUST support one-way messages, and secondly, only one key management protocol SHALL be offered (i.e. no negotiation will be possible).

This document does NOT address one-to-many distribution scenarios, as this would require different types of key management protocols. The support for such scenarios is for future standardization.

[3.2. SIP usage](#)

When used with SIP and the offer/answer model, the offerer SHOULD include the key management data within an offer that contains the media description it should apply to. At the answerer's side, the key

management protocol checks the validity of the key management message, together with the availability of the offered attribute

values, and then provides the key management data to be included in the answer.

If the offer is not accepted, the answerer SHOULD return a "606 Not Acceptable" message, including one or more Warning headers (e.g. a 306 "Attribute not understood" when one of the parameters is not supported). The session is then aborted (and it is up to local policy or end user to decide how to continue).

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. In general, the re-INVITE (and the key exchange) must be finalized before the security protocol can change the keys. The same key management protocol used in the original INVITE SHALL also be used in the re-INVITE carrying re-keying. If the re-INVITE carrying re-keying fails (e.g., the authentication verification fails), the answerer SHOULD send a "606 Not Acceptable" message, including one or more Warning headers (at least a 306). The offerer MUST then abort the security setup.

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 has some kind of binding to the media, so that the response to the server will be processed as required.

To obtain a session description, the client initially contacts the server via a DESCRIBE message (according to RTSP, a media description could also be obtained by other means e.g. using http). The initial key management message from the RTSP server is sent to the client in the SDP of the 200 OK in response to the DESCRIBE. When responding to this, the client uses the new RTSP header to send back an answer (included in the SETUP message). If a server receives a SETUP message in which it expects a key management message, but none is included, a 403 Forbidden SHOULD be returned to the client, whereby the current setup MUST be aborted.

The processing of creating a key management header in RTSP SHALL be as follow:

- * The identifier of the key management protocol used (e.g. MIKEY) MUST be placed in the "prot" field of the header. The prot values are specified by IANA ([Section 6](#)).

- * The list of protocol identifiers is sent by the RTSP application to (each) key management protocol as described in [Section 3.4](#). (to defeat bidding-down attacks).
- * The keymgmt-data field MUST be created as follows. The key management protocol MUST be used to create the key management message. This message SHALL be base64 encoded by the SDP application and then encapsulated in the "data" field of the header. The data may e.g. be a MIKEY message (see [[MIKEY](#)], [Section 7](#)).

A received key management header SHALL be processed in the following manner:

- * The key management protocol is identified according to the "prot" field.
- * The key management data from the "data" field MUST be extracted, base64 decoded to reconstruct the original message, and then passed to the key management protocol. Note that depending on the key management protocol, some extra parameters might of course be requested by the specific API, such as the source/destination network address/port(s) for the specified media (however, this will be implementation specific depending on the actual API).
- * Depending on the outcome of the key management processing (i.e. whether it was successful or not), the processing can proceed according to normal rules (see also below).

The server MAY provide re-keying/updating facilities by sending a new key management message in an ANNOUNCE message. 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 is to be used.

[3.4](#). Bidding-down attack prevention

The possibility to support multiple key management protocols may, unless properly handled, introduce so-called bidding-down attacks. Specifically, a man-in-the-middle could "peel off" cryptographically strong offers (deleting key-mgmt lines from the message), leaving only weaker ones as the responder's choice. To avoid this, the list of identifiers of the proposed key management protocols MUST be authenticated. The authentication MUST be done separately by each key management protocol (see e.g. Section 7.1 in [[MIKEY](#)]).

Accordingly, it MUST be specified (in the key management protocol specification itself or in a companion document) how the list of key

management protocol identifiers can be processed to be authenticated
from the offerer to the answerer by the specific key management

protocol. Note that even if only one key management protocol is used, that still MUST authenticate its own protocol identifier.

The list of protocol identifiers MUST then be given to each of the selected (offered) key management protocols by the application with ";" separated identifiers. All the offered protocol identifiers MUST be included, in the same order as they appear in the corresponding SDP description.

The protocol list can formally be described as

```
prtcl-list = KMID *(";" KMID)
```

where KMID is as defined in [Section 2](#).

For example, if the offered protocols are MIKEY and two yet-to-be-invented protocols KEYP1, KEYP2, the SDP is:

```
v=0
o=alice 2891092738 2891092738 IN IP4 lost.example.com
s=Secret discussion
t=0 0
c=IN IP4 lost.example.com
a=key-mgmt:mikey <data1>
a=key-mgmt:keyp1 <data2>
a=key-mgmt:keyp2 <data3>
m=audio 39000 RTP/SAVP 98
a=rtpmap:98 AMR/8000
m=video 42000 RTP/SAVP 31
a=rtpmap:31 H261/90000
```

The protocol list, "mikey;keyp1;keyp2", would be generated from the SDP description and used as input to each specified key management protocol (together with the data for that protocol). Each of the three protocols includes this protocol identifier list in its authentication coverage (according to its protocol specification).

If more than one protocol are supported by the offerer, it is RECOMMENDED that all acceptable protocols are included in the first offer, rather than making single, subsequent alternative offers in response to error messages, see "Security Considerations".

[3.5](#). 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:

$v=0$

Arkko, et al.

[Page 10]

```
o=alice 2891092738 2891092738 IN IP4 w-land.example.com
s=Cool stuff
e=alice@w-land.example.com
t=0 0
c=IN IP4 w-land.example.com
a=key-mgmt:mikey uiSDF9sdhs727ghsd/dhsoKkd0okdo7eWsnDSJD...
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 foo.example.com
s=Cool stuff
e=bob@foo.example.com
t=0 0
c=IN IP4 foo.example.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 if she wished to protect only the audio stream.

```
v=0
o=alice 2891092738 2891092738 IN IP4 w-land.example.com
s=Cool stuff
e=alice@w-land.example.com
t=0 0
c=IN IP4 w-land.example.com
m=audio 49000 RTP/SAVP 98
a=rtpmap:98 AMR/8000
a=key-mgmt:mikey uiSDF9sdhs727ghsd/dhsoKkd0okdo7eWsnDSJD...
m=video 52230 RTP/AVP 31
a=rtpmap:31 H261/90000
```

Note that even if the key management attribute were specified at

session level, the video part would 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@example.com
```

The server sends back an OK message including an 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.example.com
s=Action Movie
e=action@movie.example.com
t=0 0
c=IN IP4 movie.example.com
a=key-mgmt:mikey uiSDF9sdhs727ghsd/dhsoKkd0okdo7ewsnDSJD...
m=audio 0 RTP/SAVP 98
a=rtpmap:98 AMR/8000
control:rtsp://movie.example.com/action/audio
m=video 0 RTP/SAVP 31
a=rtpmap:31 H261/90000
control:rtsp://movie.example.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.example.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 further Key management protocols

This framework cannot be used with all key management protocols. The key management protocol needs to comply with the requirements described in [Section 3](#). In addition to this, the following needs to be defined:

- * The key management protocol identifier to be used as the protocol identifier should be registered at IANA according to [Section 6](#) (e.g. "mikey" for MIKEY).
- * The information that the key management needs from SDP and RTSP, and vice versa, as described in [Section 3](#). The exact API is implementation specific, but it SHOULD at least support to exchange the specified information.
- * The key management protocol to be added MUST be such, that the processing in [Section 3](#) (describing its interactions with SDP and RTSP) can be applied. Note in particular, [Section 3.4](#) requires each key management protocol to specify how the list of protocol identifiers is authenticated inside that key management protocol. The key management MUST always be given the protocol identifier(s) of the key management protocol(s) included in the offer in the correct order as they appear.

Finally, it is obviously crucial to analyze possible security implications induced by the introduction of a new key management protocol in the described framework.

5. Security Considerations

As a general practice, it is a good thing, not only to try to secure the session, but also to secure the session setup. However, the security of the session setup might not be possible on an end-to-end basis, but the setup may require to be protected on a hop-by-hop basis (this is generally the case for SIP/SDP when intermediate proxies need to obtain information about the sessions etc, c.f. [\[E2M\]](#)). In fact, the focus of this framework is mainly when end-to-end protection of the session setup is not used, but where the media streams need to be end-to-end protected. General security considerations for the session setup can be found in SDP [\[SDPnew\]](#), SIP [\[SIP\]](#), and RTSP [\[RTSP\]](#).

The security will also depend on the encapsulated level of security

the key management protocol offers. It follows that, under the assumption that the key management schemes are secure, the SDP can be

passed along unprotected without affecting the key management as such, and the media streams will still be secure even if some attackers gained knowledge of the SDP contents. Further security considerations can be found for each key management protocol (for MIKEY these can be found in [[MIKEY](#)]).

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 (indirectly) rely on some of the session information from SDP (e.g., address information), an attack on SDP may have indirect consequences on the key management. Even if the key management protocol does not rely on parameters of SDP and will not be affected by manipulation of these, different DoS attacks aimed at SDP (e.g. the SIMCAP extensions) may lead to undesired interruption in the setup.

The use of multiple key management protocols in the same offer may open up the possibility of a bidding-down attack, as specified in [Section 3.4](#). To exclude such possibility, the authentication of the protocol identifier list is used. Note though, that the security level of the authenticated protocol identifier will be as high (or low), as the "weakest" protocol. Therefore, it is discouraged to offer protocols with too different security levels.

Note that it is impossible to assure the authenticity of a declined offer, since even if it comes from the true respondent, the fact that the answerer declines the offer usually means that he does not support the protocol(s) offered, and consequently cannot be expected to authenticate the response either. This means that if the initiator is unsure of which protocol(s) the responder supports, we RECOMMEND that the initiator offers all acceptable protocols in a single offer. If not, this opens up the possibility for a "man-in-the-middle" (MITM) to affect the outcome of the eventually agreed upon protocol, by faking unauthenticated error messages until the initiator eventually offers a protocol "to the liking" of the MITM. This is not really a security problem, but rather a mild form of denial of service that can be avoided by following the above recommendation. In the case that the response declines any security (therefore there is impossibility of authenticating it), the session setup SHALL be aborted.

6. IANA Considerations

[6.1.](#) SDP Attribute Registration

A new SDP attribute needs to be registered for the purpose of key management protocol integration with SDP.

Contact: Fredrik Lindholm
mailto: fredrik.lindholm@ericsson.com

Arkko, et al.

[Page 14]

tel: +46 8 58531705

SDP Attribute ("att-field"):

Name: key-mgmt
Long form: key management protocol
Type of name: att-field
Type of attribute: Media and session level
Purpose: See RFC xxxx, [Section 2](#).
Reference: RFC xxxx, [Section 2.1](#)
Values: See [Section 6.3](#)

6.2. RTSP Header Registration

A new RTSP Header needs to be registered for the purpose of key management protocol integration with RTSP.

Following the guidelines of [[RTSP](#)], the registration is defined as follows:

Header name: keymgmt
Header syntax: see RFC xxxx, [Section 2.2](#)
Intended usage: see RFC xxxx, [Section 2.2](#)
Proxy treatment: Proxies SHOULD forward the header
Purpose: see RFC xxxx, [Section 2](#)

6.3. Protocol Identifier Registration

This document defines one new name space associated with the protocol identifier, KMID, defined in [Section 2](#) to be used with the above registered key-mgmt attributes in SDP and RTSP.

A new registry needs to be set up for the KMID parameter, with the following registration created initially: "mikey".

Contact: Fredrik Lindholm
mailto: fredrik.lindholm@ericsson.com
tel: +46 8 58531705

Value name: mikey
Long name: Multimedia Internet KEYing
Purpose: Usage of MIKEY with the key-mgmt attribute
Reference: [Section 7](#) in RFC yyyy

Note that this registration will imply that the protocol identifier, KMID, name space will be shared between SDP and RTSP.

Further values may be registered according to the "Specification Required" policy as defined in [[RFC2434](#)]. Each new registration needs

to indicate the parameter name, and register it within IANA. Note that the parameter name is case sensitive and it is recommended that

the name should be in lower case letters. For each new registration, it is mandatory that a permanent, stable, and publicly accessible document exists that specifies the semantics of the registered parameter and the requested details of interaction between the key management protocol and SDP, as specified in RFC xxxx.

The registration itself of new values should be sent to IANA.

Registrations should include the following information:

- * Contact: the contact name and email address
- * Value name: the name of the value being registered (which MUST comply with the KMID as defined in [Section 2](#))
- * Long Name: long-form name in English (optional)
- * Purpose: short explanation of the purpose of the registered name.
- * Reference: a reference to the specification (e.g. RFC number) of the draft providing the usage guidelines in accordance to [Section 4](#) (and also complying to the specified requirements).

[7. Acknowledgments](#)

Thanks to: Rolf Blom and Magnus Westerlund. A special thanks to Joerg Ott and Colin Perkins.

[8. Author's Addresses](#)

Jari Arkko
Ericsson
02420 Jorvas
Finland

Phone: +358 40 5079256
Email: jari.arkko@ericsson.com

Elisabetta Carrara
Ericsson Research
SE-16480 Stockholm
Sweden

Phone: +46 8 50877040
Email: elisabetta.carrara@ericsson.com

Fredrik Lindholm
Ericsson Research
SE-16480 Stockholm
Sweden

Phone: +46 8 58531705
Email: fredrik.lindholm@ericsson.com

Mats Naslund
Ericsson Research
SE-16480 Stockholm
Sweden

Phone: +46 8 58533739
Email: mats.naslund@ericsson.com

Karl Norrman
Ericsson Research
SE-16480 Stockholm

Phone: +46 8 4044502

Sweden

E-Mail: karl.norrman@ericsson.com

Arkko, et al.

[Page 16]

9. References

9.1. Normative References

[OAM] Rosenberg, J. and Schulzrinne, H., "An Offer/Answer Model with the Session Description Protocol (SDP)", IETF, [RFC 3264](#).

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

[RFC2119] Bradner, S. "Key words for use in RFCs to Indicate Requirement Levels", IETF, [RFC 2119](#).

[SDPnew] Handley, M., Jacobson, V., and Perkins, C., "SDP: Session Description Protocol", Internet Draft, IETF, Work in progress (MMUSIC), [draft-ietf-mmusic-sdp-new-13.txt](#).

[SIP] Handley, M., Schulzrinne, H., Schooler, E., and Rosenberg, J., "SIP: Session Initiation Protocol", IETF, [RFC 3261](#).

[RFC2234] Crocker, D. and Overell, P., "Augmented BNF for Syntax Specifications: ABNF", [RFC 2234](#), November 1997.

[RFC2434] Narten, T. and Alvestrand, H., "Guidelines for Writing an IANA Considerations Section in RFCs", IETF, [RFC 2434](#).

[RFC3548] Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", IETF, [RFC 3548](#).

9.2. Informative References

[E2M] Ono, K. and Tachimoto, S., "End-to-middle security in the Session Initiation Protocol (SIP)", Internet Draft, IETF, Work in Progress.

[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", IETF, RFC yyyy, [Internet Draft, 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 document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the Internet Society or other Internet organizations, except as needed for the purpose of developing Internet standards in which case the procedures for copyrights defined in the Internet Standards process must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by the Internet Society or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

This Internet-Draft expires in April 2004.

