Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: July 7, 2017

P. Kyzivat
L. Xiao
C. Groves
Huawei
R. Hansen
Cisco Systems
January 3, 2017

CLUE Signaling
draft-ietf-clue-signaling-10

Abstract

This document specifies how CLUE-specific signaling such as the CLUE protocol [I-D.ietf-clue-protocol] and the CLUE data channel [I-D.ietf-clue-datachannel] are used with each other and with existing signaling mechanisms such as SIP and SDP to produce a telepresence call.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

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 to cite them other than as "work in progress."

This Internet-Draft will expire on July 7, 2017.

Table of Contents

1. Introduction

To enable devices to participate in a telepresence call, selecting the sources they wish to view, receiving those media sources and displaying them in an optimal fashion, CLUE employs two principal and inter-related protocol negotiations. SDP, conveyed via SIP, is used to negotiate the specific media capabilities that can be delivered to specific addresses on a device. Meanwhile, CLUE protocol [I-D.ietf-clue-protocol] messages, transported via a CLUE data channel [I-D.ietf-clue-datachannel], are used to negotiate the Capture Sources available, their attributes and any constraints in their use. They also allow the far end device to specify which Captures they wish to receive.

Beyond negotiating the CLUE channel, SDP is also used to negotiate the details of supported media streams and the maximum capability of each of those streams. As the CLUE Framework [I-D.ietf-clue-framework] defines a manner in which the Media Provider expresses their maximum encoding group capabilities, SDP is also used to express the encoding limits for each potential Encoding.

Backwards-compatibility is an important consideration of the protocol: it is vital that a CLUE-capable device contacting a device that does not support CLUE is able to fall back to a fully functional non-CLUE call. The document also defines how a non-CLUE call may be upgraded to CLUE in mid-call, and similarly how CLUE functionality can be removed mid-call to return to a standard non-CLUE call.

2. 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].

This document uses terminology defined in the CLUE Framework [I-D.ietf-clue-framework].

A few additional terms specific to this document are defined as follows:

non-CLUE device: A device that supports standard SIP and SDP, but either does not support CLUE, or that does but does not currently wish to invoke CLUE capabilities.

CLUE-controlled media: A media "m=" line that is under CLUE control; the Capture Source that provides the media on this "m=" line is negotiated in CLUE. See Section 4 for details of how this control is signaled in SDP. There is a corresponding "non-CLUE-controlled" media term.

3. Media Feature Tag Definition

The "sip.clue" media feature tag indicates support for CLUE. A CLUE-capable device SHOULD include this media feature tag in its REGISTER requests and OPTION responses. It SHOULD also include the media feature tag in INVITE and UPDATE [RFC3311] requests and responses.

Presence of the media feature tag in the contact field of a request or response can be used to determine that the far end supports CLUE.

4. SDP Grouping Framework CLUE Extension Semantics

4.1. General

This section defines a new SDP Grouping Framework [RFC5888] extension called 'CLUE'.

The CLUE extension can be indicated using an SDP session-level 'group' attribute. Each SDP media "m=" line that is included in this group, using SDP media-level mid attributes, is CLUE-controlled, by a CLUE data channel also included in this CLUE group.

Currently only support for a single CLUE group is specified; support for multiple CLUE groups in a single session is beyond the scope of this document. A device MUST NOT include more than one CLUE group in its SDP unless it is following a specification that defines how multiple CLUE channels are signaled, and is either able to determine that the other side of the SDP exchange supports multiple CLUE channels, or is able to fail gracefully in the event it does not.

4.2. The CLUE data channel and the CLUE grouping semantic

The CLUE data channel [I-D.ietf-clue-datachannel] is a bidirectional data channel [I-D.ietf-rtcweb-data-channel] used for the transport of CLUE messages, conveyed within an SCTP over DTLS connection. This channel must be established before CLUE protocol messages can be exchanged and CLUE-controlled media can be sent.

The data channel is negotiated over SDP as described in [I-D.ietf-mmusic-data-channel-sdpneg]. A CLUE-capable device wishing to negotiate CLUE MUST also include a CLUE group in the SDP and include the "mid" of the "m=" line for the data channel in that group. A CLUE group MUST include the "mid" of the "m=" line for one (and only one) data channel.

Presence of the data channel in a CLUE group in an SDP offer or answer also serves, along with the "sip.clue" media feature tag, as an indication that the device supports CLUE and wishes to upgrade the call to include CLUE-controlled media. A CLUE-capable device SHOULD include a data channel "m=" line in offers and, when allowed by [RFC3264], answers.

4.3. CLUE-controlled media and the CLUE grouping semantic

CLUE-controlled media lines in an SDP are "m=" lines in which the content of the media streams to be sent is negotiated via the CLUE protocol [I-D.ietf-clue-protocol]. For an "m=" line to be CLUE-controlled, its "mid" value MUST be included in a CLUE group. CLUE-controlled media is controlled by the CLUE protocol as negotiated on the CLUE data channel with an "mid" included in the CLUE group.

"m=" lines not specified as under CLUE control follow normal rules for media streams negotiated in SDP as defined in documents such as [RFC3264].

The restrictions on CLUE-controlled media always apply to "m=" lines in an SDP offer or answer, even if negotiation of the data channel in SDP failed due to lack of CLUE support by the remote device or for any other reason, or in an offer if the recipient does not include the "mid" of the corresponding "m=" line in their CLUE group.

4.4. SDP semantics for CLUE-controlled media

4.4.1. Signaling CLUE Encodings

The CLUE Framework [I-D.ietf-clue-framework] defines the concept of "Encodings", which represent the sender's encode ability. Each Encoding the Media Provider wishes to signal is signaled via an "m=" line of the appropriate media type, which MUST be marked as sendonly with the "a=sendonly" attribute or as inactive with the "a=inactive" attribute.

The encoder limits of active (eg, "a=sendonly") Encodings can then be expressed using existing SDP syntax. For instance, for H.264 see Table 6 in [RFC6184] for a list of valid parameters for representing encoder sender stream limits.

These Encodings are CLUE-controlled and hence MUST include an "mid" in a CLUE group as defined above.

As well as the normal restrictions defined in [RFC3264] the stream MUST be treated as if the "m=" line direction attribute had been set to "a=inactive" until the Media Provider has received a valid CLUE Configure message specifying the Capture to be used for this stream. This means that RTP packets MUST NOT be sent until configuration is complete, while non-media packets such as STUN, RTCP and DTLS MUST be sent as per their relevant specifications if negotiated.

Every "m=" line representing a CLUE Encoding MUST contain a "label" attribute as defined in [RFC4574]. This label is used to identify the Encoding by the sender in CLUE Advertisement messages and by the receiver in CLUE Configure messages. Each label used for a CLUE-controlled "m=" line MUST be different from the label on all other "m=" lines in the same CLUE group in the SDP message, unless an "m=" line represents a dependent stream related to another "m=" line (such as a FEC stream), in which case it MUST have the same label value as the "m=" line on which it depends.

4.4.1.1. Referencing Encodings in the CLUE protocol

CLUE Encodings are defined in SDP, but can be referenced from CLUE protocol messages - this is how the protocol defines which Encodings are part of an Encoding group (in Advertisement messages) and which Encoding with which to encode a specific Capture (in Configure messages). The labels on the CLUE-controlled "m=" lines are the references that are used in the CLUE protocol.

Each <encID> (in encodingIDList) in a CLUE Advertisement message SHOULD represent an Encoding defined in SDP; the specific Encoding referenced is a CLUE-controlled "m=" line in the most recent SDP sent by the sender of the Advertisement message with a label value corresponding to the text content of the <encID>.

Similarly, each <encodingID> (in captureEncodingType) in a CLUE Configure message SHOULD represent an Encoding defined in SDP; the specific Encoding referenced is a CLUE-controlled "m=" line in the most recent SDP received by the sender of the Configure message with a label value corresponding to the text content of the <encodingID>.

Note that the non-atomic nature of SDP/CLUE protocol interaction may mean that there are temporary periods where an <encID>/<encodingID> in a CLUE message does not reference an SDP "m=" line, or where an Encoding represented in SDP is not referenced in a CLUE protocol message. See Section 5 for specifics.

4.4.2. Negotiating receipt of CLUE Capture Encodings in SDP

A receiver who wishes to receive a CLUE stream via a specific Encoding requires an "a=recvonly" "m=" line that matches the "a=sendonly" Encoding.

These "m=" lines are CLUE-controlled and hence MUST include their "mid" in the CLUE group corresponding to the CLUE group of the Encoding they wish to receive. They MAY include a "label" attribute, but this is not required by CLUE, as only label values associated with "a=sendonly" Encodings are referenced by CLUE protocol messages.

4.5. SDP Offer/Answer Procedures

4.5.1. Generating the Initial Offer

A CLUE-capable device sending an initial SDP offer of a SIP session SHOULD include an "m=" line for the data channel to convey the CLUE protocol, along with a CLUE group containing the "mid" of the data channel "m=" line.

For interoperability with non-CLUE devices a CLUE-capable device sending an initial SDP offer SHOULD NOT include any "m=" line for CLUE-controlled media beyond the "m=" line for the CLUE data channel, and SHOULD include at least one non-CLUE-controlled media "m=" line.

If the device has evidence that the receiver is also CLUE-capable, for instance due to receiving an initial INVITE with no SDP but including a "sip.clue" media feature tag, the above recommendation is waived, and the initial offer MAY contain "m=" lines for CLUE-controlled media.

With the same interoperability recommendations as for Encodings, the sender of the initial SDP offer MAY also include "a=recvonly" media lines to preallocate "m=" lines to receive media. Alternatively, it MAY wait until CLUE protocol negotiation has completed before including these lines in a new offer/answer exchange - see Section 5 for recommendations.

4.5.2. Generating the Answer

4.5.2.1. Negotiating use of CLUE and the CLUE data channel

If the recipient is CLUE-capable and the initial offer contains both an "m=" line for a data channel and a CLUE group containing the "mid" for that "m=" line, they SHOULD negotiate data channel support for an "m=" line, and include the "mid" of that "m=" line in a corresponding CLUE group.

A CLUE-capable recipient that receives an "m=" line for a data channel but no corresponding CLUE group containing the "mid" of that "m=" line MAY still include a corresponding data channel "m=" line if there are any other non-CLUE protocols it can convey over that channel, but MUST NOT negotiate use of the CLUE protocol on this channel.

4.5.2.2. Negotiating CLUE-controlled media

If the initial offer contained "a=recvonly" CLUE-controlled media lines the recipient SHOULD include corresponding "a=sendonly" CLUE-controlled media lines for accepted Encodings, up to the maximum number of Encodings it wishes to advertise. As CLUE-controlled media, the "mid" of these "m=" lines must be included in the corresponding CLUE group.

If the initial offer contained "a=sendonly" CLUE-controlled media lines the recipient MAY include corresponding "a=recvonly" CLUE-controlled media lines, up to the maximum number of Capture Encodings it wishes to receive. Alternatively, it MAY wait until CLUE protocol negotiation has completed before including these lines in a new offer/answer exchange - see Section 5 for recommendations.

4.5.2.3. Negotiating non-CLUE controlled media

A CLUE-controlled device implementation may prefer to render initial, single-stream audio and/or video for the user as rapidly as possible, transitioning to CLUE-controlled media once that has been negotiated. Alternatively, an implementation may wish to suppress initial media, only providing media once the final, CLUE-controlled streams have been negotiated.

The receiver of the initial offer, if making the call CLUE-enabled with their SDP answer, can make their preference clear by their action in accepting or rejecting non-CLUE-controlled media lines. Rejecting these "m=" lines will ensure that no non-CLUE-controlled media flows before the CLUE-controlled media is negotiated. In contrast, accepting one or more non-CLUE-controlled "m=" lines in this initial answer will enable initial media to flow.

If the answerer chooses to send initial non-CLUE-controlled media in a CLUE-enabled call, Section 4.5.4.1 addresses the need to disable it once CLUE-controlled media is fully negotiated.

4.5.3. Processing the initial Offer/Answer negotiation

In the event that both offer and answer include a data channel "m=" line with a mid value included in corresponding CLUE groups, CLUE has been successfully negotiated and the call is now CLUE-enabled. If not then the call is not CLUE-enabled.

4.5.3.1. Successful CLUE negotiation

In the event of successful CLUE-enablement of the call, devices MUST now begin negotiation of the CLUE channel, see [I-D.ietf-clue-datachannel] for negotiation details. If negotiation is successful, sending of CLUE protocol [I-D.ietf-clue-protocol] messages can begin.

A CLUE-capable device MAY choose not to send media on the non-CLUE-controlled channels during the period in which control of the CLUE-controlled media lines is being negotiated. However, a CLUE-capable device MUST still be prepared to receive media on non-CLUE-controlled media lines that have been successfully negotiated as defined in [RFC3264].

If either side of the call wishes to add additional CLUE-controlled "m=" lines to send or receive CLUE-controlled media they MAY now send a SIP request with a new SDP offer following the normal rules of SDP offer/answer and any negotiated extensions.

4.5.3.2. CLUE negotiation failure

In the event that the negotiation of CLUE fails and the call is not CLUE-enabled once the initial offer/answer negotiation completes then CLUE is not in use in the call. The CLUE-capable devices MUST either revert to non-CLUE behaviour or terminate the call.

4.5.4. Modifying the session

4.5.4.1. Adding and removing CLUE-controlled media

Subsequent offer/answer exchanges MAY add additional "m=" lines for CLUE-controlled media; in most cases at least one additional exchange will be required before both sides have added all the Encodings and ability to receive Encodings that they desire. Devices MAY delay adding "a=recvonly" CLUE-controlled "m=" lines until after CLUE protocol negotiation completes - see Section 5 for recommendations.

Subsequent offer/answer exchanges MAY also deactivate "m=" lines for CLUE-controlled media.

Once CLUE media has been successfully negotiated devices SHOULD ensure that non-CLUE-controlled media is deactivated in cases where it corresponds to the media type of CLUE-controlled media that has been successfully negotiated. This deactivation may require an additional SDP exchange, or may be incorporated into one that is part of the CLUE negotiation.

4.5.4.2. Enabling CLUE mid-call

A CLUE-capable device that receives an initial SDP offer from a non-CLUE device SHOULD include a new data channel "m=" line and corresponding CLUE group in any subsequent offers it sends, to indicate that it is CLUE-capable.

If, in an ongoing non-CLUE call, an SDP offer/answer exchange completes with both sides having included a data channel "m=" line in their SDP and with the "mid" for that channel in corresponding CLUE groups then the call is now CLUE-enabled; negotiation of the data channel and subsequently the CLUE protocol begin.

4.5.4.3. Disabling CLUE mid-call

If, during an ongoing CLUE-enabled call a device wishes to disable CLUE, it can do so by following the procedures for closing a data channel defined in Section 5.2.4 of [I-D.ietf-mmusic-data-channel-sdpneg]: sending a new SDP offer/answer exchange and subsequent SCTP SSN reset for the CLUE channel. It MUST also remove the CLUE group. Without the CLUE group any "m=" lines that were previously CLUE-controlled no longer are; implementations MAY disable them by setting their ports to 0 or may continue to use them - in the latter case how they are used is outside the scope of this document.

If a device follows the procedure above, or an SDP offer-answer negotiation completes in a fashion in which either the "m=" CLUE data channel line was not successfully negotiated, and/or one side did not include the data channel in a matching CLUE group then CLUE for this call is disabled. In the event that this occurs, CLUE is no longer enabled. Any active "m=" lines still included in a CLUE group are no longer CLUE-controlled and the implementation MAY either disable them in a subsequent negotiation or continue to use them in some other fashion. If the data channel is still present but not included in the CLUE group semantic CLUE protocol messages MUST no longer be sent.

Note that this is distinct from cases where the CLUE protocol negotiation fails, or an error occurs in the CLUE protocol; see [I-D.ietf-clue-protocol] for details of media and state preservation

in this circumstance. These measures also apply if the CLUE data channel fails, or is closed/reset without a corresponding SDP exchange to disable the "m=" line.

5. Interaction of CLUE protocol and SDP negotiations

Information about media streams in CLUE is split between two message types: SDP, which defines media addresses and limits, and the CLUE channel, which defines properties of Capture Devices available, scene information and additional constraints. As a result certain operations, such as advertising support for a new transmissible Capture with associated stream, cannot be performed atomically, as they require changes to both SDP and CLUE messaging.

This section defines how the negotiation of the two protocols interact, provides some recommendations on dealing with intermediate stages in non-atomic operations, and mandates additional constraints on when CLUE-configured media can be sent.

5.1. Independence of SDP and CLUE negotiation

To avoid the need to implement interlocking state machines with the potential to reach invalid states if messages were to be lost, or be rewritten en-route by middle boxes, the state machines in SDP and CLUE operate independently. The state of the CLUE channel does not restrict when an implementation may send a new SDP offer or answer, and likewise the implementation's ability to send a new CLUE Advertisement or Configure message is not restricted by the results of or the state of the most recent SDP negotiation (unless the SDP negotiation has removed the CLUE channel).

The primary implication of this is that a device may receive an SDP with a CLUE Encoding for which it does not yet have Capture information, or receive a CLUE Configure message specifying a Capture Encoding for which the far end has not negotiated a media stream in SDP.

CLUE messages contain an <encID> (in encodingIDList) or <encodingID> (in captureEncodingType), which is used to identify a specific encoding or captureEncoding in SDP; see [I-D.ietf-clue-data-model-schema] for specifics. The non-atomic nature of CLUE negotiation means that a sender may wish to send a new Advertisement before the corresponding SDP message. As such the sender of the CLUE message MAY include an <encID> which does not currently match a CLUE-controlled "m=" line label in SDP; A CLUE-capable implementation MUST NOT reject a CLUE protocol message solely because it contains <encID> elements that do not match a label in SDP.

The current state of the CLUE participant or Media Provider/Consumer state machines do not affect compliance with any of the normative language of [RFC3264]. That is, they MUST NOT delay an ongoing SDP exchange as part of a SIP server or client transaction; an implementation MUST NOT delay an SDP exchange while waiting for CLUE negotiation to complete or for a Configure message to arrive.

Similarly, a device in a CLUE-enabled call MUST NOT delay any mandatory state transitions in the CLUE Participant or Media Provider/Consumer state machines due to the presence or absence of an ongoing SDP exchange.

A device with the CLUE Participant state machine in the ACTIVE state MAY choose not to move from ESTABLISHED to ADV (Media Provider state machine) or from ESTABLISHED to WAIT FOR CONF RESPONSE (Media Consumer state machine) based on the SDP state. See [I-D.ietf-clue-protocol] for CLUE state machine specifics. Similarly, a device MAY choose to delay initiating a new SDP exchange based on the state of their CLUE state machines.

5.2. Constraints on sending media

While SDP and CLUE message states do not impose constraints on each other, both impose constraints on the sending of media - CLUE-controlled media MUST NOT be sent unless it has been negotiated in both CLUE and SDP: an implementation MUST NOT send a specific CLUE Capture Encoding unless its most recent SDP exchange contains an active media channel for that Encoding AND the far end has sent a CLUE Configure message specifying a valid Capture for that Encoding.

5.3. Recommendations for operating with non-atomic operations

CLUE-capable devices MUST be able to handle states in which CLUE messages make reference to EncodingIDs that do not match the most recently received SDP, irrespective of the order in which SDP and CLUE messages are received. While these mismatches will usually be transitory a device MUST be able to cope with such mismatches remaining indefinitely. However, this document makes some recommendations on message ordering for these non-atomic transitions.

CLUE-capable devices SHOULD ensure that any inconsistencies between SDP and CLUE signaling are temporary by sending updated SDP or CLUE messages as soon as the relevant state machines and other constraints permit.

Generally, implementations that receive messages for which they have incomplete information SHOULD wait until they have the corresponding information they lack before sending messages to make changes related

to that information. For example, an answerer that receives a new SDP offer with three new "a=sendonly" CLUE "m=" lines for which it has received no CLUE Advertisement providing the corresponding capture information SHOULD include corresponding "a=inactive" lines in its answer, and SHOULD make a new SDP offer with "a=recvonly" when and if a new Advertisement arrives with Captures relevant to those Encodings.

Because of the constraints of SDP offer/answer and because new SDP negotiations are generally more 'costly' than sending a new CLUE message, implementations needing to make changes to both channels SHOULD prioritize sending the updated CLUE message over sending the new SDP message. The aim is for the recipient to receive the CLUE changes before the SDP changes, allowing the recipient to send their SDP answers without incomplete information, reducing the number of new SDP offers required.

6. Interaction of CLUE protocol and RTP/RTCP CaptureID

[I-D.ietf-clue-framework] allows for Multiple Content Captures MCCs): Captures which contain multiple source Captures, whether composited into a single stream or switched based on some metric.

The Captures that contribute to these MCCs may or may not be defined in the Advertisement message. If they are defined and the MCC is providing them in a switched format the recipient may wish to determine which originating source Capture is currently being provided, so that they can apply geometric corrections based on that Capture's geometry, or take some other action based on the original Capture information.

To do this, [I-D.ietf-clue-rtp-mapping] allows for the CaptureID of the originating Capture to be conveyed via RTP or RTCP. A Media Provider sending switched media for an MCC with defined originating sources MUST send the CaptureID in both RTP and RTCP, as described in the mapping document.

6.1. CaptureID reception during MCC redefinition

Because the RTP/RTCP CaptureID is delivered via a different channel to the Advertisement in which in the contents of the MCC are defined there is an intrinsic race condition in cases in which the contents of an MCC are redefined.

When a Media Provider redefines an MCC which involves CaptureIDs, the reception of the relevant CaptureIDs by the recipient will either lead or lag reception and processing of the new Advertisement by the recipient. As such, a Media Consumer MUST not be disrupted by any of

the following in any CLUE- controlled media stream it is receiving, whether that stream is for a static Capture or for an MCC (as any static Capture may be redefined to an MCC in a later Advertisement):

- Receiving RTP or RTCP containing a CaptureID when the most recently processed Advertisement means that none are expected.
- Receiving RTP or RTCP without CaptureIDs when the most recently processed Advertisement means that media CaptureIDs are expected.
- Receiving a CaptureID in RTP or RTCP for a Capture defined in the most recently processed Advertisement, but which the same Advertisement does not include in the MCC.
- Receiving a CaptureID in RTP or RTCP for a Capture not defined in the most recently processed Advertisement.

7. Multiplexing of CLUE-controlled media using BUNDLE

7.1. Overview

A CLUE call may involve sending and/or receiving significant numbers of media streams. Conventionally, media streams are sent and received on unique ports. However, each separate port used for this purpose may impose costs that a device wishes to avoid, such as the need to open that port on firewalls and NATs, the need to collect ICE candidates [RFC5245], etc.

The BUNDLE [I-D.ietf-mmusic-sdp-bundle-negotiation] extension can be used to negotiate the multiplexing of multiple media lines onto a single 5-tuple for sending and receiving media, allowing devices in calls to another BUNDLE-supporting device to potentially avoid some of the above costs.

While CLUE-capable devices MAY support the BUNDLE extension for this purpose supporting the extension is not mandatory for a device to be CLUE-compliant.

7.2. Usage of BUNDLE with CLUE

This specification imposes no additional requirements or restrictions on the usage of BUNDLE when used with CLUE. There is no restriction on combining CLUE-controlled media lines and non-CLUE-controlled media lines in the same BUNDLE group or in multiple such groups. However, there are several steps an implementation may wish to take to ameliorate the cost and time requirements of extra SDP offer/answer exchanges between CLUE and BUNDLE.

7.2.1. Generating the Initial Offer

BUNDLE mandates that the initial SDP offer MUST use a unique address for each "m=" line with a non-zero port. Because CLUE implementations generally will not include CLUE-controlled media lines with the exception of the data channel in the initial SDP offer, CLUE devices that support large numbers of streams can avoid ever having to open large numbers of ports if they successfully negotiate BUNDLE.

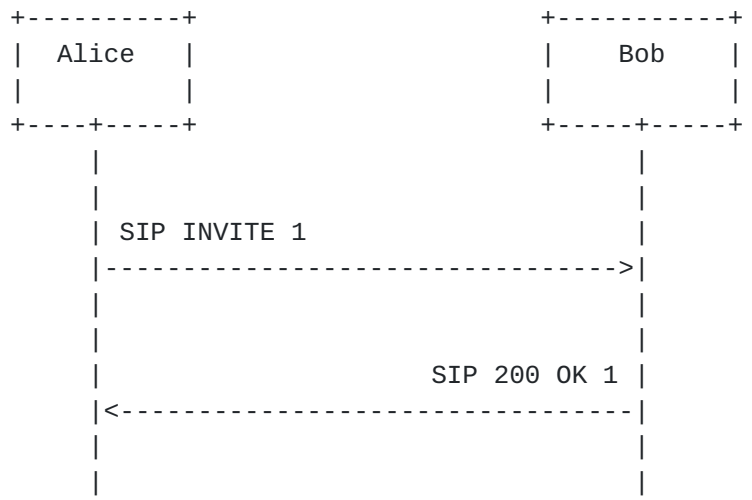
7.2.2. Multiplexing of the data channel and RTP media

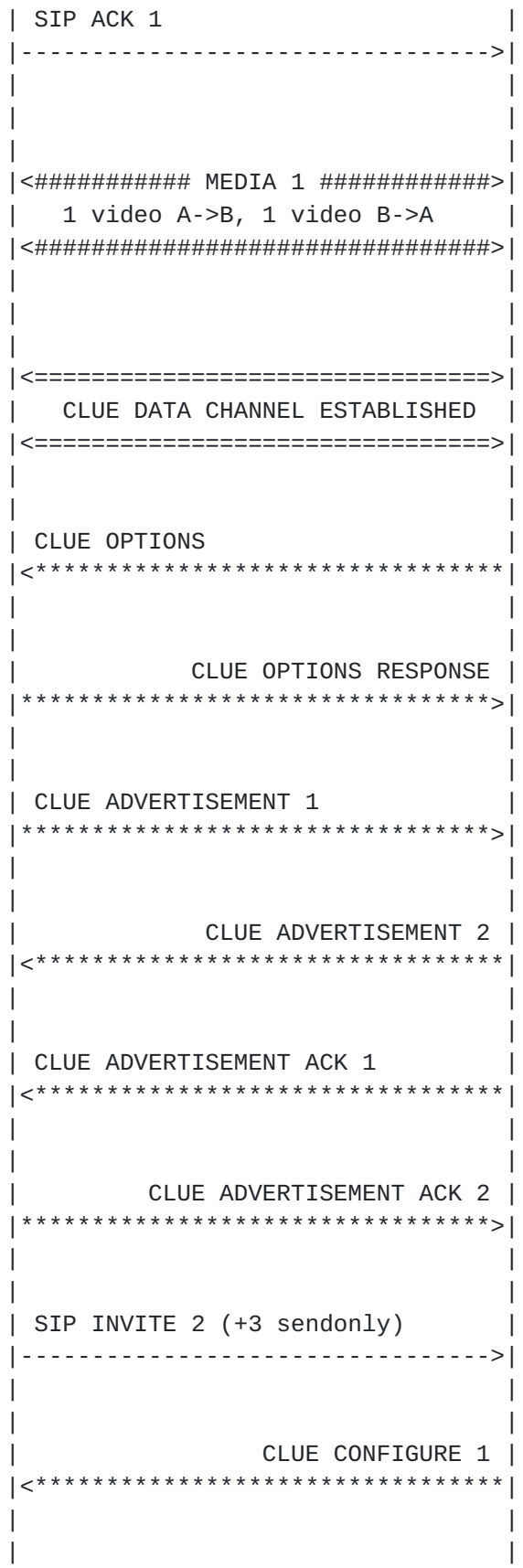
BUNDLE-supporting CLUE-capable devices MAY include the data channel in the same BUNDLE group as RTP media. In this case the device MUST be able to demultiplex the various transports - see section 9.2 of the BUNDLE draft [I-D.ietf-mmusic-sdp-bundle-negotiation]. If the BUNDLE group includes other protocols than the data channel transported via DTLS the device MUST also be able to differentiate the various protocols.

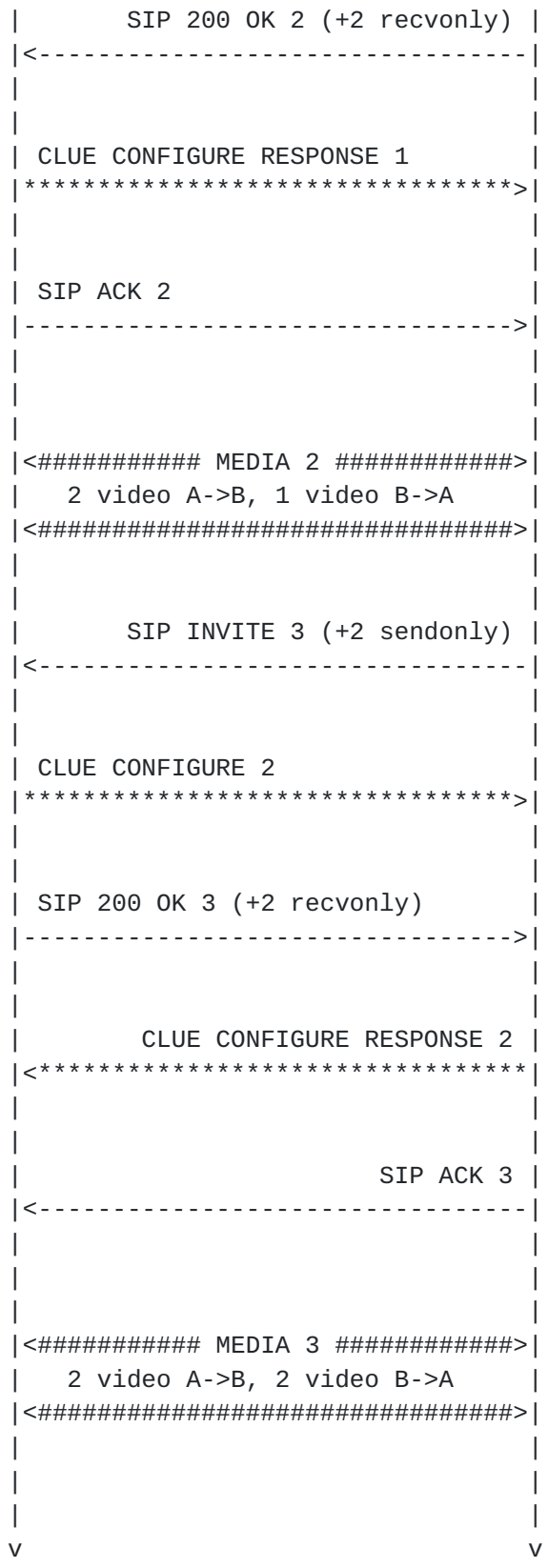
8. Example: A call between two CLUE-capable Endpoints

This example illustrates a call between two CLUE-capable Endpoints. Alice, initiating the call, is a system with three cameras and three screens. Bob, receiving the call, is a system with two cameras and two screens. A call-flow diagram is presented, followed by a summary of each message.

To manage the size of this section the SDP snippets only illustrate video "m=" lines. SIP ACKs are not always discussed. Note that BUNDLE is not in use.




SIP ACK 1
MEDIA 1
1 video A->B, 1 video B->A
CLUE DATA CHANNEL ESTABLISHED
CLUE OPTIONS
CLUE OPTIONS RESPONSE
CLUE ADVERTISEMENT 1
CLUE ADVERTISEMENT 2
CLUE ADVERTISEMENT ACK 1
CLUE ADVERTISEMENT ACK 2
SIP INVITE 2 (+3 sendonly)
CLUE CONFIGURE 1


SIP 200 OK 2 (+2 recvonly)
CLUE CONFIGURE RESPONSE 1
SIP ACK 2
MEDIA 2
2 video A->B, 1 video B->A
SIP INVITE 3 (+2 sendonly)
CLUE CONFIGURE 2
SIP 200 OK 3 (+2 recvonly)
CLUE CONFIGURE RESPONSE 2
SIP ACK 3
MEDIA 3
2 video A->B, 2 video B->A

In SIP INVITE 1, Alice sends Bob a SIP INVITE including in the SDP body the basic audio and video capabilities and the data channel as per [I-D.ietf-mmusic-sctp-sdp]. Alice also includes the "sip.clue" media feature tag in the INVITE. A snippet of the SDP showing the grouping attribute and the video "m=" line are shown below. Alice has included a "CLUE" group, and included the mid corresponding to a data channel in the group (3). Note that Alice has chosen not to include any CLUE-controlled media in the initial offer - the mid value of the video line is not included in the "CLUE" group.

```
...
a=group:CLUE 3
...
m=video 6002 RTP/AVP 96
a=rtpmap:96 H264/90000
a=fmtp:96 profile-level-id=42e016;max-mbps=108000;max-fs=3600
a=sendrecv
a=mid:2
...
m=application 6100 UDP/DTLS/SCTP webrtc-datachannel
a=sctp-port: 5000
a=dcmap:2 subprotocol="CLUE";ordered=true
a=mid:3
```

Bob responds with a similar SDP in SIP 200 OK 1, which also has a "CLUE" group including the mid value of a data channel; due to their similarity no SDP snippet is shown here. Bob wishes to receive initial media, and so includes corresponding non-CLUE-controlled audio and video lines. Bob also includes the "sip.clue" media feature tag in the 200 OK. Alice and Bob are each now able to send a single audio and video stream. This is illustrated as MEDIA 1.

With the successful initial SDP O/A Alice and Bob are also free to negotiate the CLUE data channel. This is illustrated as CLUE DATA CHANNEL ESTABLISHED.

Once the data channel is established CLUE protocol negotiation begins. In this case Bob is the Channel Initiator and sends a CLUE OPTIONS message describing his version support. On receiving that message Alice sends her corresponding CLUE OPTIONS RESPONSE.

With the OPTIONS phase complete Alice now sends her CLUE Advertisement (CLUE ADVERTISEMENT 1). She advertises three static Captures representing her three cameras. She also includes switched Captures suitable for two- and one-screen systems. All of these Captures are in a single Capture Scene, with suitable Capture Scene

Views to tell Bob that he should either subscribe to the three static Captures, the two switched Captures or the one switched Capture. Alice has no simultaneity constraints, so includes all six Captures in one simultaneous set. Finally, Alice includes an Encoding Group with three Encoding IDs: "enc1", "enc2" and "enc3". These Encoding IDs aren't currently valid, but will match the next SDP offer she sends.

Bob received CLUE ADVERTISEMENT 1 but does not yet send a Configure message, because he has not yet received Alice's Encoding information, so as yet he does not know if she will have sufficient resources to send him the two streams he ideally wants at a quality he is happy with. Because Bob is not sending an immediate Configure with the "ack" element set he must send an explicit Advertisement Acknowledgement message (CLUE ADVERTISEMENT ACK 1) to signal receipt of CLUE ADVERTISEMENT 1.

Bob also sends his CLUE Advertisement (CLUE ADVERTISEMENT 2) - though the diagram shows that this occurs after Alice sends CLUE ADVERTISEMENT 1 Bob sends his Advertisement independently and does not wait for CLUE ADVERTISEMENT 1 to arrive. He advertises two static Captures representing his cameras. He also includes a single composed Capture for single-screen systems, in which he will composite the two camera views into a single video stream. All three Captures are in a single Capture Scene, with suitable Capture Scene Views to tell Alice that she should either subscribe to the two static Captures, or the single composed Capture. Bob also has no simultaneity constraints, so includes all three Captures in one simultaneous set. Bob also includes a single Encoding Group with two Encoding IDs: "foo" and "bar".

Similarly, Alice receives CLUE ADVERTISEMENT 2 but does not yet send a Configure message, because she has not yet received Bob's Encoding information, sending instead an Advertisement Acknowledgement(CLUE ADVERTISEMENT ACK 2).

Both sides have now sent their CLUE Advertisement messages and an SDP exchange is required to negotiate Encodings. For simplicity, in this case Alice is shown sending an INVITE with a new offer; in many implementations both sides might send an INVITE, which would be resolved by use of the 491 Request Pending resolution mechanism from [RFC3261].

Alice now sends SIP INVITE 2. She maintains the sendrecv audio, video and CLUE "m=" lines, and she adds three new sendonly "m=" lines to represent the three CLUE-controlled Encodings she can send. Each of these "m=" lines has a label corresponding to one of the Encoding IDs from CLUE ADVERTISEMENT 1. Each also has its mid added to the

grouping attribute to show they are controlled by the CLUE channel. A snippet of the SDP showing the grouping attribute, data channel and the video "m=" lines are shown below:

```
...
a=group:CLUE 3 4 5 6
...
m=video 6002 RTP/AVP 96
a=rtpmap:96 H264/90000
a=fmtp:96 profile-level-id=42e016;max-mbps=108000;max-fs=3600
a=sendrecv
a=mid:2
...
m=application 6100 UDP/DTLS/SCTP webrtc-datachannel
a=sctp-port: 5000
a=dcmap:2 subprotocol="CLUE";ordered=true
a=mid:3
...
m=video 6004 RTP/AVP 96
a=rtpmap:96 H264/90000
a=fmtp:96 profile-level-id=42e016
a=sendonly
a=mid:4
a=label:enc1
m=video 6006 RTP/AVP 96
a=rtpmap:96 H264/90000
a=fmtp:96 profile-level-id=42e016
a=sendonly
a=mid:5
a=label:enc2
m=video 6008 RTP/AVP 96
a=rtpmap:96 H264/90000
a=fmtp:96 profile-level-id=42e016
a=sendonly
a=mid:6
a=label:enc3
```

Bob now has all the information he needs to decide which streams to configure, allowing him to send both a CLUE Configure message and his SDP answer. As such he now sends CLUE CONFIGURE 1. This requests the pair of switched Captures that represent Alice's scene, and he configures them with encoder ids "enc1" and "enc2".

Bob also sends his SDP answer as part of SIP 200 OK 2. Alongside his original audio, video and CLUE "m=" lines he includes two active recvonly "m= "lines and a zeroed "m=" line for the third. He adds

their mid values to the grouping attribute to show they are controlled by the CLUE channel. A snippet of the SDP showing the grouping attribute and the video "m=" lines are shown below (mid 100 represents the CLUE channel, not shown):

```
...
a=group:CLUE 11 12 100
...
m=video 58722 RTP/AVP 96
a=rtpmap:96 H264/90000
a=fmtp:96 profile-level-id=42e016;max-mbps=108000;max-fs=3600
a=sendrecv
a=mid:10
...
m=video 58724 RTP/AVP 96
a=rtpmap:96 H264/90000
a=fmtp:96 profile-level-id=42e016;max-mbps=108000;max-fs=3600
a=recvonly
a=mid:11
m=video 58726 RTP/AVP 96
a=rtpmap:96 H264/90000
a=fmtp:96 profile-level-id=42e016;max-mbps=108000;max-fs=3600
a=recvonly
a=mid:12
m=video 0 RTP/AVP 96
```

Alice receives Bob's message CLUE CONFIGURE 1 and sends CLUE RESPONSE 1 to ack its reception. She does not yet send the Capture Encodings specified, because at this stage she hasn't processed Bob's answer SDP and so hasn't negotiated the ability for Bob to receive these streams.

On receiving SIP 200 OK 2 from Bob Alice sends her SIP ACK (SIP ACK 2). She is now able to send the two streams of video Bob requested - this is illustrated as MEDIA 2.

The constraints of offer/answer meant that Bob could not include his encoding information as new "m=" lines in SIP 200 OK 2. As such Bob now sends SIP INVITE 3 to generate a new offer. Along with all the streams from SIP 200 OK 2 Bob also includes two new sendonly streams. Each stream has a label corresponding to the Encoding IDs in his CLUE ADVERTISEMENT 2 message. He also adds their mid values to the grouping attribute to show they are controlled by the CLUE channel. A snippet of the SDP showing the grouping attribute and the video "m=" lines are shown below (mid 100 represents the CLUE channel, not shown):

```
...
a=group:CLUE 11 12 13 14 100
...
m=video 58722 RTP/AVP 96
a=rtpmap:96 H264/90000
a=fmtp:96 profile-level-id=42e016;max-mbps=108000;max-fs=3600
a=sendrecv
a=mid:10
...
m=video 58724 RTP/AVP 96
a=rtpmap:96 H264/90000
a=fmtp:96 profile-level-id=42e016;max-mbps=108000;max-fs=3600
a=recvonly
a=mid:11
m=video 58726 RTP/AVP 96
a=rtpmap:96 H264/90000
a=fmtp:96 profile-level-id=42e016;max-mbps=108000;max-fs=3600
a=recvonly
a=mid:12
m=video 0 RTP/AVP 96
m=video 58728 RTP/AVP 96
a=rtpmap:96 H264/90000
a=fmtp:96 profile-level-id=42e016
a=sendonly
a=label:foo
a=mid:13
m=video 58730 RTP/AVP 96
a=rtpmap:96 H264/90000
a=fmtp:96 profile-level-id=42e016
a=sendonly
a=label:bar
a=mid:14
```

Having received this Alice now has all the information she needs to send her CLUE Configure message and her SDP answer. In CLUE CONFIGURE 2 she requests the two static Captures from Bob, to be sent on Encodings "foo" and "bar".

Alice also sends SIP 200 OK 3, matching two recvonly "m=" lines to Bob's new sendonly lines. She includes their mid values in the grouping attribute to show they are controlled by the CLUE channel. Alice also now deactivates the initial non-CLUE-controlled media, as bidirectional CLUE-controlled media is now available. A snippet of the SDP showing the grouping attribute and the video "m=" lines are shown below (mid 3 represents the data channel, not shown):

```
...
a=group:CLUE 3 4 5 7 8
...
m=video 0 RTP/AVP 96
a=mid:2
...
m=video 6004 RTP/AVP 96
a=rtpmap:96 H264/90000
a=fmtp:96 profile-level-id=42e016
a=sendonly
a=mid:4
a=label:enc1
m=video 6006 RTP/AVP 96
a=rtpmap:96 H264/90000
a=fmtp:96 profile-level-id=42e016
a=sendonly
a=mid:5
a=label:enc2
m=video 0 RTP/AVP 96
m=video 6010 RTP/AVP 96
a=rtpmap:96 H264/90000
a=fmtp:96 profile-level-id=42e016;max-mbps=108000;max-fs=3600
a=recvonly
a=mid:7
m=video 6012 RTP/AVP 96
a=rtpmap:96 H264/90000
a=fmtp:96 profile-level-id=42e016;max-mbps=108000;max-fs=3600
a=recvonly
a=mid:8
```

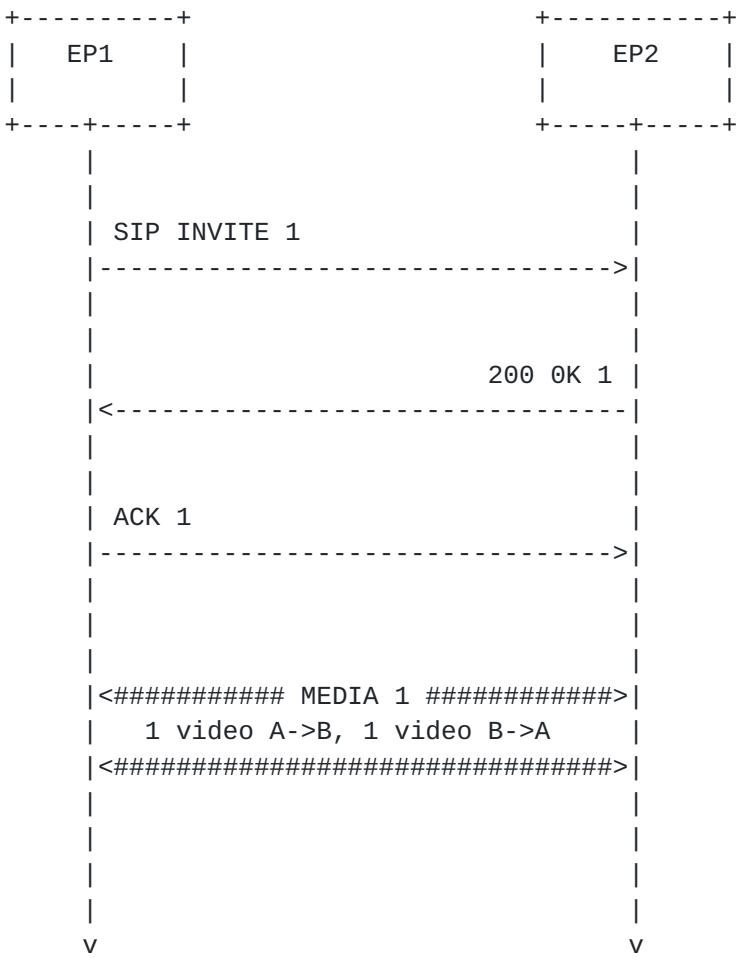
Bob receives Alice's message CLUE CONFIGURE 2 and sends CLUE RESPONSE 2 to ack its reception. Bob does not yet send the Capture Encodings specified, because he hasn't yet received and processed Alice's SDP answer and negotiated the ability to send these streams.

Finally, on receiving SIP 200 OK 3 Bob is now able to send the two streams of video Alice requested - this is illustrated as MEDIA 3.

Both sides of the call are now sending multiple video streams with their sources defined via CLUE negotiation. As the call progresses either side can send new Advertisement or Configure message or new SDP offer/answers to add, remove or change what they have available or want to receive.

9. Example: A call between a CLUE-capable and non-CLUE Endpoint

In this brief example Alice is a CLUE-capable Endpoint making a call to Bob, who is not CLUE-capable (i.e. is not able to use the CLUE protocol).



In SIP INVITE 1, Alice sends Bob a SIP INVITE including in the SDP body the basic audio and video capabilities and the data channel as per [I-D.ietf-mmusic-sctp-sdp]. Alice also includes the "sip.clue" media feature tag in the INVITE. A snippet of the SDP showing the grouping attribute and the video "m=" line are shown below. Alice has included a "CLUE" group, and included the mid corresponding to a data channel in the group (3). Note that Alice has chosen not to include any CLUE-controlled media in the initial offer - the mid value of the video line is not included in the "CLUE" group.

```
...
a=group:CLUE 3
...
m=video 6002 RTP/AVP 96
a=rtpmap:96 H264/90000
a=fmtp:96 profile-level-id=42e016;max-mbps=108000;max-fs=3600
a=sendrecv
a=mid:2
...
m=application 6100 UDP/DTLS/SCTP webrtc-datachannel
a=sctp-port: 5000
a=dcmap:2 subprotocol="CLUE";ordered=true
a=mid:3
```

Bob is not CLUE-capable, and hence does not recognize the "CLUE" semantic for grouping attribute, nor does he support the data channel. IN SIP 200 OK 1 he responds with an answer with audio and video, but with the data channel zeroed.

From the lack of the data channel and grouping framework Alice understands that Bob does not support CLUE, or does not wish to use it. Both sides are now able to send a single audio and video stream to each other. Alice at this point begins to send her fallback video: in this case likely a switched view from whichever camera shows the current loudest participant on her side.

10. Acknowledgements

The team focusing on this draft consists of: Roni Even, Rob Hansen, Christer Holmberg, Paul Kyzivat, Simon Pietro-Romano, Roberta Presta.

Christian Groves and Jonathan Lennox have contributed detailed comments and suggestions.

11. IANA Considerations

11.1. New SDP Grouping Framework Attribute

This document registers the following semantics with IANA in the "Semantics for the "group" SDP Attribute" subregistry (under the "Session Description Protocol (SDP) Parameters" registry per [RFC5888]:

Semantics	Token	Reference
CLUE-controlled m-line	CLUE	[this draft]

11.2. New SIP Media Feature Tag

This specification registers a new media feature tag in the SIP [RFC3264] tree per the procedures defined in [RFC2506] and [RFC3840].

Media feature tag name: sip.clue

ASN.1 Identifier: 1.3.6.1.8.4.29

Summary of the media feature indicated by this tag: This feature tag indicates that the device supports CLUE-controlled media.

Values appropriate for use with this feature tag: Boolean.

The feature tag is intended primarily for use in the following applications, protocols, services, or negotiation mechanisms:

This feature tag is most useful in a communications application for describing the capabilities of a device to use the CLUE control protocol to negotiate the use of multiple media streams.

Related standards or documents: [this draft]

Security Considerations: Security considerations for this media feature tag are discussed in Section 12 of [this draft].

Name(s) & email address(es) of person(s) to contact for further information:

- CLUE workgroup: clue@ietf.org

- CLUE chairs: clue-chairs@ietf.org

Intended usage: COMMON

12. Security Considerations

CLUE makes use of a number of protocols and mechanisms, either defined by CLUE or long-standing. The security considerations section of the CLUE Framework [I-D.ietf-clue-framework] addresses the need to secure these mechanisms by following the recommendations of the individual protocols.

Beyond the need to secure the constituent protocols, the use of CLUE does impose additional security concerns. One area of increased risk involves the potential for a malicious party to subvert a CLUE-capable device to attack a third party by driving large volumes of media (particularly video) traffic at them by establishing a connection to the CLUE-capable device and directing the media to the victim. While this is a risk for all media devices, a CLUE-capable device may allow the attacker to configure multiple media streams to be sent, significantly increasing the volume of traffic directed at the victim.

This attack can be prevented by ensuring that the media recipient intends to receive the media packets. As such all CLUE-capable devices MUST support key negotiation and receiver intent assurance via DTLS [RFC5763] on CLUE-controlled RTP "m=" lines. All CLUE-controlled RTP "m" lines must be secured and implemented using mechanisms such as SRTP [RFC3711]; no specific security mechanisms are made mandatory to use due to the issues addressed in [RFC7202]. Due to the requirements of backwards compatibility, this is not a mandatory requirement for non-CLUE-controlled "m=" lines.

CLUE also defines a new media feature tag that indicates CLUE support. This tag may be present even in non-CLUE calls, which increases the metadata available about the sending device, which can help an attacker differentiate between multiple devices and help them identify otherwise anonymised users via the fingerprint of features their device supports. To prevent this, SIP signaling SHOULD always be encrypted using TLS [RFC5630].

The CLUE protocol also carries additional information that could be used to help fingerprint a particular user or to identify the specific version of software being used. CLUE Framework [I-D.ietf-clue-protocol] provides details of these issues and how to mitigate them.

13. Change History

-10: Revision by Rob Hansen

* Changes to draft-ietf-clue-protocol between 07 and 11 reviewed to ensure compatibility between documents has been maintained.

* Expanded the portion of the document related to fingerprinting with info on the CLUE channel as well as SIP.

-09: Revision by Rob Hansen

* A few minor spelling tweaks

* Made removing the CLUE group mandatory when disabling CLUE mid-call. Made clear that any CLUE-controlled m-lines should be disabled or else how they're used is up to the implementation.

-08: Revision by Rob Hansen

* Spelling and grammar fixes from Paul and Christian gratefully adopted

* Expanded the section on disabling CLUE mid-call to make explicit the actions required to disable the CLUE channel gracefully, or to handle someone else doing the same.

* Made a number of fixes to the example call flow to better reflect the recommendations in the document.

-07: Revision by Rob Hansen

* Removed the entire 'Media line directionality' section as a discussion of the pros/cons of using bidirectional vs unidirectional schemes wasn't suitable for a finalised version. The unidirectionality requirement is covered normatively in an earlier section.

* BUNDLE no longer includes an address synchronisation step so the suggestion to wait until that done has been replaced with some general language about following any negotiated extensions.

* Added OPTIONS negotiation to the example flow, and revised the flow to ensure it matched protocol document.

* Section on not sending CLUE control media until CLUE negotiation completes narrowed to notify that only RTP should not be sent until negotiation completes and add RTCP to the list of things that should be sent as normal, in line with a=inactive.

* Make explicit that m=recvonly lines don't need to have a label, as only m=sendonly lines are referenced by CLUE protocol messages.

* Fix formatting of IANA sections. Improve syntax of feature tag section in line with Paul's suggestions. Definition of feature tag narrowed to be multiple media lines *negotiated via CLUE protocol* rather than more generic 'multiple media lines'.

* General corrections to grammar, spelling and readability based on Christian, Paul and Mark; in many cases suggested text was gratefully accepted.

-06: Revision by Rob Hansen

* State machine interactions updated to match versions in -04 of protocol doc.

* Section on encoding updated to specify both encID and encodingID from data model doc.

* Removed the limitations on describing H264 encoding limits using SDP syntax as an open issue.

* Previous draft had SRTP and DTLS mandatory to implement and to use on CLUE- controlled m lines. Current version has DTLS mandatory to implement, and 'security' mandatory to use but does not define what that security is.

* Terminology reference to framework doc reinforced. All terminology that duplicates framework removed. All text updated with capitalisation that matches framework document's terminology.

* SDP example syntax updated to match that of ietf-clue-datachannel and hence ietf-mmusic-data-channel-sdpneg.

-05: Revision by Rob Hansen

* SRTP/DTLS made mandatory for CLUE-controlled media lines.

* IANA consideration section added (text as proposed by Christian Groves).

* Includes provision for dependent streams on seperate "m" lines having the same encID as their parent "m" line.

* References to putting CLUE-controlled media and data channels in more than one CLUE group removed, since the document no longer supports using more than one CLUE group.

* Section on CLUE controlled media restrictions still applying even if the call does not end up being CLUE enabled being rewritten to hopefully be clearer.

* Other minor syntax improvements.

-04: Revision by Rob Hansen

* Updated DTLS/SCTP channel syntax in examples to fix errors and match latest format defined in draft-ietf-mmusic-sctp-sdp-07.

* Clarified the behaviour if an SDP offer includes a CLUE-controlled "m" line and the answer accepts that "m" line but without CLUE control of that line.

* Added a new section on the sending and receiving of CaptureIDs in RTP and RTCP. Includes a section on the necessity of the receiver coping with unexpected CaptureIDs (or the lack thereof) due to MCCs being redefined in new Advertisement messages.

* Added reminder on IANA section on registering grouping semantic and media feature tag, removed the less formal sections that did the same job.

* Fixed and clarified issues raised by Christian's document review.

* Added a number of security considerations.

-03: Revision by Rob Hansen

* Clarified text on not rejecting messages because they contain unknown encIDs.

* Removed normative language in section on accepting/rejecting non-CLUE-controlled media in the initial answer.

* Example SDP updated to include the data channel "m" lines.

* Example call flow updated to show disablement of non-CLUE-controlled media once CLUE-controlled media is flowing.

-02: Revision by Rob Hansen

* Added section on not accepting non-CLUE-controlled "m" lines in the initial answer when CLUE is to be negotiated.

* Removed previous language attempting to describe media restrictions for CLUE-controlled "m" lines that had not been configured, and replaced it with much more accurate 'treat as "a=inactive" was set'.

* Made label element mandatory for CLUE-controlled media (was previously "SHOULD include", but there didn't seem a good reason for this - anyone wishing to include the "m" line but not immediately use it in CLUE can simply leave it out of the <encodingIDList>.)

* Added a section on the specifics of relating encodings in SDP to <encID> elements in the CLUE protocol, including the fact that both Advertisement and Configure messages reference the *encoding* (eg, in the Configure case the sender of the Configure message includes the labels of the recipient's "m" lines as their <encID> contents).

* Minor revisions to the section on complying with normative SDP/CLUEstate machine language to clarify that these were not new normative language, merely that existing normative language still applies.

* Removed appendices which previously contained information to be transferred to the protocol and data channel drafts. Removed other text that discussed alternatives to the current approach.

* Cleaned up some 'todo' text.

-01: Revision by Rob Hansen

* Revised terminology - removed the term 'CLUE-enabled' device as insufficiently distinct from 'CLUE-capable' and instead added a term for 'CLUE-enabled' calls.

* Removed text forbidding RTCP and instead added text that ICE/DTLS negotiation for CLUE controlled media must be done as normal irrespective of CLUE negotiation.

* Changed 'sip.telepresence' to 'sip.clue' and 'TELEPRESENCE' grouping semantic back to CLUE.

* Made it mandatory to have exactly one mid corresponding to a data channel in a CLUE group

* Forbade having multiple CLUE groups unless a specification for doing so is published.

* Refactored SDP-related text; previously the encoding information had been in the "initial offer" section despite the fact that we recommend that the initial offer doesn't actually include any encodings. I moved the specifications of encodings and how they're received to an earlier, seperate section.

* Added text on how the state machines in CLUE and SDP are allowed to affect one another, and further recommendations on how a device should handle the sending of CLUE and SDP changes.

-00: Revision by Rob Hansen

* Submitted as -00 working group document

draft-kyzivat-08: Revisions by Rob Hansen

* Added media feature tag for CLUE support ('sip.telepresence')

* Changed grouping semantic from 'CLUE' to 'TELEPRESENCE'

* Restructured document to be more centred on the grouping semantic and its use with O/A

* Lots of additional text on usage of the grouping semantic

* Stricter definition of CLUE-controlled m lines and how they work

* Some additional text on defining what happens when CLUE supports is added or removed

* Added details on when to not send RTCP for CLUE-controlled "m" lines.

* Added a section on using BUNDLE with CLUE

* Updated data channel references to point at new WG document rather than indivual draft

draft-kyzivat-07: Revisions by Rob Hansen

* Removed the text providing arguments for encoding limits being in SDP and encoding groups in the CLUE protocol in favor of the specifics of how to negotiate encodings in SDP

* Added normative language on the setting up of a CLUE call, and added sections on mid-call changes to the CLUE status.

* Added references to [I-D.ietf-clue-datachannel] where appropriate.

* Added some terminology for various types of CLUE and non-CLUE states of operation.

* Moved language related to topics that should be in [I-D.ietf-clue-datachannel] and [I-D.ietf-clue-protocol], but that has not yet been resolved in those documents, into an appendix.

draft-kyzivat-06: Revisions by Rob Hansen

* Removed CLUE message XML schema and details that are now in draft-presta-clue-protocol

* Encoding limits in SDP section updated to note that this has been investigated and discussed and is the current working assumption of the WG, though consensus has not been fully achieved.

* A section has also been added on the current mandation of unidirectional "m" lines.

* Updated CLUE messaging in example call flow to match draft-presta-clue-protocol-03

draft-kyzivat-05: Revisions by pkyzivat:

* Specified versioning model and mechanism.

* Added explicit response to all messages.

* Rearranged text to work with the above changes. (Which rendered diff almost useless.)

draft-kyzivat-04: Revisions by Rob Hansen: ???

draft-kyzivat-03: Revisions by pkyzivat:

* Added a syntax section with an XML schema for CLUE messages. This is a strawhorse, and is very incomplete, but it establishes a template for doing this based on elements defined in the data model. (Thanks to Roberta for help with this!)

* Did some rewording to fit the syntax section in and reference it.

* Did some relatively minor restructuring of the document to make it flow better in a logical way.

draft-kyzivat-02: A bunch of revisions by pkyzivat:

* Moved roberta's call flows to a more appropriate place in the document.

* New section on versioning.

* New section on NAK.

* A couple of possible alternatives for message acknowledgment.

* Some discussion of when/how to signal changes in provider state.

* Some discussion about the handling of transport errors.

* Added a change history section.

These were developed by Lennard Xiao, Christian Groves and Paul, so added Lennard and Christian as authors.

draft-kyzivat-01: Updated by roberta to include some sample call flows.

draft-kyzivat-00: Initial version by pkyzivat. Established general outline for the document, and specified a few things thought to represent wg consensus.

14. References

14.1. Normative References

[I-D.ietf-clue-framework]
Duckworth, M., Pepperell, A., and S. Wenger, "Framework for Telepresence Multi-Streams", draft-ietf-clue-framework-25 (work in progress), January 2016.

[I-D.ietf-clue-data-model-schema]
Presta, R. and S. Romano, "An XML Schema for the CLUE data model", draft-ietf-clue-data-model-schema-17 (work in progress), August 2016.

[I-D.ietf-clue-protocol]
Presta, R. and S. Romano, "CLUE protocol", draft-ietf-clue-protocol-11 (work in progress), January 2017.

[I-D.ietf-clue-datachannel]
Holmberg, C., "CLUE Protocol data channel", draft-ietf-clue-datachannel-14 (work in progress), August 2016.

[I-D.ietf-clue-rtp-mapping]
Even, R. and J. Lennox, "Mapping RTP streams to CLUE Media Captures", draft-ietf-clue-rtp-mapping-10 (work in progress), November 2016.

[I-D.ietf-mmusic-sctp-sdp]
Holmberg, C., Shpount, R., Loreto, S., and G. Camarillo, "Session Description Protocol (SDP) Offer/Answer Procedures For Stream Control Transmission Protocol (SCTP) over Datagram Transport Layer Security (DTLS) Transport.", draft-ietf-mmusic-sctp-sdp-20 (work in progress), December 2016.

[I-D.ietf-mmusic-data-channel-sdpneg]
Drage, K., Makaraju, M., Stoetzer-Bradler, J., Ejzak, R., and (. (Unknown), "SDP-based Data Channel Negotiation", draft-ietf-mmusic-data-channel-sdpneg-10 (work in progress), September 2016.

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

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

[RFC4574] Levin, O. and G. Camarillo, "The Session Description Protocol (SDP) Label Attribute", RFC 4574, DOI 10.17487/RFC4574, August 2006, <http://www.rfc-editor.org/info/rfc4574>.

[RFC5763] Fischl, J., Tschofenig, H., and E. Rescorla, "Framework for Establishing a Secure Real-time Transport Protocol (SRTP) Security Context Using Datagram Transport Layer Security (DTLS)", RFC 5763, DOI 10.17487/RFC5763, May 2010, <http://www.rfc-editor.org/info/rfc5763>.

14.2. Informative References

[RFC2506] Holtman, K., Mutz, A., and T. Hardie, "Media Feature Tag Registration Procedure", BCP 31, RFC 2506, DOI 10.17487/RFC2506, March 1999, <http://www.rfc-editor.org/info/rfc2506>.

[RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, DOI 10.17487/RFC3261, June 2002, <http://www.rfc-editor.org/info/rfc3261>.

[RFC3264] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model with Session Description Protocol (SDP)", RFC 3264, DOI 10.17487/RFC3264, June 2002, <http://www.rfc-editor.org/info/rfc3264>.

[RFC3311] Rosenberg, J., "The Session Initiation Protocol (SIP) UPDATE Method", RFC 3311, DOI 10.17487/RFC3311, October 2002, <http://www.rfc-editor.org/info/rfc3311>.

[RFC3840] Rosenberg, J., Schulzrinne, H., and P. Kyzivat, "Indicating User Agent Capabilities in the Session Initiation Protocol (SIP)", RFC 3840, DOI 10.17487/RFC3840, August 2004, <http://www.rfc-editor.org/info/rfc3840>.

[RFC5245] Rosenberg, J., "Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal for Offer/Answer Protocols", RFC 5245, DOI 10.17487/RFC5245, April 2010, <http://www.rfc-editor.org/info/rfc5245>.

[RFC5630] Audet, F., "The Use of the SIPS URI Scheme in the Session Initiation Protocol (SIP)", RFC 5630, DOI 10.17487/RFC5630, October 2009, <http://www.rfc-editor.org/info/rfc5630>.

[RFC5888] Camarillo, G. and H. Schulzrinne, "The Session Description Protocol (SDP) Grouping Framework", RFC 5888, DOI 10.17487/RFC5888, June 2010, <http://www.rfc-editor.org/info/rfc5888>.

[RFC6184] Wang, Y., Even, R., Kristensen, T., and R. Jesup, "RTP Payload Format for H.264 Video", RFC 6184, DOI 10.17487/RFC6184, May 2011, <http://www.rfc-editor.org/info/rfc6184>.

[RFC7202] Perkins, C. and M. Westerlund, "Securing the RTP Framework: Why RTP Does Not Mandate a Single Media Security Solution", RFC 7202, DOI 10.17487/RFC7202, April 2014, <http://www.rfc-editor.org/info/rfc7202>.

[I-D.ietf-mmusic-sdp-bundle-negotiation]
Holmberg, C., Alvestrand, H., and C. Jennings, "Negotiating Media Multiplexing Using the Session Description Protocol (SDP)", draft-ietf-mmusic-sdp-bundle-negotiation-36 (work in progress), October 2016.

[I-D.ietf-rtcweb-data-channel]
Jesup, R., Loreto, S., and M. Tuexen, "WebRTC Data Channels", draft-ietf-rtcweb-data-channel-13 (work in progress), January 2015.

Authors' Addresses

Paul Kyzivat

Email: pkyzivat@alum.mit.edu

Lennard Xiao
Huawei

Email: lennard.xiao@huawei.com

Christian Groves
Huawei

Email: Christian.Groves@nteczone.com

Robert Hansen
Cisco Systems

Email: rohanse2@cisco.com