Internet Engineering Task Force (IETF)
Request for Comments: 8864
Category: Standards Track
ISSN: 2070-1721

K. Drage
M. Makaraju
R. Ejzak
J. Marcon
Unaffiliated
R. Even, Ed.

January 2021

Negotiation Data Channels Using the Session Description Protocol (SDP)

Abstract

Data channel setup can be done using either the in-band Data Channel Establishment Protocol (DCEP) or some out-of-band non-DCEP protocol. This document specifies how the SDP (Session Description Protocol) offer/answer exchange can be used to achieve an out-of-band non-DCEP negotiation for establishing a data channel.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

The concept of establishing a bidirectional data channel running on top of the Stream Control Transmission Protocol (SCTP) is discussed in [RFC8831], allowing applications to use data channels. An in-band Data Channel Establishment Protocol (DCEP) is described in [RFC8832]; however, other in-band or out-of-band protocols may be used for establishing data channels. Each data channel consists of paired SCTP streams sharing the same SCTP Stream Identifier. Data channels are created by endpoint applications using (1) the WebRTC API (Application Programming Interface) [WebRtcAPI] or (2) other protocols (e.g., Controlling Multiple Streams for Telepresence (CLUE) [RFC8850]). The protocols can be signaled by the data channel 'subprotocol' parameter, conceptually similar to a WebSocket subprotocol as described in [RFC6455]. However, apart from the "subprotocol" value transmitted to the peer, an endpoint application can agree on how to instantiate a given subprotocol on a data channel, and whether it is signaled in-band using DCEP or out-of-band using a non-DCEP protocol (or both).

This document defines Session Description Protocol (SDP) offer/answer procedures [RFC3264] that enable out-of-band negotiation for establishing data channels for transport of well-defined subprotocols. These procedures are based on generic SDP offer/answer negotiation rules for SCTP-based media transport as specified in [RFC8841] for the SDP "m=" line proto values UDP/DTLS/SCTP and TCP/DTLS/SCTP.

This document uses MSRP (the Message Session Relay Protocol) [RFC4975] and BFCP (the Binary Floor Control Protocol) [RFC8855] in several examples. It does not provide a complete specification of how to negotiate the use of a data channel to transport MSRP. Procedures specific to each subprotocol would have to be documented elsewhere. For MSRP, they are documented in [RFC8873]. The use of MSRP in some examples is only to show how the generic procedures described herein might apply to a specific subprotocol.

2. Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. Terminology

This document uses the following terms:

Data channel: A WebRTC data channel as specified in [RFC8831].

Data channel stack: An entity that, upon application request, runs the data channel protocol to keep track of states as well as the sending and receiving of data. If the application is a browser-based JavaScript application, then this stack resides in the browser. If the application is a native application, then this stack resides in the application and is accessible via some sort of API or APIs.

Data channel properties: Fixed properties assigned to a data channel at the time of its creation. Some of these properties determine the way the data channel stack transmits data on this channel (e.g., stream identifier, reliability, order of delivery).

Data channel subprotocol: The application protocol that is transported over a single data channel. Data channel subprotocol messages are sent as data channel payload over an established data channel. An SDP offer/answer exchange can be used as specified in this document to negotiate the establishment of data channels, corresponding data channel properties, associated data channel subprotocols, and data channel subprotocol properties. In this case, the data channel subprotocols may be identified by the values of the 'subprotocol' parameters of the SDP "a=dcmap:" attribute as described in Section 5.1.4. Within this document, the term "data channel subprotocol" is often abbreviated as just "subprotocol".

DCEP: Data Channel Establishment Protocol, as defined in [RFC8832].

In-band: Transmission through the peer-to-peer SCTP association.

Out-of-band: Transmission through the application signaling path.

Peer: From the perspective of one of the agents in a session, its peer is the other agent. Specifically, from the perspective of the SDP offerer, the peer is the SDP answerer. From the perspective of the SDP answerer, the peer is the SDP offerer.

SCTP Stream Sequence Number (SSN): The SCTP Stream Sequence Number, as specified in [RFC4960].

Stream identifier: The identifier of the outbound and inbound SCTP streams composing a data channel.

4. Applicability Statement

The mechanism described in this document only applies to SDP [RFC8866] when used together with the SDP offer/answer mechanism [RFC3264]. Declarative usage of SDP is out of scope for this document and is thus undefined.

5. SDP Data Channel Attributes

This section defines two new SDP media-level attributes that can be used together with the SDP Offer/Answer mechanism to negotiate data-channel-specific and subprotocol-specific parameters without the usage of DCEP [RFC8832]. The first attribute (Section 5.1) provides for negotiation of channel-specific parameters. The second attribute (Section 5.2) provides for negotiation of subprotocol-specific parameters.

> Note: Appendix A provides information regarding how data channels work in general. In particular, it summarizes some key aspects that should be considered for the negotiation of data channels if DCEP is not used.

5.1. SDP DCMAP Attribute

This section defines a new media-level attribute, "a=dcmap:", that defines the data channel parameters for each data channel to be negotiated.

This attribute is used to create bidirectional SCTP data channels having the same set of attributes. The data channel properties (reliable / partially reliable, ordered/unordered) need to be suitable per the subprotocol transport requirements.

5.1.1. DCMAP Attribute Syntax

"a=dcmap:" is a media-level attribute having the following definition and ABNF (Augmented Backus-Naur Form) syntax [RFC5234].

"a=dcmap:" Attribute	
Name	dcmap
Value	dcmap-value
Usage Level	media
Charset Dependent	No

Table 1: "a=dcmap:" Attribute Definition

Formal syntax:

```
dcmap-value     = dcmap-stream-id
                  [ SP dcmap-opt *(";" dcmap-opt) ]
dcmap-opt       = ordering-opt / subprotocol-opt / label-opt
                  / maxretr-opt / maxtime-opt / priority-opt
                  ; maxretr-opt and maxtime-opt are
                  ; mutually exclusive

dcmap-stream-id = 1*5DIGIT
ordering-opt    = "ordered=" ordering-value
ordering-value  = "true" / "false"
subprotocol-opt = "subprotocol=" quoted-string
label-opt       = "label=" quoted-string
maxretr-opt     = "max-retr=" maxretr-value
maxretr-value   = "0" / integer
                  ; number of retransmissions,
                  ; less than 2^32,
                  ; derived from 'Reliability Parameter' [RFC8832]
maxtime-opt     = "max-time=" maxtime-value
maxtime-value   = "0" / integer
                  ; milliseconds,
                  ; less than 2^32,
                  ; derived from 'Reliability Parameter' [RFC8832]
priority-opt    = "priority=" priority-value
priority-value  = "0" / integer
                  ; unsigned integer value indicating the priority of
                  ; the data channel,
                  ; less than 2^16,
                  ; derived from 'Priority' [RFC8832]

quoted-string   = DQUOTE *(quoted-char / escaped-char) DQUOTE
quoted-char     = SP / quoted-visible
quoted-visible  = %x21 / %x23-24 / %x26-7E ; VCHAR without " or %
```

```
escaped-char    = "%" HEXDIG HEXDIG
DQUOTE          = <from RFC 5234>
integer         = <from RFC 8866>
```

Examples:

```
a=dcmap:0
a=dcmap:1 subprotocol="bfcp";max-time=60000;priority=512
a=dcmap:2 subprotocol="msrp";ordered=true;label="msrp"
a=dcmap:3 label="Label 1";ordered=false;max-retr=5;priority=128
a=dcmap:4 label="foo%09bar";ordered=true;max-time=15000
```

> Note: The last example (a=dcmap:4) shows a 'label' parameter value that contains one nonprintable 'escaped-char' character (the tabulator character).

Within an "a=dcmap:" attribute line's 'dcmap-opt' value, only one 'maxretr-opt' parameter or one 'maxtime-opt' parameter may be present. Both parameters MUST NOT be present.

5.1.2. 'dcmap-stream-id' Parameter

The 'dcmap-stream-id' parameter indicates the SCTP stream identifier within the SCTP association used to form the data channel.

5.1.3. 'label' Parameter

The 'label' parameter indicates the name of the channel. It represents a label that can be used to distinguish, in the context of the WebRTC API [WebRtcAPI], an RTCDataChannel object from other RTCDataChannel objects. This parameter maps to the 'Label' parameter defined in [RFC8832]. The 'label' parameter is optional. If it is not present, then its value defaults to the empty string.

In order to communicate with the WebRTC API, the 'label' parameter should

* Serialize the WebRTC label as a UTF-8 string [RFC3629].

* Treat the UTF-8 serialization as a series of bytes.

* For each byte in the serialization,

   - If the byte can be expressed as a 'quoted-char', do so.

   - Otherwise, express the byte as an 'escaped-char'.

> Note: The empty string can also be explicitly used as a 'label' value, such that 'label=""' is equivalent to the 'label' parameter not being present at all. [RFC8832] allows the DATA_CHANNEL_OPEN message's 'Label' value to be an empty string.

5.1.4. 'subprotocol' Parameter

The 'subprotocol' parameter indicates which protocol the client expects to exchange via the channel. This parameter maps to the 'Protocol' parameter defined in [RFC8832]. Section 9.1 specifies how values for new subprotocol parameters are registered. 'subprotocol' is an optional parameter. If the 'subprotocol' parameter is not present, then its value defaults to an empty string.

> Note: The empty string can also be explicitly used as a 'subprotocol' value, such that 'subprotocol=""' is equivalent to the 'subprotocol' parameter not being present at all. [RFC8832] allows the DATA_CHANNEL_OPEN message's 'Protocol' value to be an empty string.

5.1.5. 'max-retr' Parameter

This parameter indicates that the data channel is partially reliable. The 'max-retr' parameter indicates the maximal number of times a user message will be retransmitted. The 'max-retr' parameter is optional. If the 'max-retr' parameter and the 'max-time' parameter are not present, then reliable transmission is performed as specified in [RFC4960]. This parameter maps to the 'Number of RTX' parameter defined in [RFC8832].

5.1.6. 'max-time' Parameter

This parameter indicates that the data channel is partially reliable. A user message will no longer be transmitted or retransmitted after a specified lifetime, given in milliseconds, in the 'max-time' parameter. The lifetime starts when providing the user message to the protocol stack. The 'max-time' parameter is optional. If the 'max-retr' parameter and the 'max-time' parameter are not present, then reliable transmission is performed as specified in [RFC4960]. This parameter maps to the 'Lifetime in ms' parameter defined in [RFC8832].

5.1.7. 'ordered' Parameter

The 'ordered' parameter with value "true" indicates that the receiver will dispatch DATA chunks in the data channel to the upper layer while preserving the order. The 'ordered' parameter is optional and takes two values -- "true" for ordered delivery and "false" for unordered delivery -- with "true" as the default value. Any other value is ignored, and the default "ordered=true" is assumed. In the absence of this parameter, "ordered=true" is assumed. This parameter maps to the ordered or unordered data channel types as defined in [RFC8832].

5.1.8. 'priority' Parameter

The 'priority' parameter indicates the data channel's priority relative to the priorities of other data channels, which may additionally exist over the same SCTP association. The 'priority' parameter maps to the 'Priority' parameter defined in [RFC8832]. The 'priority' parameter is optional. In the absence of this parameter, "priority=256" is assumed.

5.1.9. DCMAP Multiplexing Category

The multiplexing category [RFC8859] of the "a=dcmap:" attribute is SPECIAL.

As the usage of multiple SCTP associations on top of a single DTLS association is outside the scope of [RFC8841], no "a=dcmap:" attribute multiplexing rules are specified for the UDP/DTLS/SCTP and TCP/DTLS/SCTP proto values. If future extensions of [RFC8841] define how to negotiate multiplexing of multiple SCTP associations on top of a single DTLS association or how to add multiple SCTP associations to one BUNDLE group, then multiplexing rules for the "a=dcmap:" attribute need to be defined as well -- for instance, in an extension of this specification.

5.2. SDP DCSA Attribute

In the SDP media description, each data channel declaration MAY also be followed by other SDP attributes, which apply to the corresponding data channel and its subprotocol. Each of these attributes is represented by one new "a=dcsa:" attribute line that references another SDP attribute defined for use with this data channel's subprotocol. Instructions for registering attributes for use with a data channel are given in Section 9.3.

Each SDP attribute that is related to the subprotocol and that would normally be used to negotiate the subprotocol using the SDP offer/answer mechanism is replaced with an attribute of the form "a=dcsa:stream-id original-attribute", where "dcsa" stands for "data channel subprotocol attribute", "stream-id" is the SCTP stream identifier assigned to this subprotocol instance, and "original-attribute" represents the contents of the subprotocol-related attribute to be included.

The same syntax applies to any other SDP attribute required for negotiation of this instance of the subprotocol.

The detailed offer/answer procedures for the dcsa attribute are dependent on the associated subprotocol. If no offer/answer procedures exist for the subprotocol when used outside of the dcsa attribute, no specification is needed for use with dcsa. The IANA (Internet Assigned Numbers Authority) registration procedures for the "WebSocket Subprotocol Name Registry" (Section 9.1) do not strictly require a specification of the offer/answer procedures for the subprotocol when used with dcsa. If the subprotocol has defined offer/answer procedures when used outside of dcsa, such a specification is encouraged to ensure interoperability. If the subprotocol has defined offer/answer procedures when used outside of dcsa but no specification exists for the offer/answer procedures for the subprotocol when used with dcsa, implementations SHOULD assume the use of the default values for all otherwise-negotiable and applicable subprotocol parameters.

5.2.1. DCSA Attribute Syntax

"a=dcsa:" is a media-level attribute having the following definition and ABNF (Augmented Backus-Naur Form) syntax [RFC5234].

"a=dcsa:" Attribute	
Name	dcsa
Value	dcsa-value
Usage Level	media
Charset Dependent	No

Table 2: "a=dcsa:" Attribute Definition

Formal syntax:

```
dcsa-value      = stream-id SP attribute
stream-id       = 1*5DIGIT
attribute       = <from RFC 8866>
```

Example:

```
a=dcmap:2 subprotocol="msrp";ordered=true;label="msrp"

a=dcsa:2 accept-types:text/plain
```

The reference to [RFC8866] defines where the attribute definition can be found; it does not provide any limitations on support of attributes defined in other documents in accordance with this attribute definition. However, not all SDP attributes are suitable as an "a=dcsa:" parameter. The registry of IANA SDP parameters contains the lists of IANA-registered session-level and media-level or media-level-only SDP attributes.

Thus, in the example above, the original attribute line "a=accept-types:text/plain" is represented by the attribute line "a=dcsa:2 accept-types:text/plain", which specifies that this instance of the MSRP subprotocol being transported on the SCTP association using the data channel with stream id 2 accepts plaintext files.

As opposed to the data channel "a=dcmap:" attribute parameters, these parameters are subject to offer/answer negotiation, following the procedures defined in the subprotocol-specific documents.

It is assumed that in general the usages of subprotocol-related media-level attributes are independent from the subprotocol's transport protocol. Such transport-protocol-independent subprotocol-related attributes are used in the same way as defined in the original subprotocol specification, also if the subprotocol is transported over a data channel and if the attribute is correspondingly embedded in an "a=dcsa:" attribute.

There may be cases where the usage of a subprotocol-related media-level attribute depends on the subprotocol's transport protocol. In such cases, the subprotocol-related usage of the attribute is expected to be described for the data channel transport. A data-channel-specific usage of a subprotocol attribute is expected to be specified in the same document that registers the subprotocol's identifier for data channel usage as described in Section 9.1.

5.2.2. DCSA Multiplexing Category

The multiplexing category of the "a=dcsa:" attribute is SPECIAL.

As the usage of multiple SCTP associations on top of a single DTLS association is outside the scope of [RFC8841], no "a=dcsa:" attribute multiplexing rules are specified for the UDP/DTLS/SCTP and TCP/DTLS/SCTP proto values. If future extensions of [RFC8841] define how to negotiate multiplexing of multiple SCTP associations on top of a single DTLS association or how to add multiple SCTP associations to one BUNDLE group, then multiplexing rules for the "a=dcsa:" attribute need to be defined as well -- for instance, in an extension of this specification.

6. SDP Offer/Answer Procedures

This section defines how data channels can be negotiated using the SDP offer/answer mechanism. A given media description can describe multiple data channels (each represented by a separate SDP dcmap attribute) that can be created, modified, and closed using different offer/answer exchanges. The procedures in this section apply for a given data channel.

The generic offer/answer procedures for negotiating the SCTP association used to realize data channels are defined in [RFC8841]. This section only defines the data-channel-specific procedures.

"Initial offer" refers to the offer in which a data channel is opened. It can be either the initial offer or a subsequent offer of the associated SDP session.

The detailed offer/answer procedures for the dcsa attribute are dependent on the associated subprotocol; see Section 5.2.

6.1. Managing Stream Identifiers

In order to avoid SCTP Stream identifier collisions, in alignment with [RFC8832], the endpoint acting as a DTLS client (for the SCTP association used to realize data channels) MUST use even identifier values, and the endpoint acting as a DTLS server MUST use odd identifier values.

SCTP stream identifiers associated with data channels that have been negotiated using DCEP MUST NOT be included in SDP offers and answers.

6.2. Negotiating Data Channel Parameters

The data channel types defined in [RFC8832] are mapped to the dcmap SDP attribute parameters in the following manner, where "ordered=true" is the default and may be omitted:

```
DATA_CHANNEL_RELIABLE
   ordered=true

DATA_CHANNEL_RELIABLE_UNORDERED
   ordered=false

DATA_CHANNEL_PARTIAL_RELIABLE_REXMIT
   ordered=true;max-retr=<number of retransmissions>

DATA_CHANNEL_PARTIAL_RELIABLE_REXMIT_UNORDERED
   ordered=false;max-retr=<number of retransmissions>

DATA_CHANNEL_PARTIAL_RELIABLE_TIMED
   ordered=true;max-time=<lifetime in milliseconds>

DATA_CHANNEL_PARTIAL_RELIABLE_TIMED_UNORDERED
   ordered=false;max-time=<lifetime in milliseconds>
```

By definition, 'max-retr' and 'max-time' are mutually exclusive, so both MUST NOT be present in the "a=dcmap:" attribute line. If an SDP offer contains both of these parameters, then the receiver of such an SDP offer MUST reject the SDP offer. If an SDP answer contains both of these parameters, then the offerer MUST treat the associated SDP offer/answer as failed.

6.3. Generating the Initial Offer for a Data Channel

When an offerer sends an initial offer, in order to negotiate an SCTP stream for a data channel, the offerer

* SHALL include an SDP dcmap attribute (Sections 5.1 and 6.2) associated with the data channel in the "m=" section representing the SCTP association used to realize the data channel, and

* MAY include one or more SDP dcsa attributes (Section 5.2) associated with the data channel. The value of the 'stream-id' part of each attribute SHALL match the 'dcmap-stream-id' value of the dcmap attribute.

6.4. Generating the SDP Answer

When an answerer receives an offer that includes an "m=" section for an SCTP association, the offer describes an SCTP stream for a data channel, if the answerer accepts the data channel, it

* SHALL include an SDP dcmap attribute (Sections 5.1 and 6.2) associated with the data channel in the "m=" section representing the SCTP association used to realize the data channel. The value of the 'dcmap-stream-id', 'max-retr', and 'max-time' values of the dcmap attribute SHALL be identical to the value used for the data channel in the offer, and

* MAY include one or more SDP dcsa attributes (Section 5.2) associated with the data channel.

6.5. Offerer Processing of the SDP Answer

An offerer receiving an SDP answer performs the following:

* It SHALL close any created data channels as described in Section 6.6.1 for which the expected "a=dcmap:" attributes are not present in the SDP answer. If the SDP answer has no "a=dcmap:" attributes, either the peer does not support "a=dcmap:" attributes or it rejected all the data channels. In either case, the offerer closes all the data channels offered by SDP that were open at the time of the offer. The DTLS association and SCTP association will still be set up. At this point, the offerer may use DCEP negotiation [RFC8832] to open data channels.

Each agent application MUST wait to send data until it has confirmation that the data channel at the peer is instantiated. For WebRTC, this is when both data channel stacks have channel parameters instantiated and occurs as follows:

* At both peers when a data channel is created without a previously established SCTP association, as soon as the SCTP association is successfully established.

* At the agent receiving an SDP offer for which there is an established SCTP association, as soon as it creates the negotiated data channel based on information signaled in the SDP offer.

* At the agent sending an SDP offer to create a new data channel for which there is an established SCTP association, when it receives the SDP answer confirming acceptance of the data channel or when it begins to receive data on the data channel from the peer, whichever occurs first.

6.6. Modifying the Session

When an offerer sends a subsequent offer that includes information for a previously negotiated data channel, unless the offerer intends to close the data channel (Section 6.6.1), the offerer SHALL include the previously negotiated SDP attributes and attribute values associated with the data channel. The answerer may reject the offer. The means for rejecting an offer are dependent on the higher-layer protocol. The offer/answer exchange is atomic; if the answer is rejected, the session reverts to the state prior to the offer [RFC3264].

6.6.1. Closing a Data Channel

In order to close a data channel, the endpoint that wants to close the data channel SHALL send an SCTP SSN Reset message [RFC6525], following the procedure in Section 6.7 of [RFC8831]. In addition, if the closed data channel was negotiated using the offer/answer mechanism (Section 6.3), the endpoint that closed the data channel SHALL send a subsequent offer in which it does one of the following:

* Removes the SDP dcmap attribute and SDP dcsa attributes associated

with the closed data channel. Once the endpoint receives a successful answer, the SCTP stream identifier value can later be used for a new data channel (negotiated using either SCTP or the offer/answer mechanism), or

* After a reset has been performed, reuses the SCTP stream used for the closed data channel for a new data channel, following the procedure in Section 6.3. The offerer SHALL use a different SDP dcmap attribute value for the data channel using the same SCTP stream.

6.7. Various SDP Offer/Answer Considerations

An SDP offer or answer has no "a=dcmap:" attributes but has "a=dcsa:" attributes:

* This is considered an error case. In this case, the receiver of such an SDP offer or answer MUST discard the "a=dcsa:" attributes.

An SDP offer or answer has an "a=dcsa:" attribute whose subprotocol attribute is unknown:

* The receiver of such an SDP offer or answer SHOULD ignore this entire "a=dcsa:" attribute line.

An SDP offer or answer has an "a=dcsa:" attribute whose subprotocol attribute is known but whose subprotocol attribute semantic is not known for the data channel transport case:

* The receiver of such an SDP offer or answer SHOULD ignore this entire "a=dcsa:" attribute line.

7. Examples

Figure 1 shows an example of an SDP offer and answer where the SDP answerer rejects the data channel with stream id 0 either for explicit reasons or because it does not understand the "a=dcmap:" attribute. As a result, the offerer will close the data channel created with the SDP offer/answer negotiation option. The SCTP association will still be set up over DTLS. At this point, the offerer or the answerer may use DCEP negotiation to open data channels.

```
m=application 10001 UDP/DTLS/SCTP webrtc-datachannel
c=IN IP6 2001:db8::3
a=max-message-size:100000
a=sctp-port:5000
a=setup:actpass
a=fingerprint:SHA-1 \
    4A:AD:B9:B1:3F:82:18:3B:54:02:12:DF:3E:5D:49:6B:19:E5:7C:AB
a=tls-id:abc3de65cddef001be82
a=dcmap:0 subprotocol="bfcp";label="bfcp"
m=application 10002 UDP/DTLS/SCTP webrtc-datachannel
c=IN IP6 2001:db8::1
a=max-message-size:100000
a=sctp-port:5002
```

```
   a=setup:passive
   a=fingerprint:SHA-1 \
       5B:AD:67:B1:3E:82:AC:3B:90:02:B1:DF:12:5D:CA:6B:3F:E5:54:FA
   a=tls-id:dcb3ae65cddef0532d42
```

Figure 1: Example 1

Figure 2 shows an example of an SDP offer and answer where the SDP offer contains data channels for BFCP and MSRP subprotocols. The SDP answer rejects BFCP and accepts MSRP. So, the offerer closes the data channel for BFCP, and both the offerer and the answerer may start using the MSRP data channel (after the SCTP association is set up). The data channel with stream id 0 is free and can be used for future DCEP or SDP offer/answer negotiation.

```
   m=application 10001 UDP/DTLS/SCTP webrtc-datachannel
   c=IN IP4 192.0.2.1
   a=max-message-size:100000
   a=sctp-port:5000
   a=setup:actpass
   a=fingerprint:SHA-1 \
       4A:AD:B9:B1:3F:82:18:3B:54:02:12:DF:3E:5D:49:6B:19:E5:7C:AB
   a=tls-id:abc3de65cddef001be82
   a=dcmap:0 subprotocol="bfcp";label="bfcp"
   a=dcmap:2 subprotocol="msrp";label="msrp"
   a=dcsa:2 accept-types:message/cpim text/plain
   a=dcsa:2 path:msrp://alice.example.com:10001/2s93i93idj;dc
   m=application 10002 UDP/DTLS/SCTP webrtc-datachannel
   c=IN IP4 192.0.2.2
   a=max-message-size:100000
   a=sctp-port:5002
   a=setup:passive
   a=fingerprint:SHA-1 \
       5B:AD:67:B1:3E:82:AC:3B:90:02:B1:DF:12:5D:CA:6B:3F:E5:54:FA
   a=tls-id:dcb3ae65cddef0532d42
   a=dcmap:2 subprotocol="msrp";label="msrp"
   a=dcsa:2 accept-types:message/cpim text/plain
   a=dcsa:2 path:msrp://bob.example.com:10002/si438dsaodes;dc
```

Figure 2: Example 2

The example in Figure 3 is a continuation of the example in Figure 2. The SDP offerer now removes the MSRP data channel with stream id 2 but opens a new MSRP data channel with stream id 4. The answerer accepts the entire offer. As a result, the offerer closes the previously negotiated MSRP-related data channel, and both the offerer and the answerer may start using the new MSRP-related data channel.

```
   m=application 10001 UDP/DTLS/SCTP webrtc-datachannel
   c=IN IP4 192.0.2.1
   a=max-message-size:100000
   a=sctp-port:5000
   a=setup:actpass
   a=fingerprint:SHA-1 \
       4A:AD:B9:B1:3F:82:18:3B:54:02:12:DF:3E:5D:49:6B:19:E5:7C:AB
   a=tls-id:abc3de65cddef001be82
   a=dcmap:4 subprotocol="msrp";label="msrp"
   a=dcsa:4 accept-types:message/cpim text/plain
   a=dcsa:4 path:msrp://alice.example.com:10001/2s93i93idj;dc
   m=application 10002 UDP/DTLS/SCTP webrtc-datachannel
   c=IN IP4 192.0.2.2
   a=max-message-size:100000
   a=sctp-port:5002
   a=setup:passive
   a=fingerprint:SHA-1 \
       5B:AD:67:B1:3E:82:AC:3B:90:02:B1:DF:12:5D:CA:6B:3F:E5:54:FA
   a=tls-id:dcb3ae65cddef0532d42
   a=dcmap:4 subprotocol="msrp";label="msrp"
   a=dcsa:4 accept-types:message/cpim text/plain
   a=dcsa:4 path:msrp://bob.example.com:10002/si438dsaodes;dc
```

Figure 3: Example 3

8. Security Considerations

This document specifies new SDP attributes used in the negotiation of data channel parameters.

These parameters are negotiated as part of opening an SCTP channel over DTLS as specified in [RFC8841]. Each subprotocol may come with its own security considerations that need to be documented as part of the subprotocol definition. Otherwise, this document does not add any security considerations to those specified in [RFC8841].

Error cases such as the use of unknown parameter values or violations of the odd/even rule (Section 6.1) MUST be handled by closing the corresponding data channel.

9. IANA Considerations

9.1. Subprotocol Identifiers

Registration of new subprotocol identifiers is performed using the existing IANA "WebSocket Subprotocol Name Registry" table.

The following text has been added below the title of the table.

"This table also includes subprotocol identifiers specified for usage within a WebRTC data channel."

This document (RFC 8864) has been added to the "Reference" list for the registry.

This document assigns no new values to this table.

A subprotocol may simultaneously be defined for data channel transport and for WebSocket transport. In such a case, the "Subprotocol Definition" and "Reference" cells in the subprotocol's row of the IANA "WebSocket Subprotocol Name Registry" table should contain two entries. One entry in each of these cells should refer to the WebSocket-related subprotocol specification, and the other entry should refer to the data-channel-related subprotocol specification.

9.2. New SDP Attributes

9.2.1. dcmap

This document defines a new SDP media-level attribute, "a=dcmap:", as follows:

"a=dcmap:"	
Contact name	IESG
Contact email	iesg@ietf.org
Attribute name	dcmap
Attribute syntax	As per Section 5.1.1
Attribute semantics	As per Section 5.1.1
Usage level	media
Charset dependent	No
Purpose	To define data-channel-specific parameters
Appropriate values	As per Section 5.1.1
O/A procedures	SDP offer/answer procedures as per Section 6
Mux category	SPECIAL. See Section 5.1.9
Reference	RFC 8864

Table 3: New "a=dcmap:" Attribute

9.2.2. dcsa

This document defines a new SDP media-level attribute, "a=dcsa:", as follows:

"a=dcsa:"	
Contact name	IESG
Contact email	iesg@ietf.org
Attribute name	dcsa
Attribute syntax	As per Section 5.2.1

Attribute semantics	As per Section 5.2.1
Usage level	media
Charset dependent	No
Purpose	To define attributes that are specific to data channel subprotocols
Appropriate values	As per Section 5.2.1
O/A procedures	SDP offer/answer procedures as per Section 6
Mux category	SPECIAL. See Section 5.2.2
Reference	RFC 8864

Table 4: New "a=dcsa:" Attribute

9.3. Registering Attributes for Use with Data Channels

When a subprotocol is defined for use over data channels with the SDP offer/answer mechanism, any SDP attributes that may be negotiated using the "a=dcsa:" attribute MUST be added to the IANA "attribute-name registry (formerly "att-field")", as specified in [RFC8866], Section 8.2.4. This document specifies that new Usage Levels of the form "dcsa (foo)" (where "foo" is a placeholder for the subprotocol name) should be registered by documents that specify negotiation of particular subprotocols.

IANA has updated the "attribute-name (formerly "att-field")" registry to point to this document.

10. References

10.1. Normative References

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

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

[RFC3629] Yergeau, F., "UTF-8, a transformation format of ISO 10646", STD 63, RFC 3629, DOI 10.17487/RFC3629, November 2003, <https://www.rfc-editor.org/info/rfc3629>.

[RFC4960] Stewart, R., Ed., "Stream Control Transmission Protocol", RFC 4960, DOI 10.17487/RFC4960, September 2007, <https://www.rfc-editor.org/info/rfc4960>.

[RFC5234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, DOI 10.17487/RFC5234, January 2008, <https://www.rfc-editor.org/info/rfc5234>.

[RFC6525] Stewart, R., Tuexen, M., and P. Lei, "Stream Control Transmission Protocol (SCTP) Stream Reconfiguration", RFC 6525, DOI 10.17487/RFC6525, February 2012, <https://www.rfc-editor.org/info/rfc6525>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8831] Jesup, R., Loreto, S., and M. Tüxen, "WebRTC Data Channels", RFC 8831, DOI 10.17487/RFC8831, January 2021, <https://www.rfc-editor.org/info/rfc8831>.

[RFC8832] Jesup, R., Loreto, S., and M. Tüxen, "WebRTC Data Channel Establishment Protocol", RFC 8832, DOI 10.17487/RFC8832, January 2021, <https://www.rfc-editor.org/info/rfc8832>.

[RFC8841] 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", RFC 8841, DOI 10.17487/RFC8841, January 2021, <https://www.rfc-editor.org/info/rfc8841>.

[RFC8859] Nandakumar, S., "A Framework for Session Description Protocol (SDP) Attributes When Multiplexing", RFC 8859, DOI 10.17487/RFC8859, January 2021, <https://www.rfc-editor.org/info/rfc8859>.

[RFC8866] Begen, A., Kyzivat, P., Perkins, C., and M. Handley, "SDP: Session Description Protocol", RFC 8866, DOI 10.17487/RFC8866, January 2021, <https://www.rfc-editor.org/info/rfc8866>.

10.2. Informative References

[RFC4975] Campbell, B., Ed., Mahy, R., Ed., and C. Jennings, Ed., "The Message Session Relay Protocol (MSRP)", RFC 4975, DOI 10.17487/RFC4975, September 2007, <https://www.rfc-editor.org/info/rfc4975>.

[RFC6455] Fette, I. and A. Melnikov, "The WebSocket Protocol", RFC 6455, DOI 10.17487/RFC6455, December 2011, <https://www.rfc-editor.org/info/rfc6455>.

[RFC8850] Holmberg, C., "Controlling Multiple Streams for Telepresence (CLUE) Protocol Data Channel", RFC 8850, DOI 10.17487/RFC8850, January 2021, <https://www.rfc-editor.org/info/rfc8850>.

[RFC8855] Camarillo, G., Drage, K., Kristensen, T., Ott, J., and C. Eckel, "The Binary Floor Control Protocol (BFCP)", RFC 8855, DOI 10.17487/RFC8855, January 2021, <https://www.rfc-editor.org/info/rfc8855>.

[RFC8873] Recio, JM., Ed. and C. Holmberg, "Message Session Relay Protocol (MSRP) over Data Channels", RFC 8873, DOI 10.17487/RFC8873, January 2021, <https://www.rfc-editor.org/info/rfc8873>.

[T38] International Telecommunication Union, "Procedures for real-time Group 3 facsimile communication over IP networks", ITU-T Recommendation T.38, November 2015, <https://www.itu.int/rec/T-REC-T.38-201511-I/en>.

[WebRtcAPI] Jennings, C., Boström, H., and J-I. Bruaroey, "WebRTC 1.0: Real-time Communication Between Browsers", W3C Proposed Recommendation, <https://www.w3.org/TR/webrtc/>.

Appendix A. Generic Data Channel Negotiation Aspects when Not Using DCEP

This appendix summarizes how data channels work in general and discusses some key aspects that should be considered for the out-of-band negotiation of data channels if DCEP is not used.

A WebRTC application creates a data channel by providing a number of setup parameters (subprotocol, label, maximal number of retransmissions, maximal retransmission time, order of delivery, priority). The application also specifies whether it wants to make use of the negotiation using DCEP [RFC8832] or intends to negotiate data channels using the SDP offer/answer protocol.

In any case, the SDP offer generated by the application is per [RFC8841]. In brief, it contains one "m=" line for the SCTP association on top of which the data channels will run:

```
m=application 54111 UDP/DTLS/SCTP webrtc-datachannel
c=IN IP4 192.0.2.1
a=max-message-size:100000
a=sctp-port:5000
a=tls-id:abc3de65cddef001be82
a=setup:actpass
a=fingerprint:SHA-1 \
    4A:AD:B9:B1:3F:82:18:3B:54:02:12:DF:3E:5D:49:6B:19:E5:7C:AB
```

> Note: A WebRTC application will only use the "m=" line format "webrtc-datachannel" and will not use other formats in the "m=" line for other protocols such as T.38 [T38]. [RFC8841] supports only one SCTP association to be established on top of a DTLS association.

> Note: The above SDP media description does not contain any channel-specific information.

A.1. Stream Identifier Numbering

Independently from the requested type of negotiation, the application creating a data channel can either (1) pass the stream identifier to the data channel stack to assign to the data channel or (2) let the data channel stack pick one identifier from the unused ones.

Moreover, to avoid glare situations [RFC3264], each endpoint can own an exclusive set of stream identifiers, in which case an endpoint can only create a data channel with a stream identifier it owns.

Which set of stream identifiers is owned by which endpoint is determined by convention or other means.

> Note: For data channels negotiated with DCEP, one endpoint owns by convention the even stream identifiers, whereas the other owns the odd stream identifiers, as defined in [RFC8832].

> Note: For data channels negotiated via a protocol other than DCEP, no convention is defined by default.

A.2. Generic Data Channel Negotiation Not Using DCEP

A.2.1. Overview

DCEP negotiation only provides for negotiation of data channel transport parameters and does not provide for negotiation of subprotocol-specific parameters. Non-DCEP data channel negotiation can be defined to allow negotiation of parameters beyond those handled by DCEP, e.g., parameters specific to the subprotocol instantiated on a particular data channel.

The following procedures are common to all methods of data channel negotiation not using DCEP, whether in-band (communicated using proprietary means on an already-established data channel) or out-of-band (using the SDP offer/answer mechanism or some other protocol associated with the signaling channel).

A.2.2. Opening a Data Channel

In the case of non-DCEP negotiation, the endpoint application has the option to fully control the stream identifier assignments. However, these assignments have to coexist with the assignments controlled by the data channel stack for data channels negotiated using DCEP (if any). It is the responsibility of the application to ensure consistent assignment of stream identifiers.

When the application requests that the creation of a new data channel be set up via non-DCEP negotiation, the data channel stack creates the data channel locally without sending any DATA_CHANNEL_OPEN messages in-band. However, even if the ICE (Interactive Connectivity Establishment), DTLS, and SCTP procedures were already successfully completed, the application can't send data on this data channel until the negotiation with the peer is complete. This is because the peer needs to be aware of and accept the usage of this data channel. The peer, after accepting the data channel offer, can start sending data immediately. This implies that the offerer may receive data channel subprotocol messages before the negotiation is complete, and the application should be ready to handle it.

If the peer rejects the data channel part of the offer, then it doesn't have to do anything, as the data channel was not created using the stack. The offerer, on the other hand, needs to close the data channel that was opened by invoking relevant data channel stack API procedures.

It is also worth noting that a data channel stack implementation may not provide any APIs to create and close data channels; instead, the data channels may be used on the fly as needed, just by communicating via non-DCEP means or even by having some local configuration/ assumptions on both of the peers.

The application then negotiates the data channel properties and subprotocol properties with the peer's application using a mechanism different from DCEP.

The peer then symmetrically creates a data channel with these negotiated data channel properties. This is the only way for the peer's data channel stack to know which properties to apply when transmitting data on this channel. The data channel stack must allow data channel creation with any nonconflicting stream identifier so that both peers can create the data channel with the same stream identifier.

A.2.3. Closing a Data Channel

When the application requests the closing of a data channel negotiated without DCEP, the data channel stack always performs an SCTP SSN Reset for this channel.

Depending upon the method used for non-DCEP negotiation and the subprotocol associated with the data channel, the closing of the data channel might also be signaled to the peer via SDP offer/answer negotiation.

Acknowledgements

The authors wish to acknowledge the borrowing of ideas from other draft documents by Salvatore Loreto, Gonzalo Camarillo, Peter Dunkley, and Gavin Llewellyn. The authors also wish to thank Flemming Andreasen, Christian Groves, Gunnar Hellström, Paul Kyzivat, Jonathan Lennox, Uwe Rauschenbach, and Roman Shpount for their invaluable comments.

Special thanks to Christer Holmberg for helping finish the document and cleaning up Section 6.

Contributors

Juergen Stoetzer-Bradler made significant contributions to this document and should be considered a coauthor.

Authors' Addresses

Keith Drage
Unaffiliated

Email: drageke@ntlworld.com

Maridi R. Makaraju (Raju)
Unaffiliated

Email: mmraju@gmail.com

Richard Ejzak
Unaffiliated

Email: richard.ejzak@gmail.com

Jerome Marcon
Unaffiliated

Email: jeromee.marcon@free.fr

Roni Even (editor)

Email: ron.even.tlv@gmail.com