AVT Working Group
Internet-Draft
Intended status: Experimental
Expires: April 28, 2011

V. Singh
T. Karkkainen
J. Ott
S. Ahsan
Aalto University
L. Eggert
Nokia

October 25, 2010

Multipath RTP (MPRTP)
draft-singh-avt-mprtp-01

Abstract

The Real-time Transport Protocol (RTP) is used to deliver real-time content and, along with the RTP Control Protocol (RTCP), forms the control channel between the sender and receiver. However, RTP and RTCP assume a single delivery path between the sender and receiver and make decisions based on the measured characteristics of this single path. Increasingly, endpoints are becoming multi-homed, which means that they are connected via multiple Internet paths. Network utilization can be improved when endpoints use multiple parallel paths for communication. The resulting increase in reliability and throughput can also enhance the user experience. This document extends the Real-time Transport Protocol (RTP) so that a single session can take advantage of the availability of multiple paths between two endpoints.

Status of this Memo

Copyright Notice

Table of Contents

1. Introduction

Multi-homed endpoints are becoming common in today’s Internet, e.g., devices that support multiple wireless access technologies such as 3G and Wireless LAN. This means that often there is more than one network path available between two endpoints. Transport protocols, such as RTP, have not been designed to take advantage of the availability of multiple concurrent paths and therefore cannot benefit from the increased capacity and reliability that can be achieved by pooling their respective capacities.

Multipath RTP (MPRTP) is an OPTIONAL extension to RTP [1] that allows splitting a single RTP stream into multiple subflows that transmit over different paths. In effect, this pools the resource capacity of multiple paths.

Other IETF transport protocols that are capable of using multiple paths include SCTP [7], MPTCP MPTCP [8] and SHIM6 [9]. However, these protocols are not suitable for realtime communications.

1.1. Requirements Language

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 [2].

1.2. Terminology

- Endpoint: host either initiating or terminating an RTP connection.
- Interface: A logical or physical component that is capable of acquiring a unique IP address.
- Path: sequence of links between a sender and a receiver. Typically, defined by a set of source and destination addresses.
- Subflow: A flow of RTP packets along a specific path, i.e., a subset of the packets belonging to an RTP stream. The combination of all RTP subflows forms the complete RTP stream.

1.3. Use cases

The primary use case for MPRTP is transporting high bit-rate streaming multimedia content between endpoints, where at least one is multi-homed. Such endpoints could be residential IPTV devices that connect to the Internet through two different Internet service providers (ISPs), or mobile devices that connect to the Internet through 3G and WLAN interfaces. By allowing RTP to use multiple paths for transmission, the following gains can be achieved:

- Higher quality: Pooling the resource capacity of multiple Internet paths allows higher bit-rate and higher quality codecs to be used. From the application perspective, the available bandwidth between the two endpoints increases.

- Load balancing: Transmitting one RTP stream over multiple paths can reduce the bandwidth usage, compared to transmitting the same stream along a single path. This reduces the impact on other traffic.

- Fault tolerance: When multiple paths are used in conjunction with redundancy mechanisms (FEC, re-transmissions, etc.), outages on one path have less impact on the overall perceived quality of the stream.

A secondary use case for MPRTP is transporting Voice over IP (VoIP) calls to a device with multiple interfaces. Again, such an endpoint could be a mobile device with multiple wireless interfaces. In this case, little is to be gained from resource pooling, i.e., higher capacity or load balancing, because a single path should be easily capable of handling the required load. However, using multiple concurrent subflows can improve fault tolerance, because traffic can shift between the subflows when path outages occur. This results in very fast transport-layer handovers that do not require support from signaling.

2. Goals

This section outlines the basic goals that multipath RTP aims to meet. These are broadly classified as Functional goals and Compatibility goals.

2.1. Functional goals

Allow unicast RTP session to be split into multiple subflows in order to be carried over multiple paths. This may prove beneficial in case of video streaming.

- Increased Throughput: Cumulative capacity of the two paths may meet the requirements of the multimedia session. Therefore, MPRTP MUST support concurrent use of the multiple paths.

- Improved Reliability: MPRTP SHOULD be able to send redundant or re-transmit packets along any available path to increase reliability.

The protocol SHOULD be able to open new subflows for an existing session when new paths appear and MUST be able to close subflows when paths disappear.

2.2. Compatibility goals

MPRTP MUST be backwards compatible; an MPRTP stream needs to fall back to be compatible with legacy RTP stacks if MPRTP support is not successfully negotiated.

- o Application Compatibility: MPRTP service model MUST be backwards compatible with existing RTP applications, i.e., an MPRTP stack MUST be able to work with legacy RTP applications and not require changes to them. Therefore, the basic RTP APIs MUST remain unchanged, but an MPRTP stack MAY provide extended APIs so that the application can configure any additional features provided by the MPRTP stack.

- o Network Compatibility: individual RTP subflows MUST themselves be well-formed RTP flows, so that they are able to traverse NATs and firewalls. This MUST be the case even when interfaces appear after session initiation. Interactive Connectivity Establishment (ICE) [3] MAY be used for discovering new interfaces or performing connectivity checks.

3. RTP Topologies

RFC 5117 [10] describes a number of scenarios using mixers and translators in single-party (point-to-point), and multi-party (point-to-multipoint) scenarios. RFC 3550 [1] (Section 2.3 and 7.x) discuss in detail the impact of mixers and translators on RTP and RTCP packets. MPRTP assumes that if a mixer or translator exists in the network, then either all of the multiple paths or none of the multiple paths go via this component.

4. MPRTP Architecture

In a typical scenario, an RTP session uses a single path. In an MPRTP scenario, an RTP session uses multiple subflows that each use a different path. Figure 1 shows the difference.

```
+--------------+                Signaling           +--------------+
|              |----------------------------------->|              |
|    Client    |<-----------------------------------|    Server    |
|              |             Single RTP flow        |              |
+--------------+                                    +--------------+

+--------------+              Signaling             +--------------+
|              |----------------------------------->|              |
|    Client    |<-----------------------------------|    Server    |
|              |<-----------------------------------|              |
+--------------+            MPRTP sub-flows         +--------------+
```

Figure 1: Comparison between traditional RTP streaming and MPRTP

```
+-----------------------+         +-------------------------------+
|      Application      |         |          Application          |
+-----------------------+         +-------------------------------+
|                       |         |             MPRTP             |
+          RTP          +         +- - - - - - - -+- - - - - - - -+
|                       |         | RTP subflow   | RTP subflow   |
+-----------------------+         +---------------+---------------+
|          UDP          |         |      UDP      |      UDP      |
+-----------------------+         +---------------+---------------+
|           IP          |         |       IP      |       IP      |
+-----------------------+         +---------------+---------------+
```

Figure 2: MPRTP Architecture

Figure 2 illustrates the differences between the standard RTP stack and the MPRTP stack. MPRTP receives a normal RTP session from the application and splits it into multiple RTP subflows. Each subflow is then sent along a different path to the receiver. To the network, each subflow appears as an independent, well-formed RTP flow. At the receiver, the subflows are combined to recreate the original RTP session. The MPRTP layer performs the following functions:

o Path Management: The layer is aware of alternate paths to the peer, which may, for example, be the peer's multiple interfaces to send differently marked packets along separate paths. MPRTP also selects interfaces to send and receive data. Furthermore, it manages the port and IP address pair bindings for each interface.

o Packet Scheduling: the layer splits a single RTP flow into multiple subflows and sends them across multiple interfaces (paths). The splitting MAY BE done using different path characteristics.

- o Subflow recombination: the layer creates the original stream by recombining the independent subflows. Therefore, the multipath subflows appear as a single RTP stream to applications.

4.1. Relationship of MPRTP and Session Signaling

Session signaling (e.g., SIP[11], RTSP [12]) SHOULD be done over failover-capable or multipath-capable transport for e.g., SCTP [7] or MPTCP [8] instead of TCP or UDP.

5. Example Media Flow Diagrams

There may be many complex technical scenarios for MPRTP, however, this memo only considers the following two scenarios: 1) an unidirectional media flow that represents the streaming use case, and 2) a bidirectional media flow that represents a conversational use case.

5.1. Streaming Use Case

In the unidirectional scenario, the receiver (client) initiates a multimedia session with the sender (server). The receiver or the sender may have multiple interfaces and both the endpoints are MPRTP-capable. Figure 3 shows this scenario. In this case, host A has multiple interfaces. Host B performs connectivity checks on host A’s multiple interfaces. If the interfaces are reachable, then host B streams multimedia data along multiple paths to host A. Furthermore, host B splits the multimedia stream into two subflows based on the individually measured path characteristics.

Alternatively, to reduce media startup time, host B may start streaming multimedia data to host A’s initiating interface and then perform connectivity checks for the other interfaces. This method of updating a single path session to a multipath session is called "multipath session upgrade".

```
             Host A                           Host B
     -----------------------                ----------
     Address A1   Address A2                Address B1
     -----------------------                ----------
         |            Session Setup                 |
         |-------------------------------------->|  connection for the
         |<--------------------------------------|  peers may be "preloaded"
         |            |                          |  (e.g., with ICE)
         |            |                          |
         |       (RTP data B1->A1, B1->A2)       |
         |<======================================|
         |            |<=========================|
         |            |                          |
         Note: there maybe more scenarios.
```

Figure 3: Unidirectional media flow

5.2. Conversational Use Case

In the bidirectional scenario, multimedia data flows in both directions. The two hosts exchange their lists of interfaces with each other and perform connectivity checks. Communication begins after each host finds suitable address, port pairs. All interfaces that receive data send back RTCP receiver statistics for each path. The peers balance their own multimedia stream over multiple links based on the reception statistics from its peer and its own volume of traffic. Figure 4 describes an example of a bidirectional flow.

```
           Host A                               Host B
   -----------------------              -----------------------
   Address A1   Address A2              Address B1   Address B2
   -----------------------              -----------------------
       |            |                       |            |
       |            Session Setup           |            | connection for
       |---------------------------------->|            | the peers may
       |<-----------------------------------|            | be "preloaded"
       |            |                       |            | (e.g., with ICE)
       |            |                       |            |
       |    (RTP data B1<->A1, B2<->A2)     |            |
       |<===================================|            |
       |            |<===================================|
       |===================================>|            |
       |            |===================================>|
       |            |                       |            |
       Note: the address pairs may have other permutations,
       and they maybe symmetric or asymmetric combinations.
```

Figure 4: Bidirectional flow

5.3. Challenges with Multipath Interface Discovery

For some applications, where the user expects immediate playback, e.g., High Definition Media Streaming or IPTV, it may not be possible to perform connectivity checks within the given time bound. In these cases, connectivity checks MAY need to be done ahead of time.

[Editor: ICE or any other system would need to aware of the peer's interfaces ahead of time].

6. MPRTP Functional Blocks

This section describes some of the functional blocks needed for MPRTP. We then investigate each block and consider available mechanisms in the next section.

1. Session Setup: Multipath session setup is an upgrade or add-on to a typical RTP session. Interfaces may appear or disappear at anytime during the session. To preserve backward compatibility with legacy applications, a multipath session MUST look like a bundle of individual RTP sessions.

2. Expanding RTP: For a multipath session, each subflow MUST look like an independent RTP flow, so that individual RTCPs can be generated per subflow. Furthermore, MPRTP splits the single multimedia stream into multiple subflows based on path characteristics and dynamically adjusts the load on each link.

3. Adding Interfaces: Interfaces on the host need to be regularly discovered and signaled. This can be done at the session setup and/or during the session. When discovering and receiving new interfaces, the MPRTP layer needs to select address and port pairs.

4. Expanding RTCP: MPRTP MUST recombine RTCP reports from each path to re-create a single RTCP message to maintain backward compatibility with legacy applications.

5. Maintenance and Failure Handling: In a multi-homed endpoint interfaces may appear and disappear. If this happens at the sender, it has to re-adjust the load on the available links. On the other hand, if this occurs on the receiver, then the multimedia data transmitted by the sender to those interfaces is lost. This data may be re-transmitted along a different path i.e., to a different interface on the receiver. Furthermore, the receiver has to explicitly signal the disappearance of an interface, or the sender has to detect it. What happens if the

interface that setup the session disappears? does the control channel also failover? re-start the session?

6. Teardown: The MPRTP layer releases the occupied ports on the interfaces.

7. Available Mechanisms Within the Functional Blocks

This section discusses some of the possible alternatives for each functional block mentioned in the previous section.

7.1. Session Setup

MPRTP session can be set up in many possible ways e.g., during handshake, or upgraded mid-session. The capability exchange may be done using out-of-band signaling (e.g., SDP[13] in SIP[11], RTSP [12]) or in-band signaling (e.g., RTP/RTCP header extension). Furthermore, ICE [3] may be used for discovering and performing connectivity checks during session setup.

7.2. Expanding RTP

RTCP [1] is generated per media session. However, with MPRTP, the media sender spreads the RTP load across several interfaces. The media sender SHOULD make the path selection, load balancing and fault tolerance decisions based on the characteristics of each path. Therefore, apart from normal RTP sequence numbers defined in [1], the MPRTP sender SHOULD add subflow-specific sequence numbers and RTP timestamps to help calculate fractional losses, jitter, RTT, playout time, etc., for each path. An example RTP header extension for MPRTP is shown in Section 8.5).

7.3. Adding New Interfaces

When interfaces appear and disappear mid-session, ICE [3] may be used for discovering interfaces and performing connectivity checks. However, MPRTP may require a capability re-negotiation (using SDP) to include all these new interfaces. This method is referred to as out-of-band multipath advertisement.

Alternatively, when new interfaces appear the interface advertisements may be done in-band using RTP/RTCP extensions. The peers perform connectivity checks (see Figure 5 for more details). If the connectivity packets are received by the peers, then multimedia data can flow between the new address, port pairs.

7.4. Expanding RTCP

Multiple subflows in MPRTP affect RTCP bandwidth and RTCP reporting interval calculations. RTCP report scheduling for each subflow may cause a problem for RTCP recombined and reconstruction in cases when 1) RTCP for a subflow is lost, and 2) RTCP for a subflow arrives slower than other subflows. (There maybe other cases as well.)

The subflow RTCP RR reports at the sender help balance the load along each path. However, this document doesn't cover algorithms for congestion control or load balancing.

7.5. Checking and Failure Handling

[Editor:If the original interface that setup the session disappears then does the session signaling failover to another interface? Can we recommend that SIP/RTSP be run over MPTCP, SCTP].

8. MPRTP Protocol

To provide a more concrete basis for discussion, in this section we illustrate a solution. To enable a quick start to a multimedia session, we presume that a multipath session SHOULD be upgraded from a single path session. Therefore, no explicit changes are needed in multimedia session setup and the session can be setup as before.

8.1. MPRTP Session Establishment

Initially, the session is set up as a standard single path multimedia session. The bullet points below explain the different steps shown in Figure 5.

(1) The first two interactions between the hosts describes the standard session setup. This may be SIP or RTSP.

(2) Following the setup, like in a conventional RTP scenario, host B using interface B1 starts to stream data to host A at interface A1.

(3) Host B is an MPRTP-capable media sender and becomes aware of another interface B2.

(4) Host B advertises the multiple interface addresses using an RTP header extensions.

(5) Host A is an MPRTP-capable media receiver and becomes aware of another interface A2. It advertises the multiple interface

addresses using an RTCP RR extension.

Side note, if the MPRTP-capable hosts have no additional interfaces, then the hosts SHOULD still advertise a single interface.

(6) Each hosts receives information about the additional interfaces and perform connectivity tests (not shown in figure) and if the paths are reachable then the hosts start to stream the multimedia content using the additional paths.

```
           Host A                                    Host B
   -----------------------                  -----------------------
   Address A1   Address A2                  Address B1   Address B2
   -----------------------                  -----------------------
       |            |                           |             |
       |            |      (1)                  |             |
       |--------------------------------------->|             |
       |<---------------------------------------|             |
       |            |      (2)                  |             |
       |<=======================================|             |
       |<=======================================|     (3)     |
       |            |      (4)                  |             |
       |<=======================================|             |
       |<=======================================|             |
       |<=======================================|             |
       |            |      (5)                  |             |
       |- - - - - - - - - - - - - - - - - - - ->|             |
       |<=======================================|     (6)     |
       |<=======================================|             |
       |            |<========================================|
       |<=======================================|             |
       |            |<========================================|

   Key:
   |  Interface
   ---> Signaling Protocol
   <=== RTP Packets
   - -> RTCP Packet
```

Figure 5: MPRTP New Interface

8.1.1. Subflow or Interface Advertisement

MPRTP-capable media senders SHOULD use the RTP header extension defined in Figure 7 to advertise their interfaces. Each unique address is encapsulated in a Interface Advertisement block and contains the IP address, RTP port, and RTCP port addresses. The Interface Advertisement blocks are ordered based on decreasing priority level.

On receiving the MPRTP Interface Advertisement, the receiver will either ignore the RTP header extension if it is not MPRTP capable or MUST respond with its own set of interfaces in decreasing order of priority. If the sender and receiver are MPRTP-capable but have only one interface, then they MUST respond with the default interface address, RTP and RTCP port addresses. If the sender receives an RTCP report without the MPRTP RTCP block after advertising its interfaces, then the sender MUST presume that the receiver is not MPRTP capable. Figure 9 illustrates an RTCP format for MPRTP Interface Advertisement.

8.1.2. Path selection

After MPRTP support has been discovered and interface advertisements have been exchanged, the sender MUST initiate connectivity checks to determine which interface pairs offer valid paths between the sender and the receiver. To initiate a connectivity check, the sender sends an RTP packet with MPRTP extension header with MPR_Type = 0x02 and no RTP payload. The receiver replies with an MPRTP RTCP packet with type MPRR_Type = 0x02. After the sender receives the reply, the path is considered a valid candidate for subflow establishment.

The sender MAY choose to do any number of connectivity checks for any interface pairs at any point in a session.

8.1.3. Opening subflows

The sender may open any number of subflows after performing connectivity checks. MPRTP MUST associate a Flow ID to each subflow. To open a new subflow, the sender simply starts sending the RTP packets with an MPRTP extension shown in Figure 6. The MPRTP extension carries a mapping of a subflow packet to the aggregate flow. Namely, sequence numbers and timestamps associated to the subflow.

8.2. Packet Transmission

The MPRTP layer SHOULD associate an RTP packet to a subflow based on a scheduling strategy. The scheduling strategy may either choose to augment the paths to create higher throughput or use the alternate paths for enhancing resilience or error-repair. Due to the changing path characteristics, an MPRTP sender might change its scheduling strategy during an ongoing session. The MPRTP sender MUST also populate the flow specific fields described in the MPRTP extension header (see Section 8.5.1).

8.3. Playout Considerations at the Receiver

A media receiver, irrespective of MPRTP support or not, should be able to playback the media stream because the received RTP packets are compliant to [1], i.e., a non-MPRTP receiver will ignore the MPRTP header and still be able to playback the RTP packets. However, the variation of jitter and loss per path may affect proper playout. The receiver can compensate for the jitter by modifying the playout delay (adaptive playout) of the received RTP packets.

8.4. Flow specific RTCP Statistics and RTCP Aggregation

The aggregate RTCP report may not provide sufficient per path information to an MPRTP scheduler. Specifically, the scheduler should be aware of each path’s RTT, which an aggregate RTCP cannot provide.

[Editor: 1) Should the RTCP RRs sent per path carry a) the aggregate and the path’s RR or b) the aggregate and RR of each path.

2) Should the per path RTCP Interval be dependent on the overall session bitrate or per path interval receiver rate?]

8.5. Packet Format

In this sub-section we define the protocol structures described in the previous sections.

8.5.1. MPRTP RTP Header Extension

The MPRTP header extension is used 1) to pack single stream RTP data into multiple subflows, 2) to advertise the multiple interface addresses for a media sender, and 3) perform connectivity check on the new interfaces.

MPRTP RTP header extension for a subflow:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|V=2|P|X|  CC   |M|     PT      |       sequence number         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           timestamp                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           synchronization source (SSRC) identifier            |
+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
|            contributing source (CSRC) identifiers             |
|                             ....                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  RTP H-Ext ID |     length    |         MPR_Type=0x00         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|            flow ID            | flow specific sequence number |
+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
|                          RTP payload                          |
|                             ....                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 6: MPRTP header for subflow

RTP H-Ext ID and length: 8-bits each

It conforms to the 2-byte RTP header extension defined in [4].

RTP H-Ext=TBD

The 8-bit length field is the length of extension data in bytes not including the RTP H-Ext ID and length fields. The value zero indicates there is no data following.

MPR_Type: 16-bits

The MPR_Type field corresponds to the type of RTP packet. Namely:

0x00: Subflow RTP Header

0x01: Interface Advertisement

0x02: Connectivity Check

Flow ID: Identifier of the subflow. Every RTP packet belonging to the same subflow carries the same unique flow identifier.

Flow specific Sequence No.: Sequence of the packet in the subflow. Each subflow has its own strictly monotonically increasing sequence number space.

MPRTP RTP header extension for Interface Advertisements:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  RTP H-Ext ID |     length    |         MPR_Type=0x01         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                Interface #1 Advertisement block               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                Interface #2 Advertisement block               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               Interface #... Advertisement block              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                Interface #n Advertisement block               |
+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
|                         RTP Payload                           |
|                             ....                              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 7: Media Sender’s Interface Advertisement (RTP header extension)

Interface Advertisement block: variable size

Defined later in the section.

8.5.2. Interface Address Advertisement block

This block describes a method to represent IPv4, IPv6 and generic DNS-type addresses in a block format. It is based on the sub-reporting block in RFC 5760 [5].

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Type={0,1,2}  |     Length    |           RTP Port            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           RTCP Port           |            Address            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+                               +
:                                                               :
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 8: Interface Address Advertisement block during path discovery

Type: 8 bits

The Type corresponds to the type of address. Namely:

0: IPv4 address

1: IPv6 address

2: DNS name

Length: 8 bits

The length of the Interface Advertisement block in bytes.

For an IPv4 address, this should be 9 (i.e., 5 octets for the header and 4 octets for IPv4 address).

For an IPv6 address, this should be 21.

For a DNS name, the length field indicates the number of octets making up the string plus the 5 byte header.

RTP Port: 2 octets

The port number to which the sender sends RTP data. A port number of 0 is invalid and MUST NOT be used.

RTCP Port: 2 octets

The port number to which receivers send feedback reports. A port number of 0 is invalid and MUST NOT be used.

Address: 4 octets (IPv4), 16 octets (IPv6), or n octets (DNS name)

The address to which receivers send feedback reports. For IPv4 and IPv6, fixed-length address fields are used. A DNS name is an arbitrary-length string. The string MAY contain Internationalizing Domain Names in Applications (IDNA) domain names and MUST be UTF-8 encoded [6].

8.5.3. MPRTP RTCP Header Extension

The MPRTP RTCP header extension is used 1) to provide RTCP feedback per subflow to gauge the characteristics of each path, 2) to advertise the multiple interface addresses for a media receiver, and 3) perform connectivity check on the new interfaces.

MPRTP RTCP header extension for flow specific SR/RR: TBD

MPRTP RTCP header extension for Interface advertisement:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|V=2|P|    RC   | PT=MP_RR=2xx  |             length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     SSRC of packet sender                     |
+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
|                 SSRC_1 (SSRC of first source)                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           MPRR_Type           |     length    |      RESV     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                Interface #1 Advertisement block               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                Interface #2 Advertisement block               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|               Interface #... Advertisement block              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                Interface #m Advertisement block               |
+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
```

Figure 9: Media Receiver’s Interface Advertisement. (RTCP header Extension)

MP_RR: 8 bits

Indicates that it is a RTCP Receiver Report extension for MPRTP.

MPRR_Type: 16-bits

The MPRR_Type field corresponds to the type of MPRTP RTCP packet. Namely:

0x00: Subflow RTCP Statistics Aggregation

0x01: Interface Advertisement

0x02: Connectivity Check

length: 8-bits

The 8-bit length field is the length of extension data in bytes not including the MPRR_Type and length fields. The value zero indicates there is no data following.

Interface Advertisement block: variable size

Already defined in Section 8.5.2.

9. SDP Considerations

The packet formats specified in this document define extensions for RTP and RTCP. The use of MPRTP is left to the discretion of the sender and receiver.

A participant of a media session MAY use SDP to signal that it supports MPRTP. Not providing this information may/will make the sender or receiver ignore the header extensions. However, MPRTP MAY be used by either sender or receiver without prior signaling.

```
mprtp-attrib = "a=" "mprtp" [":"
        mprtp-optional-parameter]
        CRLF   ; flag to enable MPRTP
```

The literal 'mprtp' MUST be used to indicate support for MPRTP. Generally, senders and receivers SHOULD indicate this capability if they support MPRTP and would like to use it in the specific media session being signaled. However, it is possible for an MPRTP sender to stream data using multiple paths to a non-MPRTP client.

Currently, there are no extensions defined for the literal 'mprtp' but we provide the opportunity to extend it using the mprtp-optional-parameter.

9.1. Increased Throughput

The MPRTP layer MAY choose to augment paths to increase throughput. If the desired media rate exceeds the current media rate, the peers MUST renegotiate the application specific ("b=AS:") [14] bandwidth.

10. Acknowledgements

Varun Singh, Saba Ahsan, and Teemu Karkkainen are supported by Trilogy (http://www.trilogy-project.org), a research project (ICT-216372) partially funded by the European Community under its Seventh Framework Program. The views expressed here are those of the author(s) only. The European Commission is not liable for any use that may be made of the information in this document.

11. IANA Considerations

TBD.

12. Security Considerations

All drafts are required to have a security considerations section. See RFC 3552 [15] for a guide.

13. References

13.1. Normative References

[1] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, July 2003.

[2] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[3] Rosenberg, J., "Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal for Offer/Answer Protocols", RFC 5245, April 2010.

[4] Singer, D. and H. Desineni, "A General Mechanism for RTP Header Extensions", RFC 5285, July 2008.

[5] Ott, J., Chesterfield, J., and E. Schooler, "RTP Control Protocol (RTCP) Extensions for Single-Source Multicast Sessions with Unicast Feedback", RFC 5760, February 2010.

[6] Yergeau, F., "UTF-8, a transformation format of ISO 10646", STD 63, RFC 3629, November 2003.

13.2. Informative References

[7] Stewart, R., "Stream Control Transmission Protocol", RFC 4960, September 2007.

[8] Ford, A., Raiciu, C., Handley, M., and J. Iyengar, "Architectural Guidelines for Multipath TCP Development", draft-ietf-mptcp-architecture-02 (work in progress), October 2010.

[9] Nordmark, E. and M. Bagnulo, "Shim6: Level 3 Multihoming Shim Protocol for IPv6", RFC 5533, June 2009.

[10] Westerlund, M. and S. Wenger, "RTP Topologies", RFC 5117, January 2008.

[11] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, June 2002.

[12] Schulzrinne, H., Rao, A., Lanphier, R., Westerlund, M., and M. Stiemerling, "Real Time Streaming Protocol 2.0 (RTSP)", draft-ietf-mmusic-rfc2326bis-25 (work in progress), October 2010.

[13] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model with Session Description Protocol (SDP)", RFC 3264, June 2002.

[14] Handley, M., Jacobson, V., and C. Perkins, "SDP: Session Description Protocol", RFC 4566, July 2006.

[15] Rescorla, E. and B. Korver, "Guidelines for Writing RFC Text on Security Considerations", BCP 72, RFC 3552, July 2003.

Authors' Addresses

Varun Singh
Aalto University
School of Science and Technology
Otakaari 5 A
Espoo, FIN 02150
Finland

Email: varun@comnet.tkk.fi

Teemu Karkkainen
Aalto University
School of Science and Technology
Otakaari 5 A
Espoo, FIN 02150
Finland

Email: teemuk@comnet.tkk.fi

Joerg Ott
Aalto University
School of Science and Technology
Otakaari 5 A
Espoo, FIN 02150
Finland

Email: jo@comnet.tkk.fi

Saba Ahsan
Aalto University
School of Science and Technology
Otakaari 5 A
Espoo, FIN 02150
Finland

Email: sahsan@cc.hut.fi

Lars Eggert
Nokia Research Center
P.O. Box 407
Nokia Group 00045
Finland

Phone: +358 50 48 24461
Email: lars.eggert@nokia.com
URI: http://research.nokia.com/people/lars_eggert