**Source: Huawei Technologies Co., Ltd.**

**Title: Multiparty RTT architecture and call flow for IMS data channel solution**

## Document for: Discussion and Agreement

## Agenda Item: 10.9

# 1 Introduction

The MP\_RTT work item has been approved at the SA plenary #98-e in document SP-221346, and the use cases and requirements were agreed and incorporated in the PD v0.1.1 at 3GPP SA4-e (AH) RTC SWG post 122 in document S4aR230061.

This contribution proposes an architecture and a call flow over IMS data channel for the MP\_RTT.

# 2 Multi-party RTT Solutions

## 2.2 Multi-party RTT over IMS Data Channel Solution

### 2.2.1 Architecture



The Multi-party RTT over data channel solution is based on data channel architecture, which is defined in clause AC.2.1 of TS 23.228.

According to clause 5.5 of RFC8865[2], for multiparty considerations, two alternatives were considered when searching for an efficient and easily implemented multiparty method for real-time text:

Multiple DC streams, one per participant:

One DC stream per source would be sent in the same session. UE can identify the source by the “label” attribute in the DC stream ID line when receiving RTT. If a new UE is added to the conference, a new downlink stream ID indicating the new UE should be added to all the existing participants. The conference application needs to manage the mapping relationship between the UE identity and the steam ID of each participant, obtain the corresponding UE identity according to the stream ID when receiving the real-time text of each participant.

Pros:

This is a straightforward solution. The load per source is low.

Cons:

With a high number of participants, the overhead of establishing and maintaining the high number of data channels required may be high, even if the load per channel is low.

Single DC stream, each participate use only one DC stream:

Only one DC stream for each participate, no SIP negotiation procedure for each participant when a new UE is added to the conference. The conference server should add a source information in front of the RTT content by identifying the label attribute in the DC stream ID line when receiving RTT from a UE.

Pros:

No negotiation when a new UE is added to the conference.

Cons:

The conference server should add decode and re-encode the RTT content.

### 2.2.2 Call Flow

#### 2.2.2.1 Multi DC Streams

An example for three participants in a conference:



Each UE has one uplink stream ID and two downlink stream IDs, if a new UE is added to the conference, a new downlink stream ID indicating the new UE should be added to all the existing participants.

The conference application needs to manage the mapping relationship between the UE name and the steam ID of each participant, obtain the corresponding UE name according to the stream ID when receiving the real-time text of each participant, and add the UE name before the real-time text to correctly display the source.

 

The steps are shown as below:

Case 1: UE-A create a conference and join UE-B and UE-C into the conference, then run the RTT application.

1. UE-A, UE-B and UE-C enter an audio/video conference and download the RTT application on each participant.

2. The UE-A runs application and establish application data channel, this REINVITE message carries SDP including 3 DC stream IDs, one ‘sendonly’ for UE-A sending RTT to other participants, one ‘recvonly’ for receiving UE-B’s RTT, and the last one ‘recvonly’ for receiving UE-C’s RTT, the label attribute in each ‘a=dcmap’ can be get from the conference information, which can identify each DC stream belongs to whom. The SDP offer example is shown as below:

m=application 911 UDP/DTLS/SCTP webrtc-datachannel

c=IN IP6 2001:db8::3

a=max-message-size:1000

a=sctp-port 5000

a=setup:actpass

a=dcmap:200 label="A-Identity";subprotocol="t140"

a=dcsa:200 fmtp:t140 cps=20 sendonly

a=dcsa:200 hlang-send:es eo

a=dcmap:201 label="B-Identity";subprotocol="t140"

a=dcsa:201 fmtp:t140 cps=20 recvonly

a=dcsa:201 hlang-recv:es eo

a=dcmap:202 label="C-Identity";subprotocol="t140"

a=dcsa:202 fmtp:t140 cps=20 recvonly

a=dcsa:202 hlang-recv:es eo

3. DCSF establish corresponding DC stream IDs for UE-A.

4-5. IMS-A sends REINVITE message with three stream IDs to UE-B and establish corresponding DC stream IDs for UE-B. The stream IDs are similar to step2.

6-7. IMS-A sends REINVITE message with three stream IDs to UE-C and establish corresponding DC stream IDs for UE-C. The stream IDs are similar to step2.

Case 2: UE-D call into the conference and run the RTT application.

8. UE-D calls into the conference created by UE-A, and runs the RTT application.

9. UE-D runs application and establish application data channel, this REINVITE message carries SDP including one DC stream ID for UE-D sending RTT to other participants.

10. IMS-A establishes the DC stream for UE-D.

11. The IMS-A identifies that there are three participants in the conference, so IMS-A decides to add a new downlink DC stream for each participant, and finally add three downlink streams for UE-D.

12-14. The IMS-A adds a new downlink DC stream for UE-A/UE-B/UE-C simultaneously, for receiving UE-D’s RTT.

15. The IMS-A adds three downlink DC streams for UE-D, for receiving UE-A/UE-B/UE-C’s RTT.

When UE-A sends RTT over the uplink stream ID, DCMF/MRF will simultaneously send the RTT to UE-B, UE-C and UE-D through the dedicated stream ID channel, UE-B, UE-C and UE-D can identify the source by the corresponding “label” attribute that included in the ‘a=dcmap’ line.

#### 2.2.2.2 Single DC Stream

An example for three participants in a conference:



T140 protocol is too old to be extended to support adding the source label, so the conference server can add a source label getting from the “label” attribute of ‘a=dcmap’ line in front of the text content when receiving the real-time text from a UE, and the terminal can display it directly without modification.

 

The steps are shown as below:

Case 1: UE-A create a conference and join UE-B and UE-C into the conference, then run the RTT application.

1. UE-A, UE-B and UE-C enter an audio/video conference and download the RTT application on each participant.

2. The UE-A runs application and establish application data channel, this REINVITE message carries SDP including only one DC stream ID, the SDP offer example:

m=application 911 UDP/DTLS/SCTP webrtc-datachannel

c=IN IP6 2001:db8::3

a=max-message-size:1000

a=sctp-port 5000

a=setup:actpass

a=dcmap:200 label="A-Identity";subprotocol="t140"

a=dcsa:200 fmtp:t140 cps=20 sendrecv

a=dcsa:200 hlang-send:es eo

3. DCSF establish corresponding DC stream ID for UE-A.

4-5. IMS-A sends REINVITE message with only one stream IDs to UE-B and establish corresponding DC stream ID for UE-B. The stream ID is similar to step2.

6-7. IMS-A sends REINVITE message with only one stream ID to UE-C and establish corresponding DC stream ID for UE-C. The stream ID is similar to step2.

Case 2: UE-D call into the conference and run the RTT application.

8. UE-D calls into the conference created by UE-A, and runs the RTT application.

9. UE-D runs application and establish application data channel, this REINVITE message carries SDP including one DC stream ID for UE-D sending and receiving RTT.

10. IMS-A establishes the DC stream for UE-D.

When UE-A sends RTT over the uplink stream ID, DCMF/MRF will identify the source by the application data channel established is between UE-A and DCMF/MRF, and then add the UE-A’s identity as source to the RTT content. DCMF simultaneously send the RTT to UE-B, UE-C and UE-D through the dedicated stream ID channel, UE-B, UE-C and UE-D directly display the RTT content.

# 3 Proposal

We propose to agree to incorporate the architecture and the call flow into the MP\_RTT PD.

# References

1. TS 23.228: “IP Multimedia Subsystem (IMS)”
2. RFC8865: “T.140 Real-Time Text Conversation over WebRTC Data Channels”