Network Working Group
Internet-Draft
Intended status: Informational
Expires: September 6, 2012

H. Scholz
VOIPFUTURE GmbH

March 5, 2012

RTP Stream Information Export using IPFIX
draft-scholz-ipfix-rtp-msg-00

Abstract

This draft defines a set of Information Elements and matching Templates to convey RTP media stream information in IPFIX packets. The Information Elements describe the RTP header and payload characteristics of the RTP stream either for the entire duration of the monitored stream or for a smaller time slice.

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 September 6, 2012.

Table of Contents

1. Introduction

IPFIX [RFC5101] and [RFC5102] defines a framework allowing to export arbitrary data from so called IPFIX exporters. One type of IPFIX exporter may be co-located or passively observe Session Initiation Protocol (SIP) [RFC3261] based VoIP calls. The signaling messages can be exported using [I-D.trammell-ipfix-sip-msg] which leaves the Real Time Protocol (RTP) [RFC3550] media streams unmonitored. This document defines a set of additional IPFIX Information Elements (IEs) to describe RTP streams on various levels. These layers include the IP transport, RTP header information as well as Quality of Service (QoS) information of monitored streams.

1.1. Use Cases

RTP stream flow information contained in IPFIX flow records can be used for various tasks such as Quality of Service (QoS) monitoring, Service Level Agreement (SLA) validation and general troubleshooting of VoIP networks.

1.1.1. Quality of Service (QoS) Monitoring

Aggregated to higher-level metrics the in-depth information provided by the RTP (and optionally SIP) flow records allows service providers to gauge the overall quality of their network in terms of the quality of experience (QoE). On this level an individual call is less important but the overall quality (e.g. amount of minutes meeting certain quality standards) can be used to get a quick overview on the network and service performance.

1.1.2. Service Level Agreement (SLA)

SLAs are typically used as part of contracts between two network operators. The requirements on the reliability of the data may be higher compared to QoS Monitoring as the failure to meet contractually agreed quality standards often has a direct commercial impact.

1.1.3. Troubleshooting

An active network component (SIP proxy, B2BUA, media server) may not have the capabilities to store session related information for a long time to facilitate troubleshooting capabilities (e.g. due to missing harddisk). Such a system or a group of systems may run the metering process and export the data to a collector for processing or troubleshooting purposes.

2. Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

3. Base RTP Information Elements

The base RTP Information Elements cover information transported outside the Real Time Protocol [RFC3550] or defined by the Metering Process.

3.1. rtpObservationType

Description: The rtpObservationType is similar to the sipObservationType from [I-D.trammell-ipfix-sip-msg].

0: unknown: The Metering Process does not specify the observation type

1: receiver: The Metering Process is, or is co-located with, the receiver of the RTP stream.

2: sender: The Metering Process is, or is co-located with, the sender of the RTP stream.

3: passive: The Metering Process passively observed the RTP stream.

4: rtcp: TBD

Data Type: unsigned8

Data Type Semantics: identifier

PEN (provisional): xxx

ElementId (provisional): xxx

3.2. rtpFlowId

The rtpFlowId field is used to identify the RTP stream covered in this flow record. It shall be used to correlate the RTP stream to IPFIX SIP sessions. The rtpFlowId is calculated using a 5-tuple of source IP, source port, destination IP, destination port and transport protocol. Additionally the RTP SSRC is added. The exact calculation method is up for discussion. VOIPFUTURE uses a 64Bit

hash value.

Description: Hash value identifying the observed RTP stream.

Data Type: unsigned64

Data Type Semantics: identifier

PEN (provisional): xxx

ElementId (provisional): xxx

3.3. rtpStartTime

Description: Start time of the RTP stream in milliseconds since 0000 UTC Jan 1, 1970. The time is taken from the local clock and not from the RTP stream timestamp field. The local clock SHALL be NTP synchronized. If this flow record covers only part of an RTP stream the start time must be set to the start time of the observation time/interval.

Data Type: dateTimeMilliseconds

Data Type Semantics: identifier

PEN (provisional): xxx

ElementId (provisional): xxx

3.4. rtpEndTime

Description: End time of the RTP stream in milliseconds since 0000 UTC Jan 1, 1970. The time is taken from the local clock and not from the RTP stream timestamp field. The local clock SHALL be NTP synchronized. If this flow record covers only part of an RTP stream the end time must be set to the end time of the observation time/interval.

Data Type: dateTimeMilliseconds

Data Type Semantics: identifier

PEN (provisional): xxx

ElementId (provisional): xxx

3.5. rtpSampleOffset

Description: Offset of the observation interval contained in this flow record. The value is measured in milliseconds and contains the offset of the beginning of the flow record from the beginning of the RTP stream.

Data Type: unsigned32

Data Type Semantics: identifier

PEN (provisional): xxx

ElementId (provisional): xxx

3.6. rtpSampleTime

An IPFIX observer may generate and export a flow record for the entire duration of an RTP stream or for a specific part, e.g. a fixed time slice of 10 seconds. In case a single flow record is created the rtpSampleTime equals the RTP stream duration in milliseconds. In either case the rtpStreamState IE should be set to true if this flow record describes an ended RTP stream.

Description: Duration of the observation time of this flow record measured in milliseconds.

Data Type: unsigned32

Data Type Semantics: deltaCounter

PEN (provisional): xxx

ElementId (provisional): xxx

3.7. rtpStreamState

Using the rtpSampleOffset and rtpSampleTime IEs flow entries may be generated which describe only part of an RTP stream. This IE is used to describe the state of the observed stream, e.g. to indicate the reception of the last flow record belonging to a single RTP stream.

Description:

0: undefined: The state of the stream is not known.

1: running: The Metering Process expects more RTP packets or has already received packets for this RTP stream which are outside the scope of this flow record.

2: ended: The Metering Process determined that the RTP stream ended. Information sources could be signaling information or the fact that no RTP media has been received for a longer period of time.

3: no packets: The Metering Process has not received any RTP packets for this RTP stream in the observation interval but the stream has not ended. A VoIP endpoint may have requested the media stream to be suspended, i.e. put 'on hold' (tbd:reference to sendonly ..)

Data Type: unsigned8

Data Type Semantics: identifier

PEN (provisional): xxx

ElementId (provisional): xxx

3.8. rtpProtocolVersion

Description: Value of the RTP version taken from the RTP header. For RFC 3550 [RFC3550] RTP packets this will typically be set to 2.

Data Type: unsigned8

Data Type Semantics: identifier

PEN (provisional): xxx

ElementId (provisional): xxx

3.9. rtpPayloadType

Description: Payload Type (PT) as conveyed in RTP header

Data Type: unsigned8

Data Type Semantics: identifier

PEN (provisional): xxx

ElementId (provisional): xxx

3.10. rtpMediaType

Every RTP stream falls into a certain category, e.g. audio or video. These RTP media types have been defined in [RFC4566] and are carried in this Information Element. A Metering Process may learn these based on the information in the RTP stream alone or (e.g. when co-located with a SIP Metering Process) based on signaling information. New media types may be defined in the registry created by [RFC3840].

Description:

0: unknown: unknown media type

1: audio: audio RTP stream

2: video: video RTP stream

3: text: text session

4: application: tbd

5: message: tbd

Data Type: unsigned8

Data Type Semantics: identifier

PEN (provisional): xxx

ElementId (provisional): xxx

3.11. rtpMediaSubType

The string value for this IE can be found in the IANA registry for 'RTP Payload Format MIME types' on http://www.iana.org/assignments/rtp-parameters

Description: Media subtype based on IANA registry

Data Type: string

Data Type Semantics: identifier

PEN (provisional): xxx

ElementId (provisional): xxx

3.12. rtpIsSRTP

tbd: The tri-state rtpIsSRTP will be set if SRTP is used/not used or the content is not known. Might require the RTP Metering Process to be co-located with a signaling process.

3.13. rtpSSRC

Description: SSRC field as conveyed in RTP header. The SSRC field identifies the synchronization source.

Data Type: unsigned32

Data Type Semantics: identifier

PEN (provisional): xxx

ElementId (provisional): xxx

3.14. rtpCSRC

Description: Convey the CSRCs as RFC6313 basicList? To be discussed

Data Type: basicList

Data Type Semantics: identifier

PEN (provisional): xxx

ElementId (provisional): xxx

3.15. rtpTimestamp

Description: Timestamp value as conveyed in RTP header. The timestamp is taken from the first RTP packet if multiple RTP packets are covered in this flow record.

Data Type: unsigned32

Data Type Semantics: identifier

PEN (provisional): xxx

ElementId (provisional): xxx

4. RTP Payload Information Elements

This section defines additional Information Elements which describe RTP streams based on their IP transport and RTP header parameters. Complicated metrics may be subject to different measurement methods. In order to prevent data from being unusable due to incompatible formats or measurement methods most Information Elements are counter values which allow calculation of metrics on mediator or collector systems. Additionally this allows matching flow records to be aggregatd by addition, e.g. addition of the rtpPacketCount values of multiple observation intervals.

4.1. rtpPacketCount

Description: Number of RTP packets covered in this flow record. This includes observed duplicate packets.

Data Type: unsigned32

Data Type Semantics: deltaCounter

PEN (provisional): xxx

ElementId (provisional): xxx

4.2. rtpPacketCountLoss

Description: Number of RTP packets lost in the duration covered by this flow record. The number of lost packets SHOULD be calculated using the RTP sequence numbers.

Data Type: unsigned32

Data Type Semantics: deltaCounter

PEN (provisional): xxx

ElementId (provisional): xxx

4.3. rtpPacketCountDiscarded

Passive monitoring equipment shall assume a fixed 40 millisecond jitter buffer (TODO: add reference to TM Forum/ITU). A packet observed later than the expected packet inter-arrival time plus the 40ms is assumed to be received by the RTP receiver too late to be played out. Even though the packet may be received by the RTP receiver it will be discarded which has the same effect as packet loss.

Description: Number of RTP packets discarded in the duration covered by this flow record.

Data Type: unsigned32

Data Type Semantics: deltaCounter

PEN (provisional): xxx

ElementId (provisional): xxx

4.4. rtpCodecChange

Description: The codec used in the observed RTP stream may change throughout an observation interval. This value indicates the number of changes identified by changing RTP header Payload Type (PT) values.

Data Type: unsigned32

Data Type Semantics: deltaCounter

PEN (provisional): xxx

ElementId (provisional): xxx

4.5. rtpMarkerBit

Description: The RTP Marker Bit may be set on one or more packets within the observation interval. This value indicates the number of unique packets which had the Marker Bit set. Duplicate packets MUST be ignored for this calculation.

Data Type: unsigned32

Data Type Semantics: deltaCounter

PEN (provisional): xxx

ElementId (provisional): xxx

4.6. rtpComfortNoise

Description: RTP packets of the observed RTP stream may be Comfort Noise packets. This value indicates the number of unique Comfort Noise packets in this observation interval. Duplicate packets MUST be ignored for this calculation.

Data Type: unsigned32

Data Type Semantics: deltaCounter

PEN (provisional): xxx

ElementId (provisional): xxx

4.7. rtpDTMFTones

tbd: # of DTMF tones observed in interval, e.g. PT 101. May also include in-band if possible by observation method.

4.8. rtpPacketization

Description: The packetization time of the data contained in the RTP packets in milliseconds. If a packetization time change occured during the monitoring interval described by this flow record the rtpPacketization value SHALL be set to the interval used for most of the packets. If no packetization time can be determined the value MUST be set to 0.

Data Type: unsigned8

Data Type Semantics: quantity

PEN (provisional): xxx

ElementId (provisional): xxx

4.9. rtpPacketizationChange

Description: The packetization time of RTP packets may change throughout a single RTP stream. Endpoints may use the ptime SDP parameter to indicate/request changes or change the packetization time without advance notice if allowed by the codec. This value indicates the number of packetization time changes in this observation interval.

Data Type: unsigned8

Data Type Semantics: deltaCounter

PEN (provisional): xxx

ElementId (provisional): xxx

4.10. rtpDSCPChange

tbd: should be transport based - # of changes of DSCP/ToS bits on IP layer. Alternative: basicList of seen DSCP classes.

5. Quality of Service Information Elements

5.1. rtpSenderSynchronization

tbd: VOIPFUTURE specific

5.2. rtpSenderRestart

tbd: VOIPFUTURE specific, can be incorporated into rtpSequenceError

5.3. rtpSenderJitter

tbd: VOIPFUTURE specific

5.4. rtpNetworkOverload

tbd: VOIPFUTURE specific

5.5. rtpOverloadWithPacketLoss

tbd: VOIPFUTURE specific

5.6. rtpTolerableJitter

TM Forum and ITU recommend that monitoring equipment such as the IPFIX Metering Process should assume a jitter buffer with a fixed size of 40 milliseconds. Based on this value we consider jitter as

tolerable if the inter-arrival time between to packets does not exceed these 40 milliseconds. Based on a 20 millisecond packetization time (as conveyed in rtpPacketInterval) a jitter of less than 20 milliseconds can be considered as tolerable. TBD: define jitter measurement method, most likely something better than RFC3550.

Description: The number of RTP packets with a tolerable jitter as defined above. Duplicate packets MUST be ignored for this calculation.

Data Type: unsigned32

Data Type Semantics: deltaCounter

PEN (provisional): xxx

ElementId (provisional): xxx

5.7. rtpCriticalJitter

Based on the rtpTolerableJitter definition this Information Element describes the number of RTP packets which arrived with a critical value for the packet inter-arrival time, i.e. more than 40 milliseconds after the previous packet.

Description: The number of RTP packets with a critical jitter as defined above. Duplicate packets MUST be ignored for this calculation.

Data Type: unsigned32

Data Type Semantics: deltaCounter

PEN (provisional): xxx

ElementId (provisional): xxx

5.8. rtpVeryLargeJitter

RTP packets observed with a packet inter-arrival time of more than 80 milliseconds above the indicated packetization time (e.g. >100ms in case of a 20ms packetization time) are counted in this very large jitter category. Often RTP packets will match this property if network buffering occurred during the observation interval.

Description: The number of RTP packets with a very large jitter as defined above. Duplicate packets MUST be ignored for this calculation.

Data Type: unsigned32

Data Type Semantics: deltaCounter

PEN (provisional): xxx

ElementId (provisional): xxx

5.9. rtpTolerablePacketLoss

RTP packet loss can be detected using the RTP sequence number. A loss event is considered tolerable if only one packet is lost before the RTP stream continues.

Description: The number of tolerable packet loss events as defined above observed in the observation interval.

Data Type: unsigned32

Data Type Semantics: deltaCounter

PEN (provisional): xxx

ElementId (provisional): xxx

5.10. rtpCriticalPacketLoss

RTP packet loss can be detected using the RTP sequence number. A loss event is considered critical if more than one packet is lost before the RTP stream continues.

Description: The number of critical packet loss events as defined above observed in the observation interval.

Data Type: unsigned32

Data Type Semantics: deltaCounter

PEN (provisional): xxx

ElementId (provisional): xxx

5.11. rtpCriticalLossDensity

VOIPFUTURE needs to describe, should be kept in. We need to cover burst-loss

5.12. rtpOverloadWithPacketOrder

VOIPFUTURE needs to describe, optional

5.13. rtpDuplicates

Description: Duplicate RTP packets may be observed and identified based on the RTP sequence number. This counter indicates the number of observed duplicate packets in the observation interval.

Data Type: unsigned32

Data Type Semantics: deltaCounter

PEN (provisional): xxx

ElementId (provisional): xxx

5.14. rtpPacketOrder

RTP packets may be re-ordered during transmission, e.g. due to routing changes. This Information Element indicates the number of RTP packets which are received out of order during the observation interval.

Description: Number of out-of-order RTP packets observed. Duplicate packets must not count towards this value.

Data Type: unsigned32

Data Type Semantics: deltaCounter

PEN (provisional): xxx

ElementId (provisional): xxx

5.15. rtpSequenceError

The RTP header sequence numbers increases monotonically throughout an RTP session. A forward or backward jump in sequence numbers or a reset of the sequence number observed indicates problems on the sender side. Additionally the receiver of the RTP stream may suffer as handling these situations is not defined.

Description: Number of RTP sequence error events observed.
Duplicate packets must not count towards this value.

Data Type: unsigned32

Data Type Semantics: deltaCounter

PEN (provisional): xxx

ElementId (provisional): xxx

5.16. rtpLowPacketInterval

to be defined by VOIPFUTURE

5.17. rtpNoPacketInterval

to be defined by VOIPFUTURE

5.18. rtpBadRTPTimestamp

to be defined by VOIPFUTURE

5.19. rtpMinJitter

Description: The minimum jitter value is defined as the minimal packet inter-arrival time in milliseconds in the observation interval.

Data Type: unsigned8

Data Type Semantics: quantity

PEN (provisional): xxx

ElementId (provisional): xxx

5.20. Average Jitter

Calculating and transporting an average jitter value makes little sense if the data should be aggregated at a later stage, e.g. by a mediator, collector or an application processing the data. To remedy this both the number of packet inter-arrival times and the sum of the packet inter-arrival times in milliseconds are counted.

The average jitter value can be calculated by dividing rtpJitterSum by rtpJitterCount.

An average jitter value for multiple flow records can be calculated by dividing the sum of all rtpJitterSum values by the sum of all rtpJitterCount values.

5.20.1. rtpJitterCount

Description: The number of packet inter-arrival times observed.

Data Type: unsigned16

Data Type Semantics: deltaCounter

PEN (provisional): xxx

ElementId (provisional): xxx

5.20.2. rtpJitterSum

Description: The sum of all packet inter-arrival times in milliseconds.

Data Type: unsigned32

Data Type Semantics: quantity

PEN (provisional): xxx

ElementId (provisional): xxx

5.21. rtpMaxJitter

Description: The maximum jitter value is defined as the maximal packet inter-arrival time in milliseconds in the observation interval.

Data Type: unsigned8

Data Type Semantics: quantity

PEN (provisional): xxx

ElementId (provisional): xxx

5.22. Jitter histogram

In order to display and aggregate detailed jitter information of the observed RTP stream additional Information Elements are required. The Metering Process observing an RTP stream analyses the packet

inter-arrival times without needing to know the agreed packetiziation time. The following calculations are only made for packets with consecutive RTP sequence numbers. From 250 received RTP packets in an observation interval a Metering Process can calculate 249 packet inter-arrival times. In case of packet loss (i.e. missing sequence number in the packet stream) no packet inter-arrival time is calculated. Based on the calculated inter-arrival time the counters in the buckets listed below are increased. All counters start at zero per observation interval. Inter-arrival times have a millisecond granularity.

5.22.1. rtpJitterBucket0

Description: Number of packet inter-arrival times between 0 and lower than 2.5 milliseconds.

Data Type: unsigned16

Data Type Semantics: deltaCounter

PEN (provisional): xxx

ElementId (provisional): xxx

5.22.2. rtpJitterBucket5

Description: Number of packet inter-arrival times starting at 2.5 and lower than 7.5 milliseconds.

Data Type: unsigned16

Data Type Semantics: deltaCounter

PEN (provisional): xxx

ElementId (provisional): xxx

5.22.3. rtpJitterBucket10

Description: Number of packet inter-arrival times starting at 7.5 and lower than 12.5 milliseconds.

Data Type: unsigned16

Data Type Semantics: deltaCounter

PEN (provisional): xxx

ElementId (provisional): xxx

5.22.4. rtpJitterBucket15

Description: Number of packet inter-arrival times starting at 12.5 and lower than 17.5 milliseconds.

Data Type: unsigned16

Data Type Semantics: deltaCounter

PEN (provisional): xxx

ElementId (provisional): xxx

5.22.5. rtpJitterBucket20

Description: Number of packet inter-arrival times starting at 17.5 and lower than 22.5 milliseconds.

Data Type: unsigned16

Data Type Semantics: deltaCounter

PEN (provisional): xxx

ElementId (provisional): xxx

5.22.6. rtpJitterBucket25

Description: Number of packet inter-arrival times starting at 22.5 and lower than 27.5 milliseconds.

Data Type: unsigned16

Data Type Semantics: deltaCounter

PEN (provisional): xxx

ElementId (provisional): xxx

5.22.7. rtpJitterBucket30

Description: Number of packet inter-arrival times starting at 27.5 and lower than 32.5 milliseconds.

Data Type: unsigned16

Data Type Semantics: deltaCounter

PEN (provisional): xxx

ElementId (provisional): xxx

5.22.8. rtpJitterBucket35

Description: Number of packet inter-arrival times starting at 32.5 and lower than 37.5 milliseconds.

Data Type: unsigned16

Data Type Semantics: deltaCounter

PEN (provisional): xxx

ElementId (provisional): xxx

5.22.9. rtpJitterBucket40

Description: Number of packet inter-arrival times starting at 37.5 and lower than 37.5 milliseconds.

Data Type: unsigned16

Data Type Semantics: deltaCounter

PEN (provisional): xxx

ElementId (provisional): xxx

5.22.10. rtpJitterBucket45

Description: Number of packet inter-arrival times starting at 42.5 and lower than 47.5 milliseconds.

Data Type: unsigned16

Data Type Semantics: deltaCounter

PEN (provisional): xxx

ElementId (provisional): xxx

5.22.11. rtpJitterBucket50

Description: Number of packet inter-arrival times starting at 47.5 and lower than 52.5 milliseconds.

Data Type: unsigned16

Data Type Semantics: deltaCounter

PEN (provisional): xxx

ElementId (provisional): xxx

5.22.12. rtpJitterBucket55

Description: Number of packet inter-arrival times starting at 52.5 and lower than 57.5 milliseconds.

Data Type: unsigned16

Data Type Semantics: deltaCounter

PEN (provisional): xxx

ElementId (provisional): xxx

5.22.13. rtpJitterBucket60

Description: Number of packet inter-arrival times starting at 57.5 and lower than 62.5 milliseconds.

Data Type: unsigned16

Data Type Semantics: deltaCounter

PEN (provisional): xxx

ElementId (provisional): xxx

5.22.14. rtpJitterBucket65

Description: Number of packet inter-arrival times starting at 62.5 and lower than 67.5 milliseconds.

Data Type: unsigned16

Data Type Semantics: deltaCounter

PEN (provisional): xxx

ElementId (provisional): xxx

5.22.15. rtpJitterBucket70

Description: Number of packet inter-arrival times starting at 67.5 and lower than 72.5 milliseconds.

Data Type: unsigned16

Data Type Semantics: deltaCounter

PEN (provisional): xxx

ElementId (provisional): xxx

5.22.16. rtpJitterBucket75

Description: Number of packet inter-arrival times starting at 72.5 and lower than 77.5 milliseconds.

Data Type: unsigned16

Data Type Semantics: deltaCounter

PEN (provisional): xxx

ElementId (provisional): xxx

5.22.17. rtpJitterBucket80

Description: Number of packet inter-arrival times starting at 77.5 and lower than 82.5 milliseconds.

Data Type: unsigned16

Data Type Semantics: deltaCounter

PEN (provisional): xxx

ElementId (provisional): xxx

5.22.18. rtpJitterBucket85

Description: Number of packet inter-arrival times starting at 82.5 and lower than 87.5 milliseconds.

Data Type: unsigned16

Data Type Semantics: deltaCounter

PEN (provisional): xxx

ElementId (provisional): xxx

5.22.19. rtpJitterBucket90

Description: Number of packet inter-arrival times starting at 87.5 and lower than 92.5 milliseconds.

Data Type: unsigned16

Data Type Semantics: deltaCounter

PEN (provisional): xxx

ElementId (provisional): xxx

5.22.20. rtpJitterBucket95

Description: Number of packet inter-arrival times starting at 92.5 and lower than 97.5 milliseconds.

Data Type: unsigned16

Data Type Semantics: deltaCounter

PEN (provisional): xxx

ElementId (provisional): xxx

5.22.21. rtpJitterBucket100

Description: Number of packet inter-arrival times starting at 97.5 milliseconds.

Data Type: unsigned16

Data Type Semantics: deltaCounter

PEN (provisional): xxx

ElementId (provisional): xxx

5.23. rtpDelayType

The RTP one-way delay is the time RTP packets need to travel from the source (RTP sender) to the destination (RTP receiver). This value may be obtained from RTCP reports generated by RTP receivers or by actively using ICMP ping requests to obtain a rough approximation of the delay. ICMP based delay calculation is not encouraged. In any use case the observed or measured one-way delay is only a single data point which does not match the observation interval defined by the rtpSampleTime and rtpSampleOffset. TBD: discuss this, esp whether to move this in a separate section.

Description:

0: unknown: unknown delay measurement

1: rtcp: delay value taken from RTCP report

2: rtcp-xr: delay value taken from RTCP-XR report

3: ping: active ICMP ping

4: endpoint: mouth to ear delay

Data Type: unsigned8

Data Type Semantics: identifier

PEN (provisional): xxx

ElementId (provisional): xxx

5.24. rtpDelayOneWay

Description: One way RTP delay in milliseconds.

Data Type: unsigned16

Data Type Semantics: identifier

PEN (provisional): xxx

ElementId (provisional): xxx

6. MOS measurement

A multitude of Mean opinion score (MOS) assessment algorithms have been defined of which only one or few may be available to an IPFIX Metering Process. The quality (i.e. accuracy) of these algorithms varies and has to be noted when transporting MOS values.

An IPFIX Metering Process may use these Information Elements to convey information on the duration of the stream in which the quality fell into the respective category as well as the measurement algorithm used to obtain the information.

6.1. rtpMOSCAlg

The values carried in this IE are taken from the "RTCP XR QoE metric block - Calculation Algorithm" sub-registry of the "RTP Control Protocol Extended Reports (RTCP XR) Block Type Registry" as defined in [I-D.wu-xrblock-rtcp-xr-quality-monitoring].

Description: The calculation algorithm (CAlg) used by the Metering Process to calculate the MOS value.

0: undefined: The algorithm is not known/specified.

1: ITU-T P.564

2: G.107

3: G.107 / ETSI TS 101 329-5 Annex E

4: ITU-T P.NAMS

5: ITU-T P.NBAMS

6: RTCP - Real Time Control Protocol (not defined in registry!)

Data Type: unsigned8

Data Type Semantics: identifier

PEN (provisional): xxx

ElementId (provisional): xxx

The MOS values calculated are separated into MOS classes based on the ITU-T G.107 classes.

6.2. rtpMOSClass1

Description: Number of seconds the monitored stream had a MOS quality lower than 3.10

Data Type: float32

Data Type Semantics: deltaCounter

PEN (provisional): xxx

ElementId (provisional): xxx

6.3. rtpMOSClass2

Description: Number of seconds the monitored stream had a MOS quality larger than or equal 3.10 and lower than 3.60

Data Type: float32

Data Type Semantics: deltaCounter

PEN (provisional): xxx

ElementId (provisional): xxx

6.4. rtpMOSClass3

Description: Number of seconds the monitored stream had a MOS quality larger than or equal 3.60 and lower than 4.03

Data Type: float32

Data Type Semantics: deltaCounter

PEN (provisional): xxx

ElementId (provisional): xxx

6.5. rtpMOSClass4

Description: Number of seconds the monitored stream had a MOS quality larger than or equal 4.03 and lower than 4.34

Data Type: float32

Data Type Semantics: deltaCounter

PEN (provisional): xxx

ElementId (provisional): xxx

6.6. rtpMOSClass5

Description: Number of seconds the monitored stream had a MOS quality larger than or equal 4.34

Data Type: float32

Data Type Semantics: deltaCounter

PEN (provisional): xxx

ElementId (provisional): xxx

6.7. rtpMinMOS

Description: Minimum MOS value measured in the monitoring interval.

Data Type: float32

Data Type Semantics: quantity

PEN (provisional): xxx

ElementId (provisional): xxx

6.8. rtpAvgMOS

Description: Average MOS value measured in the monitoring interval.

Data Type: float32

Data Type Semantics: quantity

PEN (provisional): xxx

ElementId (provisional): xxx

6.9. rtpMaxMOS

Description: Maximum MOS value measured in the monitoring interval.

Data Type: float32

Data Type Semantics: quantity

PEN (provisional): xxx

ElementId (provisional): xxx

6.10. rtpMinRFactor

Description: Minimum R-Factor measured in the monitoring interval.

Data Type: float32

Data Type Semantics: quantity

PEN (provisional): xxx

ElementId (provisional): xxx

6.11. rtpAvgRFactor

Description: Average R-Factor measured in the monitoring interval.

Data Type: float32

Data Type Semantics: quantity

PEN (provisional): xxx

ElementId (provisional): xxx

6.12. rtpMaxRFactor

Description: Maximum R-Factor measured in the monitoring interval.

Data Type: float32

Data Type Semantics: quantity

PEN (provisional): xxx

ElementId (provisional): xxx

7. Recommended Templates

The defined RTP stream IPFIX templates must support both IPv4 and IPv6 transport. They need to carry either flow information regarding the entire duration of an RTP stream or specific to a shorter observation interval.

7.1. Entire stream

tbd

7.2. Time slice

tbd, based on previous template. Split a single RTP stream in three flow records as example including (empty) 'RTP stream ended' flow record.

8. Examples

9. Acknowledgements

tbd

10. IANA Considerations

tbd

11. Security Considerations

tbd

12. TODO

QUESTION-1: Should we try to take a biflow approach and join both stream directions of a call in a single flow record? I assume this would not give any big advantage and complicates scenarios in which Service Level Agreements are done on a per direction basis.

QUESTION-2: Should we define the CSRCs as RFC6313 basicList?

QUESTION-3: Should we degrade rtpPacketOrder or rtpSequenceError to a boolean?

QUESTION-4: Should rtpPacketCount include duplicate packets?

QUESTION-5: How about 'packet errors', e.g. packets which could not be processed properly?

QUESTION-6: Should we include a standard deviation value for rtpAvgJitter and other average values?

QUESTION-7: (from MK@VOIPFUTURE): It has to be possible to send an empty flow record indicating the end of an RTP stream.

QUESTION-8: should we reduce these deltaCounters to 16 Bit (now: 32Bit)? rtpCodecChange, rtpMarkerBit, rtpComfortNoise, rtpPacketIntervalChange, rtpSequenceError, rtp*Jitter

QUESTION-9: should we add direction indicators? E.g. caller-to-callee or callee-to-caller?

QUESTION-10: should we use milliseconds or seconds for the MOS class slots?

13. References

13.1. Normative References

[I-D.trammell-ipfix-sip-msg]
Claise, B., Trammell, B., Kaplan, H., and S. Niccolini, "SIP Message Information Export using IPFIX", draft-trammell-ipfix-sip-msg-02 (work in progress), October 2011.

[I-D.wu-xrblock-rtcp-xr-quality-monitoring]
Hunt, G., Clark, A., Wu, W., Schott, R., and G. Zorn, "RTCP XR Blocks for QoE metric reporting", draft-wu-xrblock-rtcp-xr-quality-monitoring-06 (work in progress), December 2011.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC5101] Claise, B., "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of IP Traffic Flow Information", RFC 5101, January 2008.

[RFC5102] Quittek, J., Bryant, S., Claise, B., Aitken, P., and J. Meyer, "Information Model for IP Flow Information Export", RFC 5102, January 2008.

13.2. Informative References

[RFC3261] 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.

[RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, July 2003.

[RFC3840] Rosenberg, J., Schulzrinne, H., and P. Kyzivat, "Indicating User Agent Capabilities in the Session Initiation Protocol (SIP)", RFC 3840, August 2004.

[RFC4566] Handley, M., Jacobson, V., and C. Perkins, "SDP: Session Description Protocol", RFC 4566, July 2006.

Author's Address

Hendrik Scholz
VOIPFUTURE GmbH
Wendenstrasse 4
Hamburg 20097
Germany

Phone: +49 40 688 900 100
Email: hscholz@voipfuture.com
URI: http://www.voipfuture.com/