

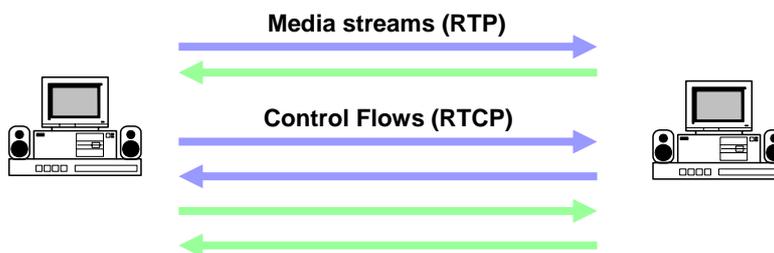


Real-Time Transport Protocol (RTP)



Real-time Transport Protocol (1)

- ▶ RTP Functionality (RFC 3550)
 - framing for audio/video information streams
 - preserve intra- and inter-stream timing
 - mechanisms for awareness of others in a conference
- ➔ RTP sessions



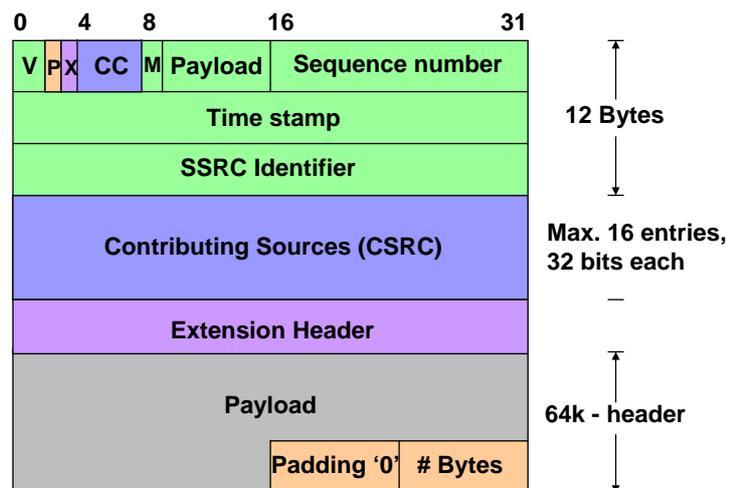


Real-time Transport Protocol (2)

- ▶ Standard RTP packet header
 - Independent of *payload type*
 - Possibly seconded by *payload header*
- ▶ Mechanisms
 - Detect packet loss, cope with reordering
 - sequence number per media stream
 - Determine variations in transmission delays
 - media specific time stamp (e.g., 8 kHz for PCM audio)
 - allows receiver to adapt playout point for continuous replay
 - Source identification
 - possibly mixed from several sources
 - Payload type identifier



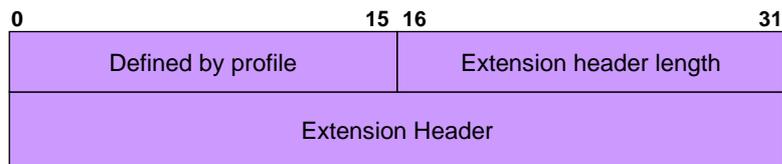
RTP Header





RTP Header Fields (1)

- V: Version — version 2 defined in RFC 1889
- P: Padding — indicates padding
bytes indicated in last byte
- X: eXtension bit — extension header is present
- Extension header — single additional header (TLV coded)



- CC: CSRC count — # of contributing sources
- CSRC: contributing sources — which sources have been “mixed”
to produce this packet’s contents



RTP Header Fields (2)

- M: Marker bit — marks semantical boundaries in
media stream (e.g. talk spurt)
- Payload type — indicates packet content type
- Sequence # — of the packet in the media stream
(strictly monotonically increasing)
- Timestamp — indicates the instant when the
packet contents was sampled
(measured to media-specific clock)
- SSRC: synchronization source —
identification of packet originator



Real-time Transport Control Protocol

Mechanisms:

- ▶ Receivers constantly measure transmission quality
 - delay, jitter, packet loss
- ▶ Regular control information exchange between senders and receivers
 - feedback to sender (receiver report)
 - feed forward to recipients (sender report)
- ▶ Allows applications to adapt to current QoS
- ▶ Overhead limited to a small fraction (default: 5% max.) of total bandwidth per RTP session
 - members estimate number of participants
 - adapt their own transmission rate

Obtaining sufficient capacity: outside of RTP

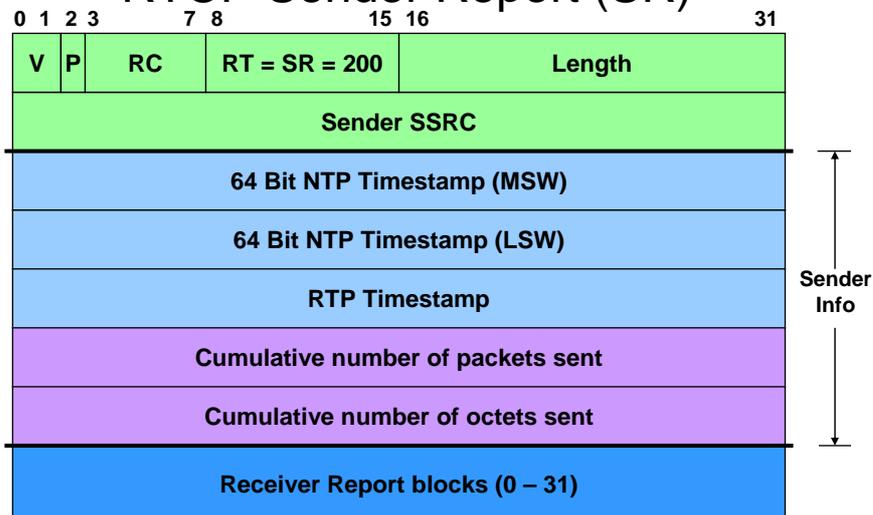


RTCP Sender Report

- ▶ Enable cross-media stream synchronization
 - Relate stream-specific RTP time stamp to wall clock time
 - NTP timestamp + RTP timestamp
 - Playout adjustment to be performed by the receivers
- ▶ Provide data point for RTT measurement
 - NTP timestamp
- ▶ Provide feed forward about data transmitted
 - Transmit sender's packet and byte count
 - Enable receiver to do proper loss calculation
- ▶ Include Receiver Reports for the sender as well



RTCP Sender Report (SR)

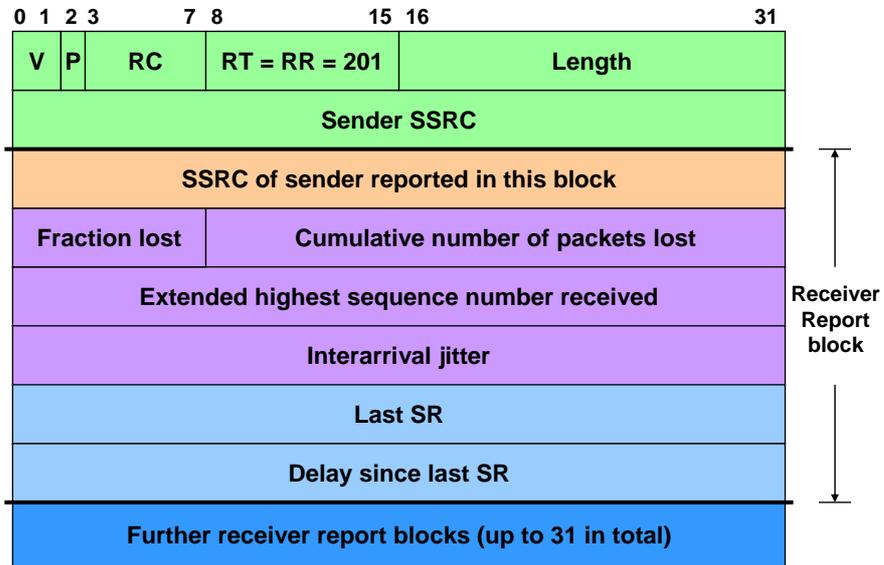


RTCP Receiver Report

- ▶ Feedback timing for RTT estimation
 - SR Timestamp
 - Middle 32 bits taken from the last SR's NTP timestamp
 - Delay since last SR
 - Local delay at receiver between receiver SR and sending the RR block
 - Measured in units of $1 / 65536$ seconds
- ▶ Provide per-sender reception statistics
 - Total number of packets lost
 - Fraction of packets lost (in units of $1 / 256$)
 - Highest sequence number received so far
 - Jitter of received packets
- ▶ Enable adaptive sender behavior
 - Adjust codecs, codec parameters, transmission rate, etc.



RTCP Receiver Report (RR)



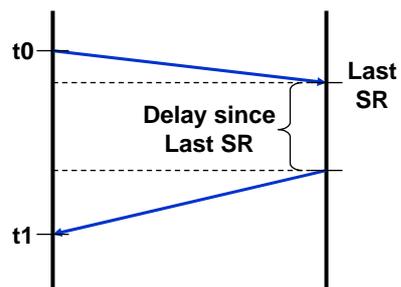
RTCP Statistics Collection (Sender)

▶ Round-Trip Time (sender only)

- Derived from time stamps in RR
- Simple formula:

$$RTT = t1 - t0 - DSL_SR$$

- RTT may be asymmetric!



▶ Byte count

▶ Packet count



RTCP Statistics Collection (Receiver)

▶ Packet Loss

- Calculated from gaps in sequence number space
 - First (lowest packet sequence number) received
- Expected number of packets = current – lowest
- Received number of packets
 - Count duplicates, out-of-order, and late packets as received!
- Absolute # of lost packets = expected – received
 - May be negative!
- Fraction of lost packet
 - Loss since last SR or RR packet was sent
- Loss of all packets not detected!

▶ Extended highest sequence number received (32 bits)

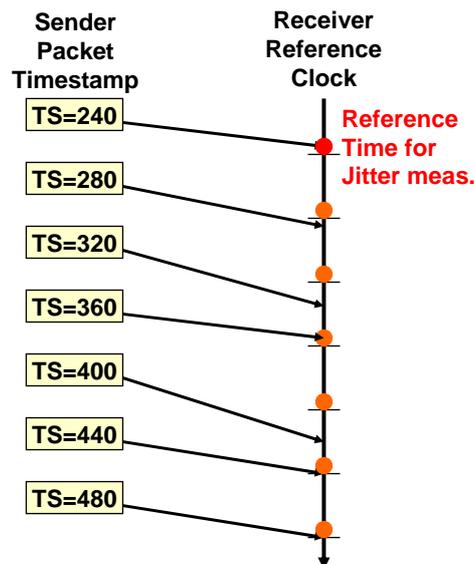
▶ Time of last SR reception

▶ Jitter



RTCP Interarrival Jitter Estimation

- ▶ Receiver measures in time units of the media clock
- ▶ Relates it to local real-time clock
- ▶ Initialized through first packet received
- ▶ Derives expected reception time
- ▶ Calculates deviation D upon packet reception
- ▶ Sampled for each packet
- ▶ Jitter derived for each peer of successively received packets
 - Ordering is not relevant
- ▶ Weighing function:
$$J = J' + (D - J') / 16$$





RTCP Source Description (SDES)

▶ Persistent Identification of an endpoint: Canonical Name

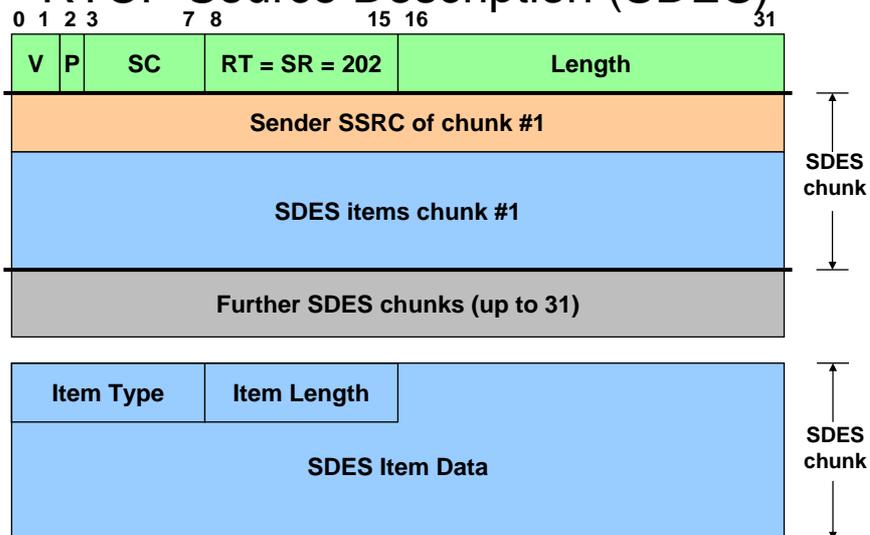
- CNAME — globally unique identifier (id@host)
- *Mandatory!*
- Binding across RTP sessions
- Identification across changes in the SSRC in an RTP session

▶ Providing additional information about an endpoint

- NAME — Name of user (or system)
- EMAIL — mailto: address
- PHONE — phone number
- LOC — location (no format defined)
- TOOL — (software) client in use
- NOTE — brief note to other participants (e.g. "on the phone")
- PRIV — private extensions



RTCP Source Description (SDES)

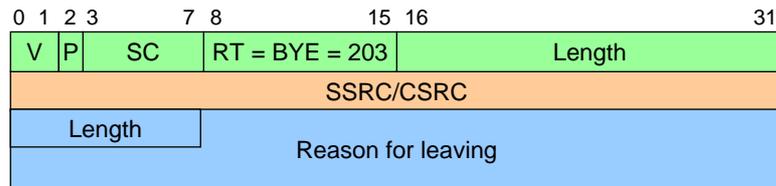




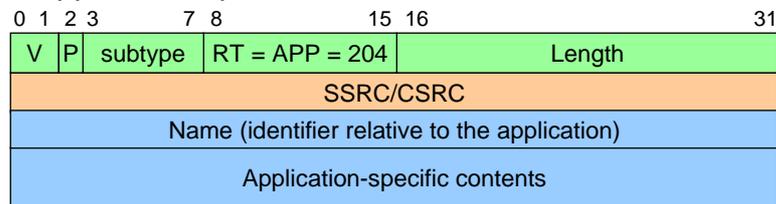
Other RTCP Packets

▶ **BYE:** Announce that an entity will be leaving a session

- Optional: provide a reason phrase



▶ **APP:** Application-specific extensions



Extended RTCP Reporting (XR)

▶ Provide more detailed feedback (and feed forward)

- Infer network characteristics (point-to-point and multicast)
- Provide detailed voice quality information

▶ Incorporate many statistics in RTCP packets

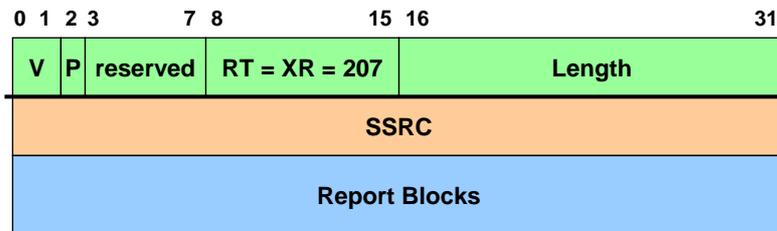
- Lost and duplicate packets
- Exact packet receipt times
- Receiver reference time and reception information
 - for RTT measurements
- Statistics summary
- VoIP metrics: Burst, gaps, delay, ...

▶ Detailed reports may get large: thinning reports

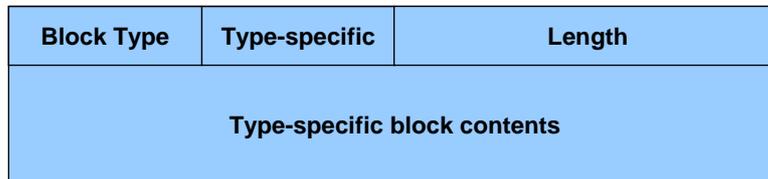
- Report only on every 2^T -th packet ($T = 0, \dots, 15$)
- Indicate the thinning factor T in the packet

RTCP XR

▶ General report header



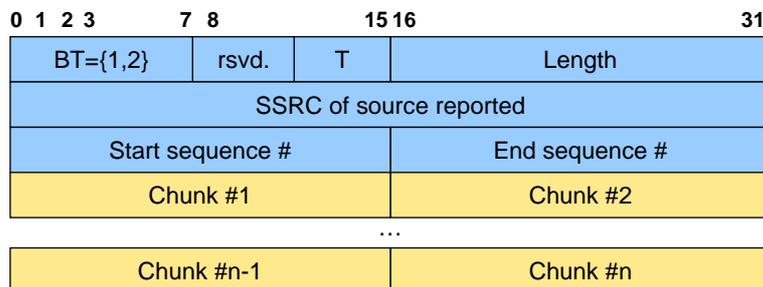
▶ Specific report blocks



RTCP XR: Detailed Packet Reporting (1)

▶ Report (individual) lost and duplicate packets

- Runlength encoding or bit maps of sequences ("chunks")



▶ Run length:

0	R
---	---

 # packets lost (R=0) or received (R=1)

▶ Bit vector:

1

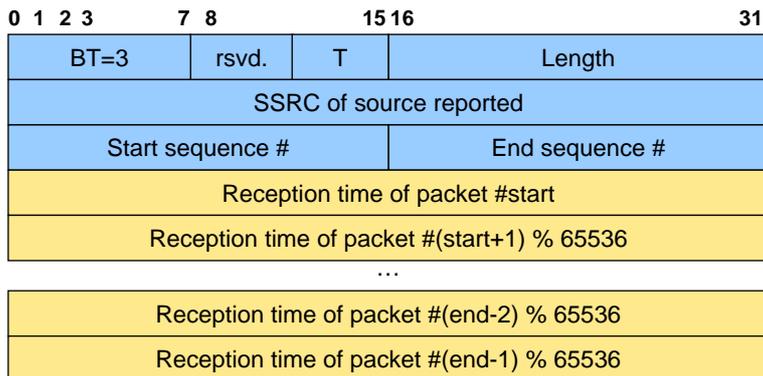
 Bit vector (0 = lost, 1 = received packet)

▶ Null chunk: 0x0000



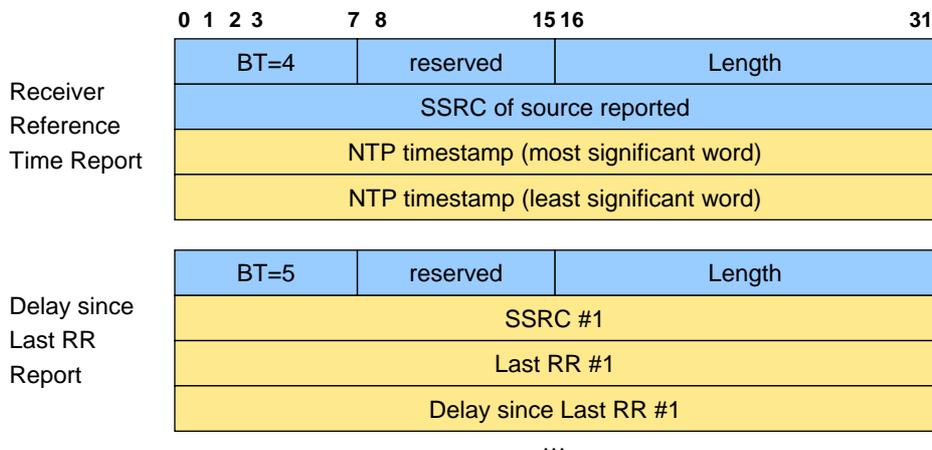
RTCP XR: Detailed Packet Reporting (2)

- ▶ Record individual packet reception times
 - Ideally obtained as close to the incoming interface as possible
- ▶ Middle 32 bits of the NTP timestamp



RTCP XR: Receiver Side RTT Calculation

- ▶ Operation similar to RTCP SR+RR mechanism
- ▶ Receivers report sending and selective reception timestamps, too





RTCP XR: Statistic Summary + VoIP Metrics

- ▶ Detailed report on reception statistics for a certain packet interval
 - BT=6
 - Lost, duplicate packets
 - Min, max, mean jitter + standard deviation

- ▶ VoIP Metrics (BT=7)
 - Lost packets (network) + discarded packets (local jitter buffer = late packets)
 - Identification of (loss/discard) bursts and (loss/discard) gaps
 - Burst: first, ..., last lost packet in a sequence with loss rate > threshold (Gmin)
 - Gap: Runs of packets which are not in a burst
 - Gap + Burst duration (ms) and respective packet loss rate

111111111101111111111111111111110001010110011111101100111111111111110111111101

Gap

Burst

Gap



RTCP XR: VoIP Metrics

- ▶ Delays
 - RTT delay
 - End system delay (estimated)

- ▶ Signal information
 - Signal + noise level

- ▶ Call quality
 - R factor, extended R factor + MOS listening, conversational

- ▶ Configuration parameters
 - Gmin, packet loss concealment, jitter buffer operation (adaptiveness)

- ▶ Jitter buffer parameters
 - Delay, maximum delay (observed), absolute maximum delay (buffer size)



RTCP Operation



RTCP Transmission Interval

- ▶ Must scale with the number of group members
 - Must not take up too much network capacity (rate-limited!)
- ▶ Overall “RTP session bandwidth”
 - Includes UDP and IP header overhead
 - Provided by the application (i.e. not measured dynamically)
- ▶ Default: 5% of the session bandwidth for RTCP
 - Takes role (sender or receiver) into account
 - Up to 25% of session members are senders
 - 3.75% for receivers, 1.25% for senders
 - More than 25% of session members are senders
 - Share data rate proportionally
- ▶ May be modified by profiles
 - Parameters S and R to indicate relative share for senders/receivers
- ▶ Scalable RTCP transmission interval
 - Based upon the group size, RTCP data rate, average RTCP packet size



RTCP Variables for Bandwidth Calculation

- ▶ Data rate
 - Session bandwidth
 - R, S: Receiver, sender bandwidth share
 - Average RTCP packet size (moving average)
- ▶ Time
 - T_p last time an RTCP packet was sent
 - t_c current time
 - t_n next scheduled transmission of an RTCP packet
- ▶ Membership
 - $p_{members}$ # members when t_n was last computed
 - $members$ current # members
 - $senders$ # senders in the session
 - n relevant # of members (depending on role, etc.)
- ▶ Intervals
 - T_d Deterministic calculated interval
 - T Calculated interval
 - T_{min} minimal interval between RTCP packets

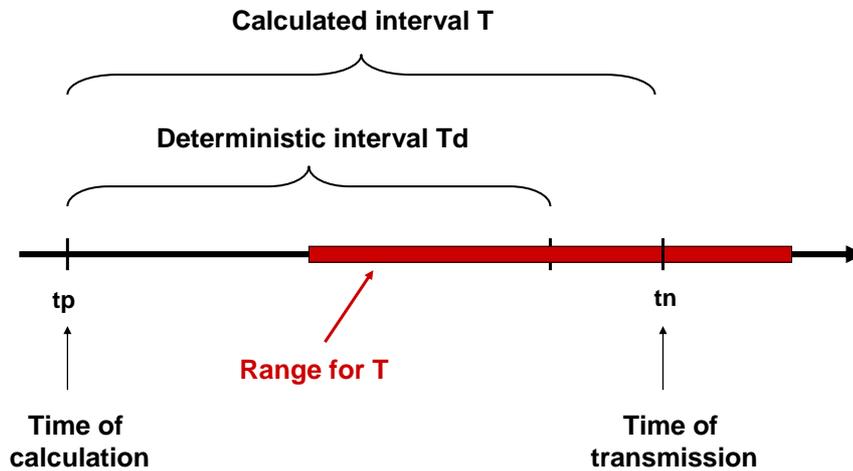


Basic Operation

- ▶ Determine role (sender or receiver)
 - Derive n as # of relevant members for calculation
 - Derive relevant bandwidth share
- ▶ $C = \text{average RTCP size} / \text{relevant bandwidth share}$
- ▶ $T_d = \max(T_{min}, n * C)$
- ▶ $T = \text{Random}[0.5 - 1.5] * T_d$



Basic RTCP Interval Calculation



Timer Reconsideration

- ▶ The group size may change between t_p and t_n
- ▶ Particularly during startup and shutdown phase
 - Many users may join / leave during a short period of time
- ▶ Many joining parties: risk of RTCP implosion
- ▶ Algorithm for joining members
 - Validate the group size at time t_n before transmission
 - Recalculate T as above
 - If $t_p + T \leq t_c$ transmit RTCP packet and update variables
 - If $t_p + T > t_c$ set $t_n = t_p + T$ and set timer to expire at t_n
- ▶ Algorithm for leaving members
 - Adjust t_p , t_n according to the observed membership change
 - Factor: members / pmembers
 - Run every time a member leaves or times out



Extended Operation

- ▶ Determine role (sender or receiver)
 - Derive n as # of relevant members for calculation
 - Derive relevant bandwidth share
- ▶ $C = \text{average RTCP size} / \text{relevant bandwidth share}$
- ▶ $T_d = \max(T_{\min}, n \cdot C)$
- ▶ $T = \text{Random}[0.5 - 1.5] \cdot T_d$
- ▶ $T = T / e^{-1.5}$ ($T = T / 1.21828$)
 - Correction factor for timer reconsideration

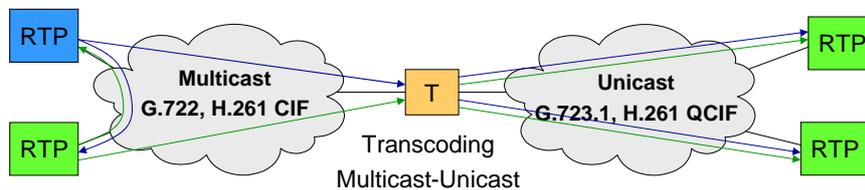


RTP/RTCP Transport and Multiplexing (1)

- ▶ RTP over UDP
 - Session Identification: a pair of destination transport addresses
 - Multicasting: Common IP multicast address as destination for all RTP entities
 - Unicasting: two independent sessions
 - Usual operation: 1 transport address RTP + 1 transport address RTCP
 - Typically the same IP address + 2 port numbers to differentiate
 - Original idea: RTP port is n (even), RTCP port is $n+1$ (odd)
 - Issues: dynamic port assignment, NATs: ports may now be arbitrary
- ▶ Further optimization (currently discussed in the IETF)
 - Use a single port for both RTP and RTCP
 - Motivation: NATs and firewalls
 - Need to open just one pin hole
 - Need to maintain just one port binding
 - Payload type name space allows for easy differentiation
 - Raises architectural issues though

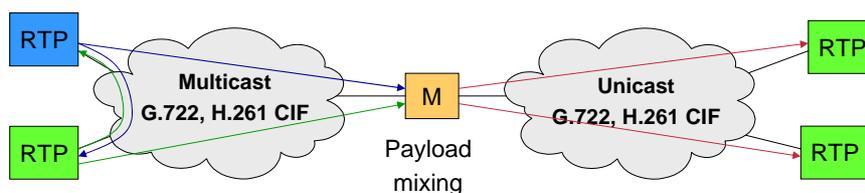
RTP Translator

- ▶ Intermediate system in an RTP session
- ▶ Operates at the transport level
- ▶ Connects two or more RTP clouds
- ▶ Leaves SSRC intact
 - Shared global SSRC space per session; end-to-end conflict resolution
- ▶ May operate on the payload, the packet size, the transport
 - IPv4 to IPv6 translation typically transparent to RTP



RTP Mixer

- ▶ Another intermediate system in an RTP session
- ▶ Creates a new media stream from one or more incoming streams
 - With its own SSRC id
 - Indicates input streams (= contributing sources) in CSRC field
 - Performs local dejittering, input synchronization, etc.
- ▶ Operates on the payload and may operate on everything else
 - Reduces bandwidth demand towards each receiver
 - Typically found in IP-based conference bridges





RTP Payloads



RTP Payload Types

- ▶ 7-bit payload type identifier
 - Some numbers statically assigned
 - Dynamic payload types identifiers for extensions – mapping to be defined outside of RTP (control protocol, e.g. SDP “a=rtpmap:”)

Payload formats defined for many audio/video encodings

- ▶ Conferencing profile document RFC 3551
 - Audio: G.711, G.722, G.723.1, G.728, GSM, CD, DVI, ...
- ▶ In codec-specific RFCs
 - Audio: Redundant Audio, MP-3, ...
 - Video: JPEG, H.261, MPEG-1, MPEG-2, H.263, H.263+, BT.656
 - Others: DTMF, text, SONET, ...
- ▶ Generic formats
 - Generic FEC, (multiplexing)



Media Packetization Schemes (1)

General principle:

- ▶ Payload specific additional header (if needed)
- ▶ Followed by media data
 - Packetized and formatted in a well-defined way
 - Trivial ones specified in RFC 3551
 - RFC 2029, 2032, 2035, 2038, 2190, 2198, 2250, 2343, 2429, 2431, RFC 2435, 2658, 2733, 2793, 2833, 2862, and many further ones
 - Guidelines for writing packet formats: RFC 2736
- ▶ Functionality
 - Enable transmission across a packet network
 - Allow for semantics-based fragmentation
 - Provide additional information to simplify processing and decoding at the recipient
 - Maximize possibility of independent decoding of individual packets

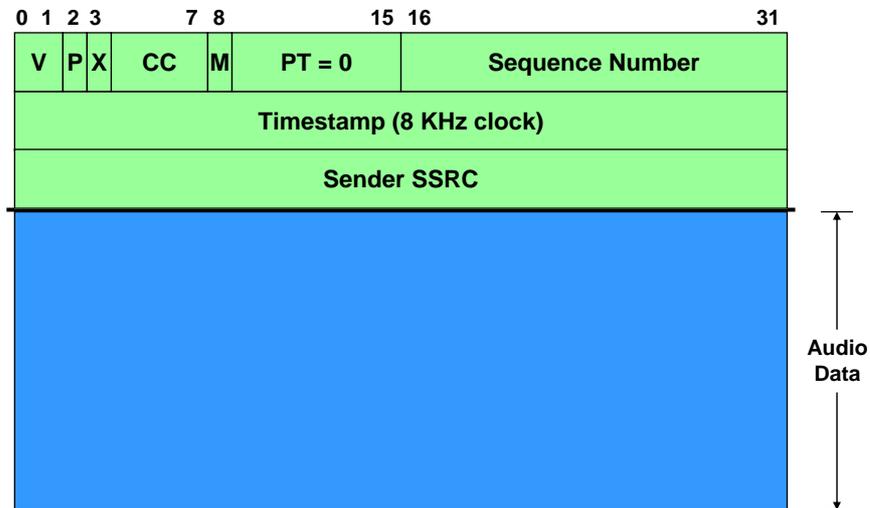


Sample RTP Payload Types

Illustrate a variety of approaches to deal with packet loss in the Internet



Audio over RTP: PCM



Video over RTP: H.261

Additional payload-specific header precedes payload

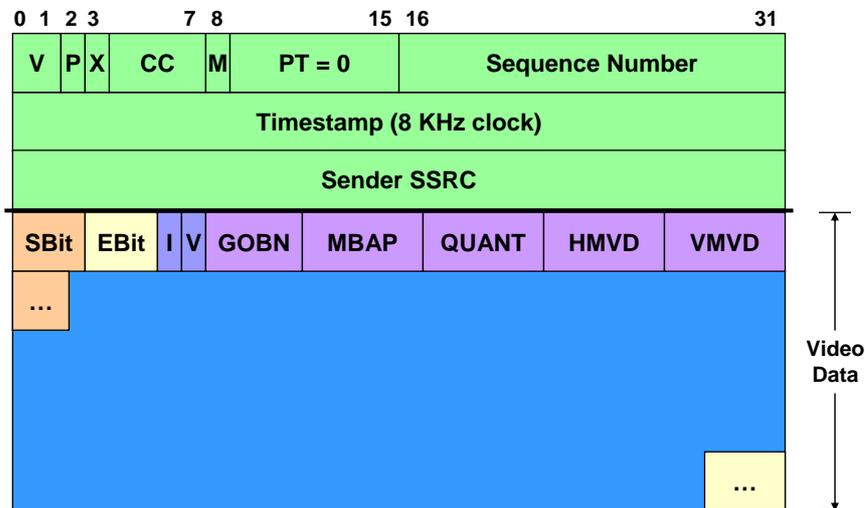
- ▶ To avoid expensive bit shifting operations
 - Indicate # invalid bits in first (SBit) and last (EBit) octet of payload
- ▶ Indicate Intra encoding (I bit)
- ▶ Indicate the presence of motion vector data (V bit)
- ▶ Carry further H.261 header information to enable decoding in the presence of packet losses

Further mechanisms for video conferencing

- ▶ FIR: Full Intra Request
 - Ask sender to send a full intra encoded picture
- ▶ NACK: Negative Acknowledgement
 - Indicate specific packet loss to sender



Video over RTP: H.261 (2)



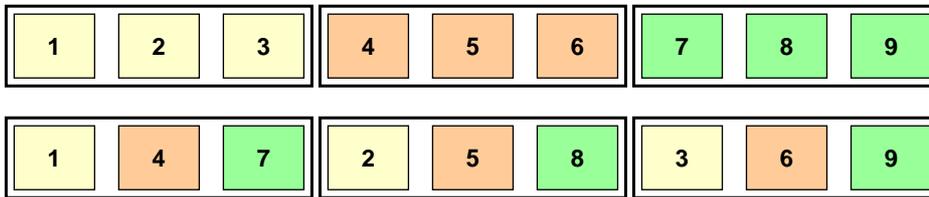
Media Packetization Schemes (2)

Error-resilience for real-time media

- ▶ Input: Observation on packet loss characteristics
- ▶ Generic mechanisms (RFC 2354)
 - Retransmissions
 - in special cases only (e.g. with no interactivity!)
 - Interleaving
 - Forward Error Correction (FEC)
 - media-dependent vs. media-independent
 - Generic FEC: RFC 2733
- ▶ Feedback loops for senders
 - based upon generic and specific RTCP messages
 - adapt transmission rate, coding scheme, error control, ...

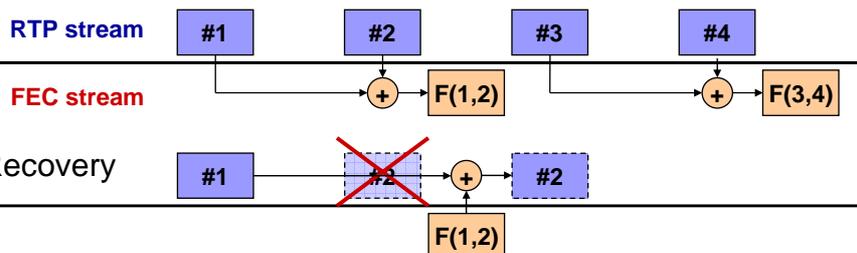
RTP Interleaving

- ▶ Distribute packets or packet contents for transmission
 - Avoid consecutive packet erasures in case of (burst) losses
 - Avoid loss of large consecutive data portions in case of single packet losses
- ▶ Motivations
 - Human perception tolerates individual losses better (with error concealment)
 - Make simple FEC schemes work better with burst losses (e.g. XOR)
- ▶ Drawback
 - Re-ordering causes additional delay



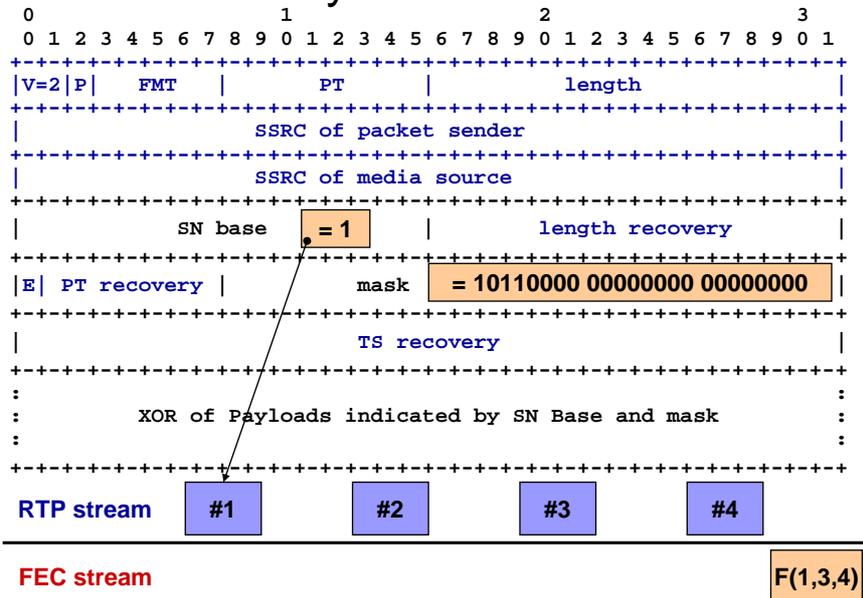
RTP FEC (RFC 2733)

- ▶ Forward Error Correction scheme for RTP packets
 - Media-independent, flexible FEC (that can be enhanced)
- ▶ Simple XOR-based (parity) FEC
 - $P_{fec} = P_1 \text{ XOR } P_2 \text{ XOR } P_3 \text{ XOR } \dots \text{ XOR } P_n$
 - Allows reconstruction of any **single** missing packets of P_1, \dots, P_n, P_{fec}
- ▶ RTP FEC stream transmitted independently of RTP stream
 - Separate transport address (IP address, port number)
 - Different SSRC



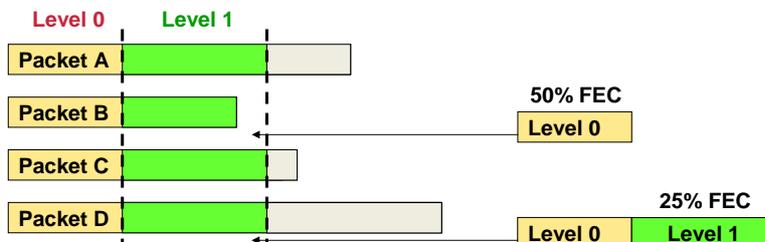


RTP Parity FEC Packet format



Unequal Error Protection

- ▶ Observation: not all parts of a packet are equally important
 - Beginning of packet contains headers/parameters, more relevant contents
 - Holds for both audio and video
- ▶ Uneven Level Protection (ULP)
 - Create independent parity packets for different parts of packets
 - Allows for selectively more overhead for the more important parts

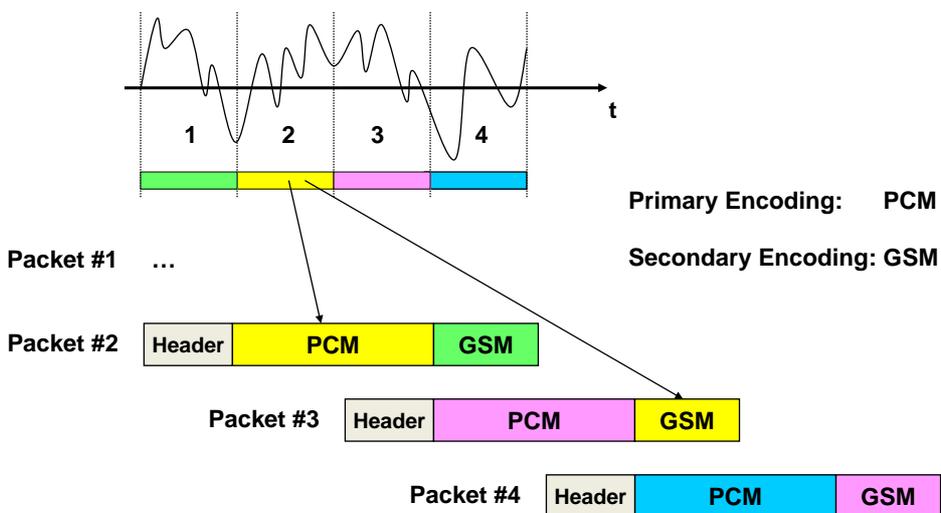


- ▶ Related thoughts: partial checksums
 - Live with bit errors in the less important parts (rather than dropping a packet)

Audio Redundancy Coding (1)

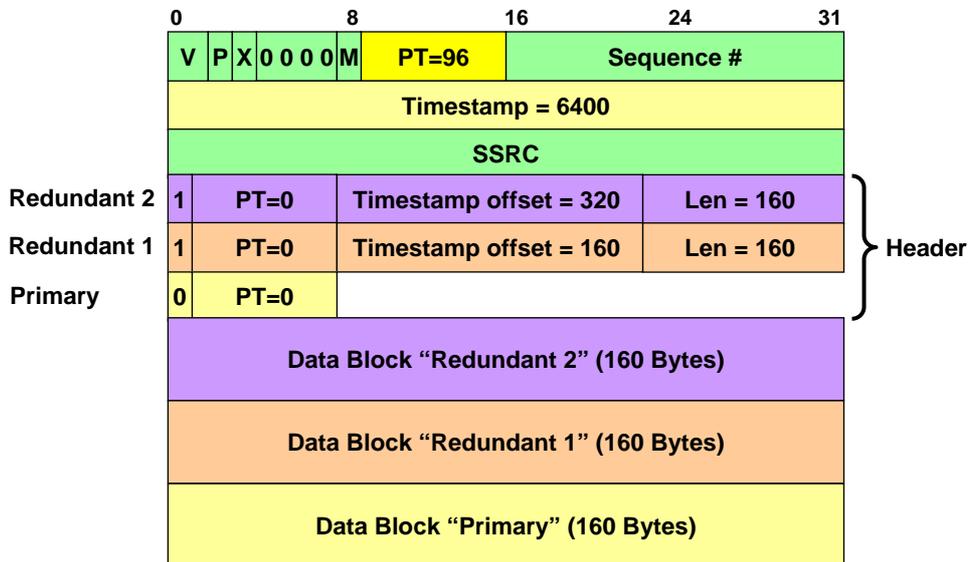
- ▶ Audio Packets are small!
 - have to be because of interactivity
 - avoid large packetization delay
 - packet loss primarily depends on packet rate
 - rather than packet size
- ▶ Payloads for multiple time slots in one packet
 - send redundant information in packet n to reconstruct packets $k, \dots, n-1$
 - redundant information typically sent at lower quality
 - details defined in RFC 2198
 - uses dynamic payload type
- ▶ Format specification, e.g. using SDP
 - `m=audio 20002 RTP/AVP 96 0 0 0`
 - `a=rtpmap:96 red/8000/1`

Audio Redundancy Coding (2)





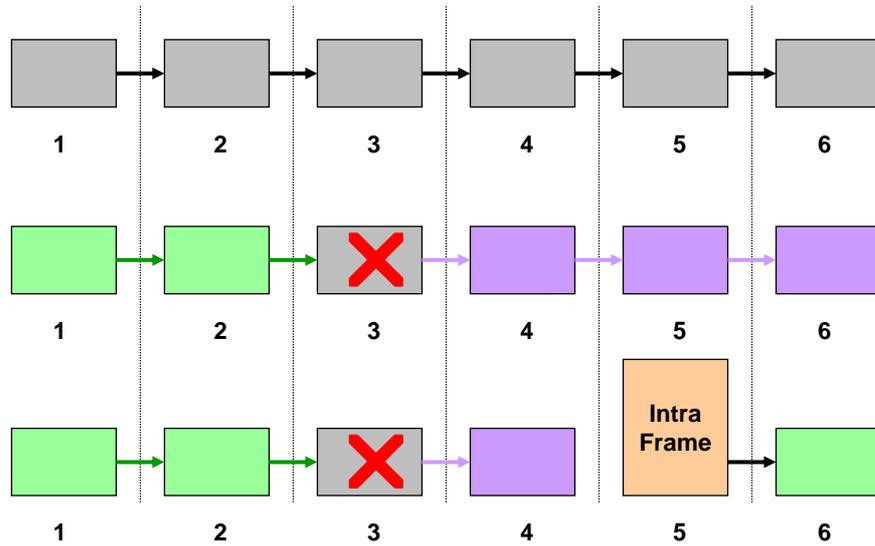
Audio Redundancy Coding (3)



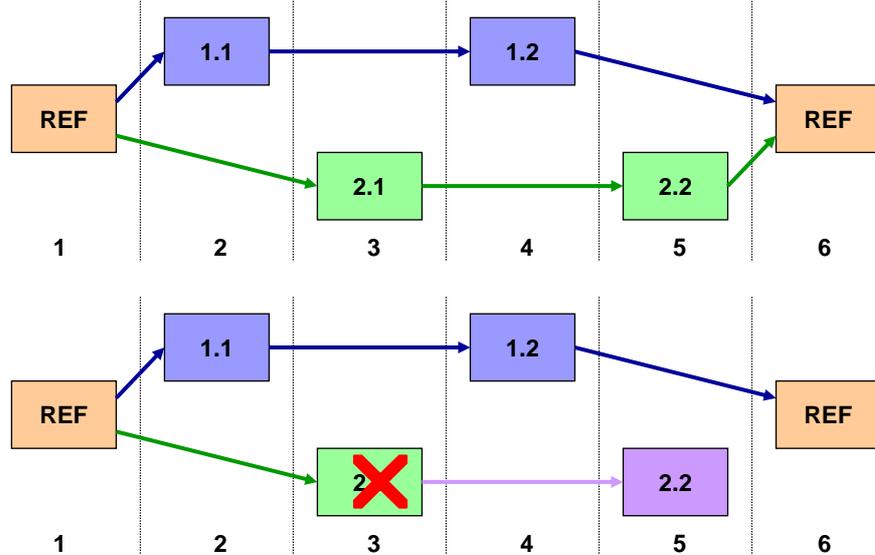
Video Redundancy Coding (1)

- ▶ Video redundancy coding
 - For H.263+ video streams
 - Transmit several interleaved sequences of predicted frames (threads) instead of one
 - improves error resilience against packet loss
- ▶ Principle
 - create several (n) independently decodable streams
 - achieved by choosing different reference pictures
 - decode only streams with no packet losses
 - reduces temporal resolution by 1/n-th per affected stream
 - bit rate penalty due to larger deltas between frames
 - RFC 2429, revised version in progress

Video Redundancy Coding (2)

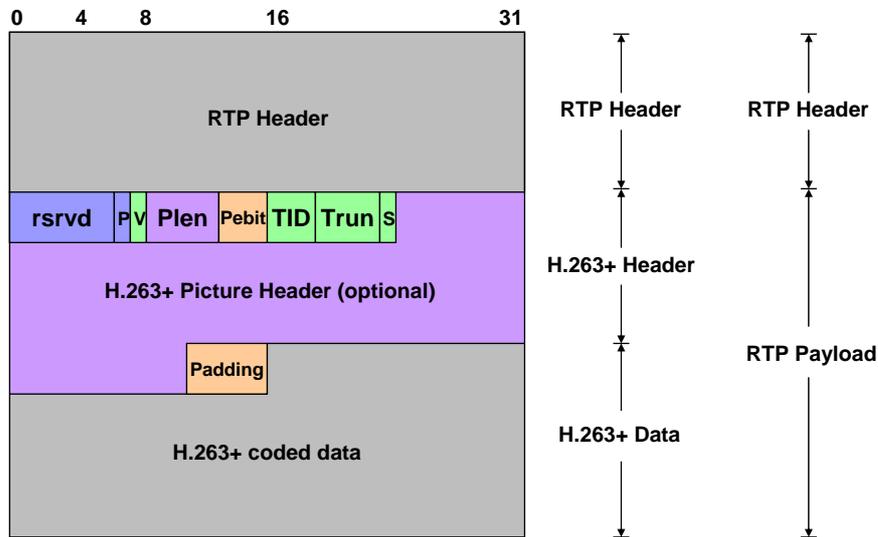


Video Redundancy Coding (3)





Video Redundancy Coding (4)

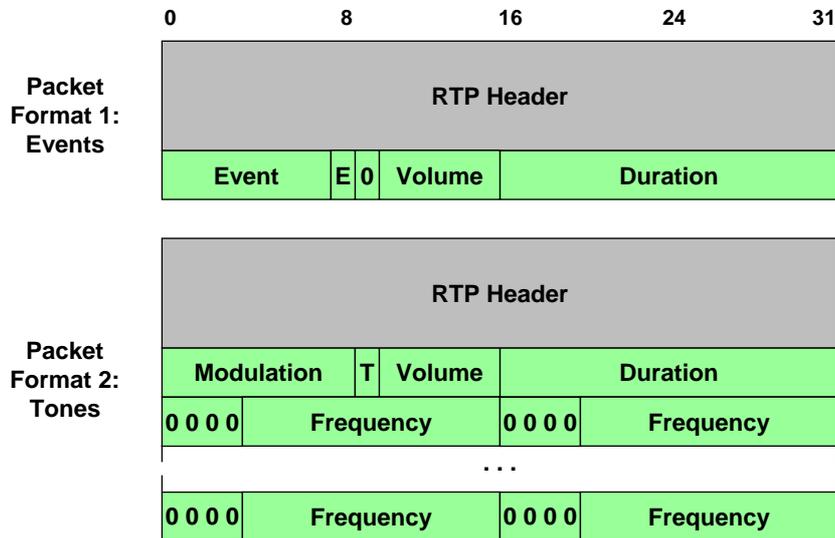


DTMF over RTP (1)

- ▶ DTMF digits, telephony tones, and telephony signals
 - two payload formats
 - 8 kHz clock by default
 - audio redundancy coding for reliability
- ▶ Format 1: reference pre-defined events
 - 0 - 9 * # A - D (Hook)Flash [17]
 - modem and fax tones [18]
 - telephony signals and line events [43]
 - dial tones, busy, ringing, congestion, on/off hook, ...
 - trunk events [44]
 - specified through identifier (8-bit value), volume, duration
- ▶ Format 2: specify tones by frequency
 - one, two, or three frequencies
 - addition, modulation
 - on/off periods, duration
 - specified through modulation, n x frequency, volume



DTMF over RTP (2)



RTP Payload Type Overview (1)

- ▶ RFC 3551 Collection of simple packetization formats (formerly RFC 1890)
- ▶ RFC 2029 Sun CellB Video encoding
- ▶ RFC 2032,4587 H.261 video
- ▶ RFC 2435 JPEG video (was RFC 2035)
- ▶ RFC 2250 MPEG-1/MPEG-2 video (was RFC 2038)
- ▶ RFC 2190 H.263 video (historic)
- ▶ RFC 2343 Bundled MPEG
- ▶ RFC 2429 H.263+ video & video redundancy support
- ▶ RFC 2431 BT.656 video
- ▶ RFC 2658 PureVoice audio
- ▶ RFC 2793,4103 Text conversation
- ▶ RFC 2833 DTMF, telephony tones, and telephony signals
- ▶ RFC 2862 Real-time Pointers
- ▶ RFC 3016 MPEG-4 Audio/visual streams
- ▶ RFC 3047 G.722.1 audio
- ▶ RFC 3119 Loss-tolerant format for MP3
- ▶ RFC 3189 DV video
- ▶ RFC 3190 12-bit DAT and 20-/24-bit linear audio



RTP Payload Type Overview (2)

- ▶ RFC 3267,4352 Adaptive Multirate (AMR, AMR-WB+) audio
- ▶ RFC 3389 Comfort noise
- ▶ RFC 3497 SMPTE 292M video
- ▶ RFC 3557 ETSI Distributed speech recognition (ES 201 108)
- ▶ RFC 3558 Enhanced variable rate codecs and selectable mode vocoders
- ▶ RFC 3640 MPEG-4 elementary streams
- ▶ RFC 3952 Low Bit Rate Codec (iLBC) Speech
- ▶ RFC 3984 H.264 Video
- ▶ RFC 4040 64 kbit/s Transparent Call
- ▶ RFC 4060 Distributed speech recognition encoding (ES 202 050/211/212)
- ▶ RFC 4175,4421 Uncompressed Video
- ▶ RFC 4184,4598 AC-3 Audio, Enhanced AC-3
- ▶ RFC 4298 BroadVoice Speech codec
- ▶ RFC 4348,4424 Variable Rate Multimodal Wideband Audio (VMR-WB)
- ▶ RFC 4351 Text conversation interleaved with audio stream
- ▶ RFC 4396 3GPP Timed Text
- ▶ RFC 4425 Video Codec 1 (VC-1)
- ▶ RFC 4588 Retransmission payload format

Many more to come...



RTP Extensions

- ▶ Timely feedback from receivers to senders
- ▶ RTP Retransmissions
- ▶ Support for Source-specific Multicast (SSM)



RTCP Feedback Issues

- ▶ Senders provide regular information about media stream
 - Seems ok
- ▶ Receivers transmit RTCP at somewhat regular intervals
- ▶ RTCP RRs provide long-term statistics on reception quality
- ▶ Senders can adapt transmission strategy to receiver observations
 - Different codecs, data rate, etc.
- ▶ BUT: No short-term feedback possible
 - Error repair or mitigation impossible
 - Not suitable for congestion control
- ▶ Problem: Value of receiver feedback decreases over time
 - Repair more expensive at later times
 - Artifacts become noticeable to the user



Approach: RTCP-based Feedback

- ▶ New Profile for RTP: AVPF

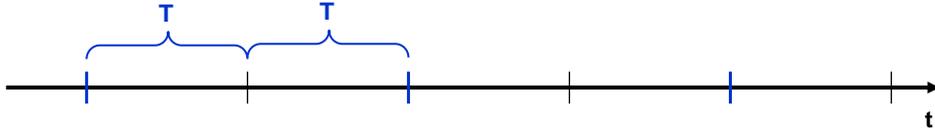
Idea:

- ▶ Packet losses are usually rare
- ▶ Provide statistical chance of virtually immediate feedback from receiver(s) to sender
- ▶ Keep the basic RTCP properties
- ▶ Eliminate T_{min}

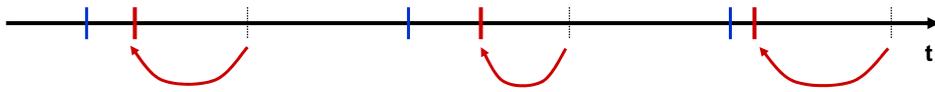
- ▶ Work most efficiently with unicast
- ▶ Also scale to moderate group sizes

Overview

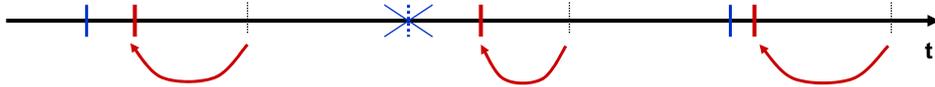
Regular RTCP operation (depicted w/o randomization, i.e. $T = T_d$)



Allow (at most every other) RTCP packet to be sent earlier

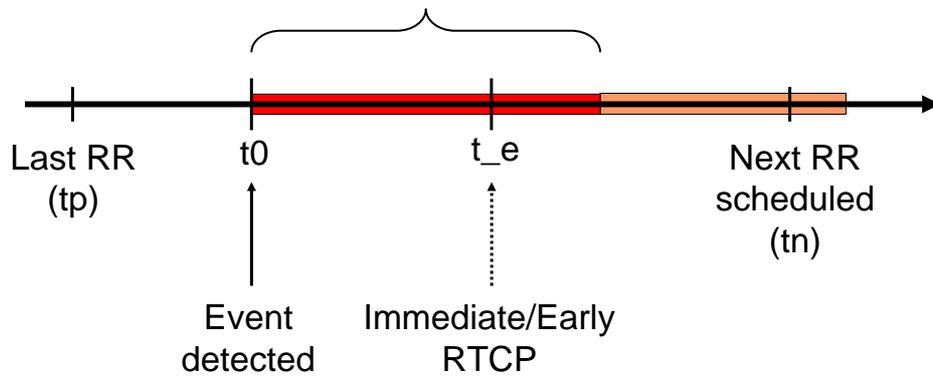


Allow to reduce the number of regular RTCP packets (w/o affecting RTCP rate)



RTCP Feedback Timing

$$T_{\text{dither_max}} = f(\text{group size, ...})$$



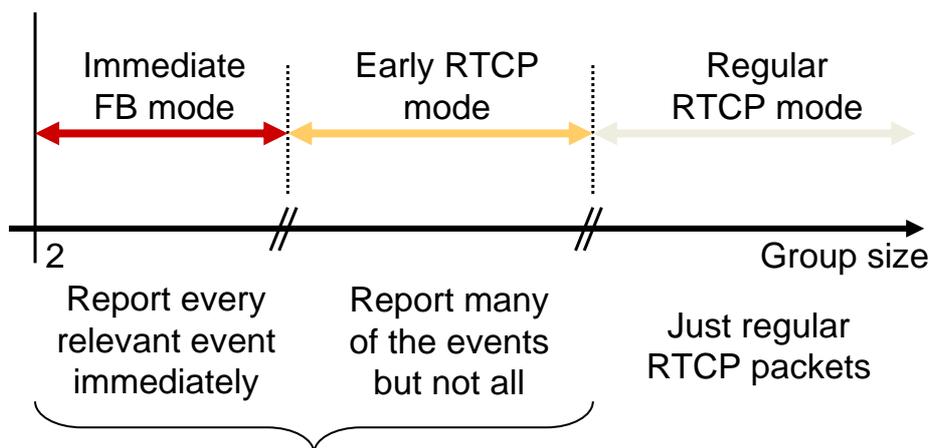
Delay calculation

$$T_{\text{dither_max}} = \begin{cases} 0 & \text{if grp size} = 2 \\ I * T & \text{otherwise} \end{cases}$$

Simulated guess: $I = 0.5$

Better approach: use RTT measurements!
But those are only available to senders...
Mixed operation (using T_d and RTT) will not work.

Modes of Operation



Send feedback + regular RTCP packets

RTCP Types of Feedback

- ▶ ACK Mode
 - Positive acknowledgements for received packets
 - Restricted to point-to-point operation
- ▶ NACK Mode
 - Negative acknowledgments e.g. for missing packets or other events
 - Scalable with suppression technique
- ▶ Other types of feedback conceivable
- ▶ Transport layer feedback packets (Generic NACK)
 - Identifies missing or received packets
- ▶ Payload-specific feedback packets
 - Specific to certain codecs (e.g. video)
 - Picture / frame loss indication, reference picture selection
- ▶ Application feedback packets

RTCP Feedback Packet Format

```

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|  FMT  |      PT      |      length      |
+-----+-----+-----+-----+-----+-----+
|
|      SSRC of packet sender
|
+-----+-----+-----+-----+-----+-----+
|
|      SSRC of media source
|
+-----+-----+-----+-----+-----+-----+
:      Feedback Control Information (FCI)      :
:

```

Example: Generic NACK Packet

```

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
+-----+-----+-----+-----+-----+-----+-----+-----+
|
|      PID      |      BLP      |
+-----+-----+-----+-----+-----+-----+

```



RTCP Feedback Packet Format (2)

Example: Slice lost indication

```

+-----+-----+-----+-----+-----+-----+-----+-----+
|                               First                               |
|                               Number                               |
|                               PictureID                             |
+-----+-----+-----+-----+-----+-----+-----+-----+

```

Example: Reference Picture Selection

```

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
+-----+-----+-----+-----+-----+-----+-----+-----+
|          PB          |0| Payload Type| Native RPSI bit string |
+-----+-----+-----+-----+-----+-----+-----+-----+
| defined per codec   ...                               | Padding (0) |
+-----+-----+-----+-----+-----+-----+-----+-----+

```



Example for Statistical Feedback

- ▶ Applicability of feedback depends on many parameters
 - Group size, RTP & RTCP bandwidth, application requirements
- 256 kbit/s video stream, 30 frames per second, 1500 bytes MTU
 Single sender, > 3 receivers (i.e. 3.75% RTP bandwidth for receivers)
 H.263+ with approximately 1 packet per frame
 5% packet loss, equally distributed, receiver independence
- Statistically yields 3 losses every two seconds per receiver
 $3.75\% * 256 \text{ kbit/s} = 9.6 \text{ kbit/s}$ for all receivers
 Assuming 120 bytes (= 960 bits) per RTCP packet: 10 packets / s
- If every receiver reports every loss event: 6 – 7 receivers on average
 If reporting every other loss event is sufficient: ~14 receivers
- Increases further if losses are correlated in some fashion

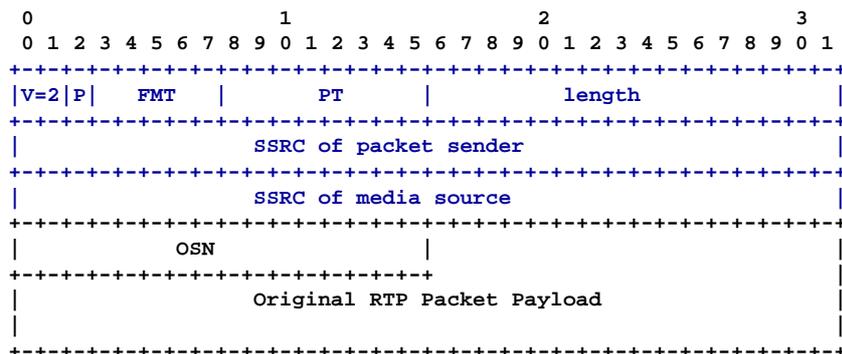


RTP Retransmissions

- ▶ Explicit repair mechanism for RTP streams
- ▶ Works for applications with acceptable higher latency
 - E.g. media streaming
- ▶ Applicable to point-to-point and small group scenarios
- ▶ Used with RTCP feedback extensions
- ▶ Approach
 - Original RTP stream
 - Augmented by retransmission RTP stream
 - Mapped to different RTP sessions or sender SSRCs
 - Use always different sessions for multicasting
 - Keeps the retransmission scheme backward compatible
 - Does not confuse RTCP statistics
 - Works with all payload types
 - Allows for multiple payload types in a session

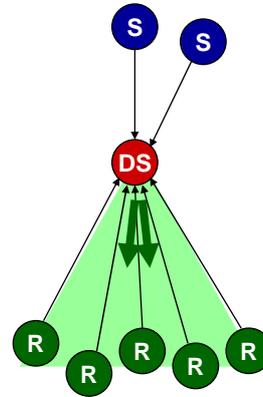


RTCP Retransmission Packet Format

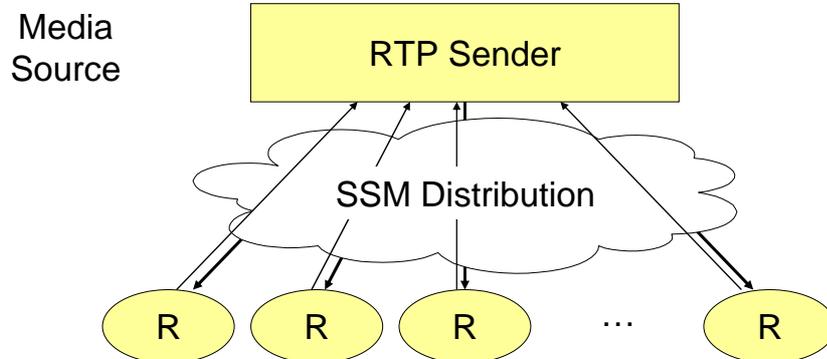


RTCP for SSM

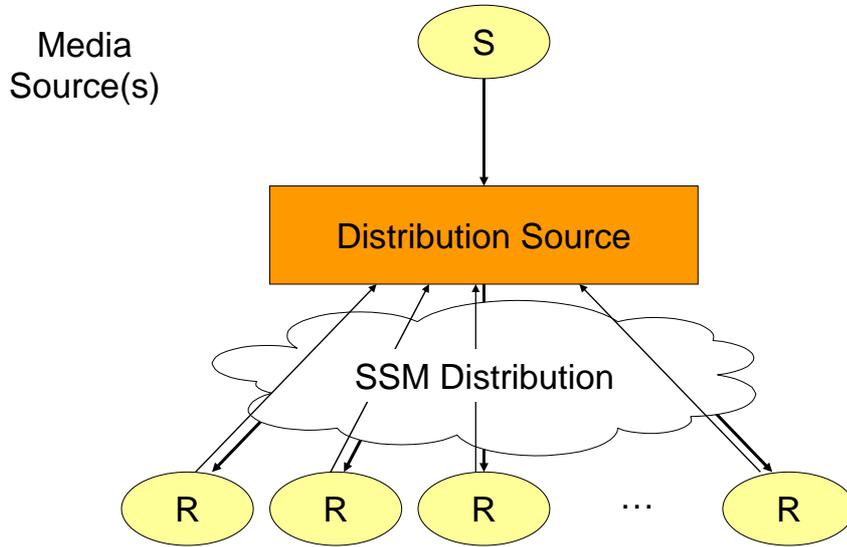
- ▶ Multicast connectivity unidirectional
 - From **Distribution Source** to receivers
 - Opposite direction needs to use unicasting
 - May follow different network path
- ▶ Result: no direct communication between receivers
- ▶ Adaptations required to make RTCP work
 - Estimate group size
 - Adjust timing of RTCP transmission (adhere to bandwidth limit)
 - Resolve SSRC collisions
- ▶ Two basic modes of operation
 - Make distribution source reflect RTCP traffic back to receiver
 - Provide summaries of relevant information along with sender reports



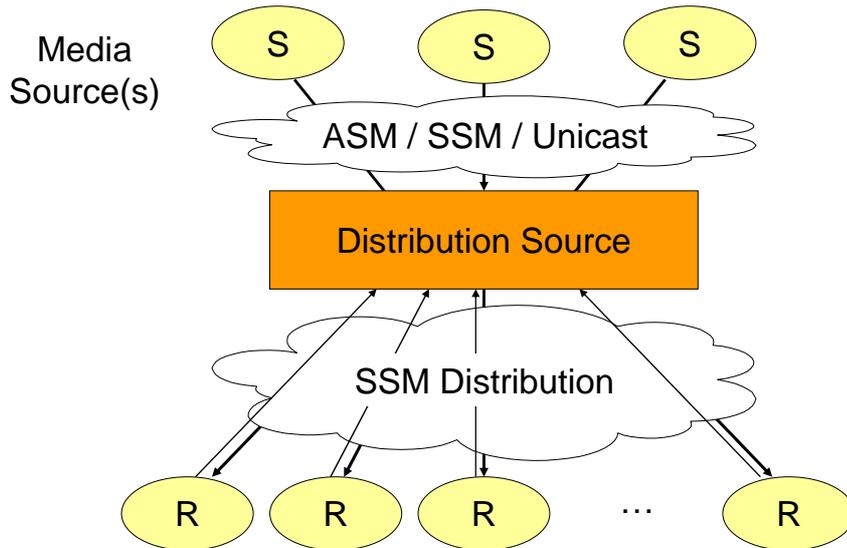
RTCP SSM Overview



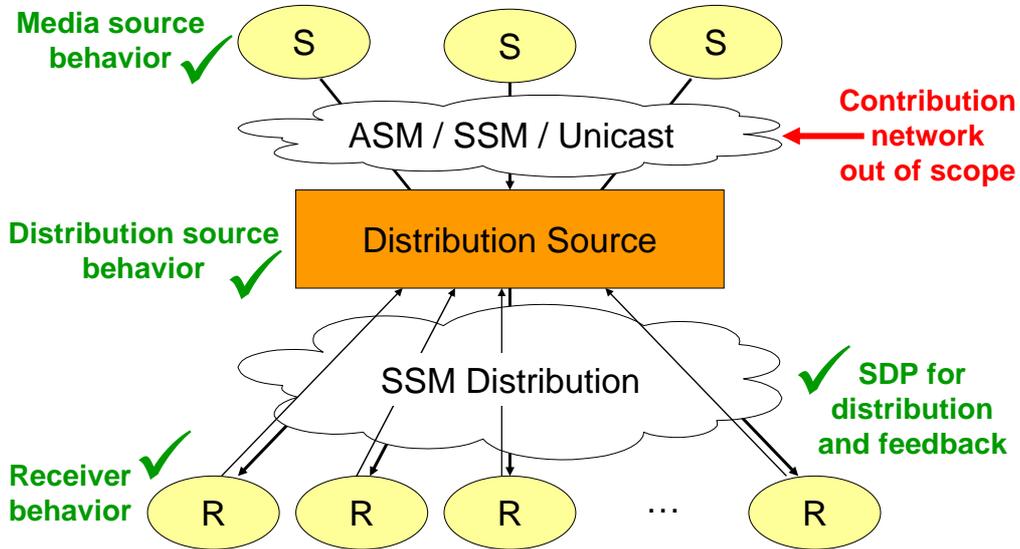
RTCP SSM Overview



RTCP SSM Overview



RTCP SSM Overview



Simple Feedback Model

- ▶ Distribution source reflects packets back to receivers
 - Simple mirroring at the transport / application layer
- ▶ Uses the bandwidth share for receivers for distribution
 - Not an issue: non-overlapping paths
- ▶ Increases delay for inter-receiver communication
 - Particularly with asymmetric networks
 - May impact e.g. feedback suppression
- ▶ Required for all RTCP packets that cannot be summarized
 - Unknown extensions
 - Packets that require knowledge of the originator
- ▶ Particularly applies to RTCP APP packets

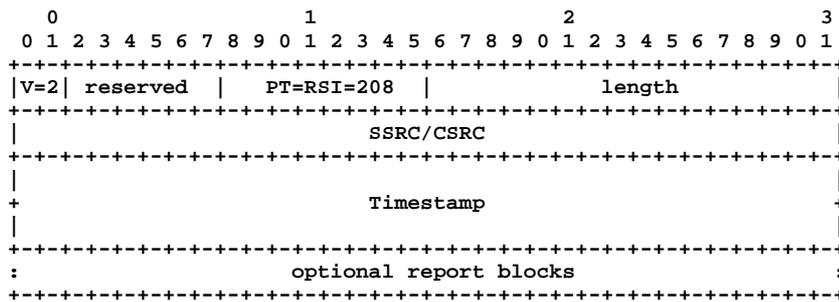


Feedback Summary Model

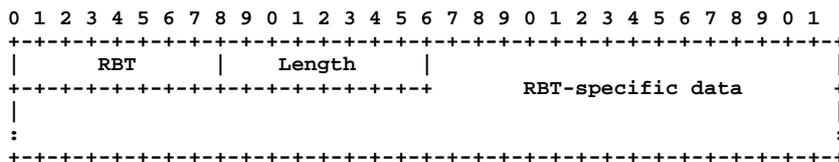
- ▶ Distribution source collects information from receivers
- ▶ Aggregates the information over time
- ▶ Distributes representative summaries back to receivers
 - In somewhat regular intervals
 - Saves bandwidth compared to simple reflection
 - Uses (part of) receiver rate in addition to sender rate
 - Acts as another receiver from an RTP/RTCP perspective (own SSRC)
- ▶ New RTCP packet: Receiver Summary Information (RSI)
 - Contains distributions for RTCP receiver statistics
 - Relative loss, cumulative loss, RTT, jitter
 - Allows receivers to relate themselves to group reception quality
 - Simple form: general statistics report on loss and jitter
 - Feedback target address
 - Where to unicast feedback packets to
 - SSRC collision reports
 - RTCP bandwidth indication



RTCP RSI Packet Format



Report Block:





Detailed Statistics Sub-Report Blocks

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	2	3	4	5	6	7	8	9
SRBT										Length										NDB										MF									
Minimum Distribution Value																																							
Maximum Distribution Value																																							
Distribution Buckets																																							
...																																							
...																																							

- ▶ Used for
 - Loss, Jitter, RTT, Cumulative Loss
- ▶ Reflects information collected from RTCP RRs



Other Report Blocks

- ▶ Feedback target address
 - In-band signaling for distribution source address
 - Security!
 - ▶ SSRC Collision
 - Initiate selection of new SSRCs
 - ▶ General statistics
 - Average loss, average jitter, highest cumulative loss
 - Calculated from received RTCP RRs
 - ▶ RTCP Bandwidth indication
 - ▶ Group size and average RTCP packet size
- } Pace RTCP RRs



RTP Specs (Summary)

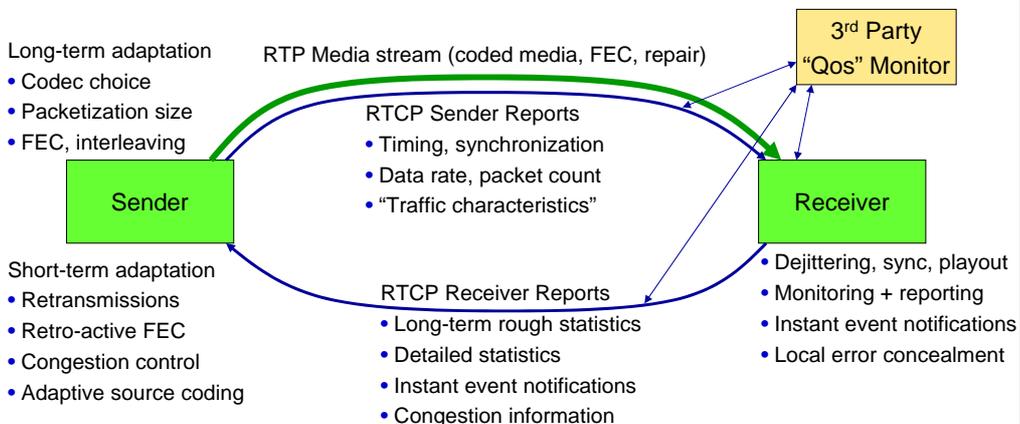
- ▶ RFC 3550 Base specification (formerly RFC 1889)
- ▶ RFC 3551 RTP Profile for Audio and Video Conference with minimal control (was RFC 1890)
- ▶ RFC 2198 Redundant (Audio) coding
- ▶ RFC 2508 RTP header compression for low-speed links
- ▶ RFC 2733,3009 Generic FEC
- ▶ RFC 2736 Guidelines for writers of RTP payload specifications
- ▶ RFC 2762 Group membership sampling ("timer reconsideration")
- ▶ RFC 3095 Robust header compression for RTP (among others)
- ▶ RFC 3096 Requirements for robust IP/UDP/RTP header compression
- ▶ RFC 3158 RTP testing strategies
- ▶ RFC 3242 Link-layer assisted profile for IP/UDP/RTP header compression
- ▶ RFC 3243 Requirements & assumptions for 0-byte IP/UDP/RTP header compression
- ▶ RFC 3409 Lower-layer guidelines for robust IP/UDP/RTP header Compression
- ▶ RFC 3545 Enhanced compressed RTP (CRTP) for high-delay links
- ▶ RFC 3555 MIME registrations of RTP payloads
- ▶ RFC 3611 RTCP XR extension
- ▶ RFC 3711 Secure RTP (SRTP)
- ▶ RFC 4362 Robust Header Compression for IP/UDP/RTP
- ▶ RFC 4383 TESLA for SRTP
- ▶ RFC 4571 Framing RTP over Connection-oriented Transport
- ▶ RFC 4585,4586 RTCP Feedback
- ▶ RFC 4588 RTP Payload format for retransmissions



Summary: Applying RTP

▶ Adaptive real-time applications

- Tunable feedback loop for individual and group communications
- From reporting per 5s and more to event-driven to once per RTT





Present Issues and Concluding Remarks

- ▶ Implementing RTCP?
 - Yes—obviously it helps implementing good real-time applications
 - Yet, many VoIP applications don't do it
- ▶ Signaling: RTP vs. RTCP
 - RTCP sent infrequently—sufficient for signaling?
 - Frequency of RTCP vs. overhead
 - RTP level (e.g., congestion control) vs. application level (tunneled signaling protocol)
 - Shim layer in RTP?
 - Unidirectional media streams?
 - Demultiplexing?
- ▶ Reliability in RTP and RTCP
 - Retransmissions and FEC for RTP
 - Positive acknowledgements for RTCP?
 - Explicit messages vs. implicitly derived from data
- ▶ Maintaining group communication capabilities in RTP/RTCP
 - Various exceptions defined
- ▶ Important: Maintaining RTP's architectural integrity



RTP / RTCP Bandwidth Calculation

- ▶ Senders and receivers estimate group size (independently!)
 - # senders from SRs
 - # receivers from RRs
 - Consider BYE packets
- ▶ RTCP bandwidth: 5% of RTP session bandwidth
 - 5% for both senders and receivers if >25% senders
 - 1.25% for senders and 3.75% for receivers otherwise
- ▶ Sample of RTCP packets sent and received
 - Average number of bytes per time unit currently transmitted
 - Consider whether you are sender or receiver
- ▶ Last time own packet was sent
 - Calculate expected next time to send based upon above data rate
 - Use a minimum of 5 seconds (initially, 2.5 seconds)
 - Dither with random function: 0.5 .. 1.5 of calculated interval
 - Timer reconsideration: double check values when timer expires