# Audio/Video Transport Working Group:
# Meeting Agenda
# Overview of Revised Real-time Transport Protocol (RTPv2)

Stephen Casner
University of Southern California
Information Sciences Institute

December 6, 1994

# Agenda

- Status of Real-time Transport Protocol
- Overview of RTP specification for newcomers
- Discussion of changes since Toronto IETF in July
  - Agreement on choices for “open issues”?
  - Comments on the specification?
- Report on RTP in “vic” video tool — McCanne
- Report on RTP in “Loki” video tool — Kastenholz
- Presentation of new drafts MPEG & Cell-B over RTP
- Future working group activity

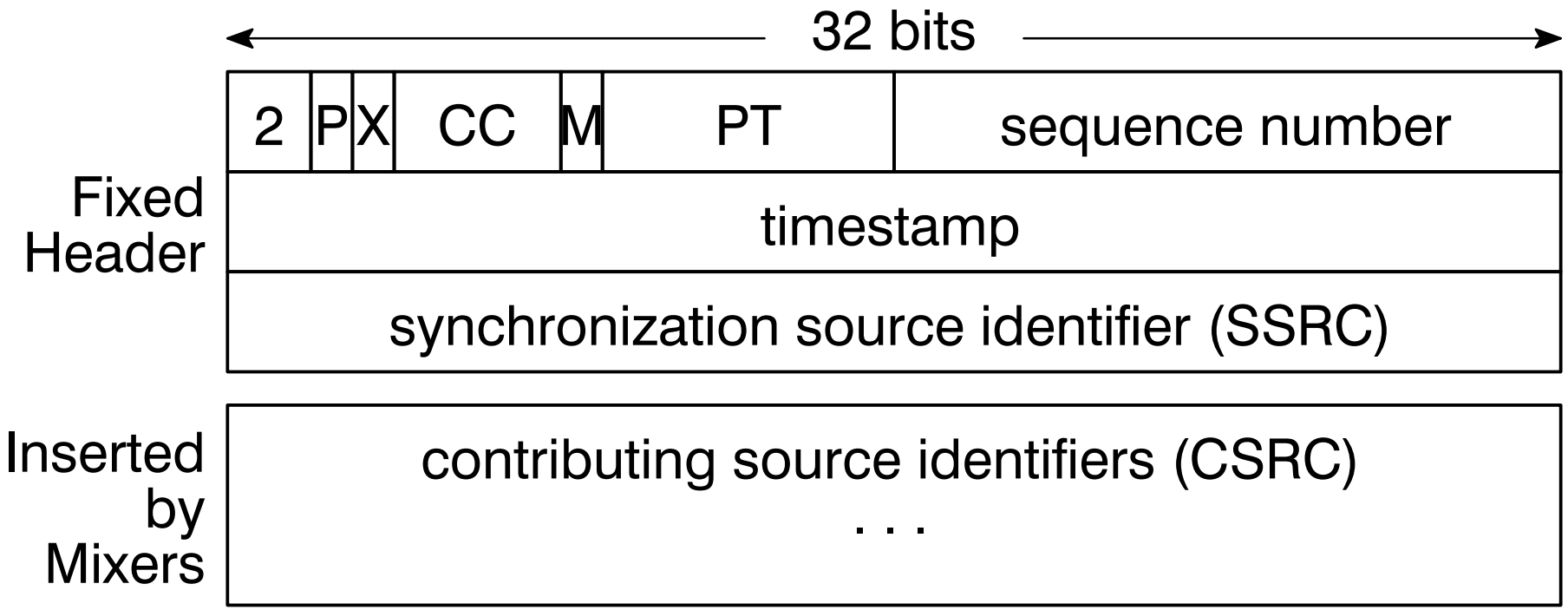
# Status of RTP

- RTP version 2 spec draft-ietf-avt-rtp-05.txt was presented at Toronto IETF in July
- Consensus to submit for Area Directorate review as soon as “TBD”s filled in
- Further work by the authors
- Revised draft-ietf-avt-rtp-06.txt posted last week is mostly done but not complete
  - A few small but incompatible changes
  - More explanatory text

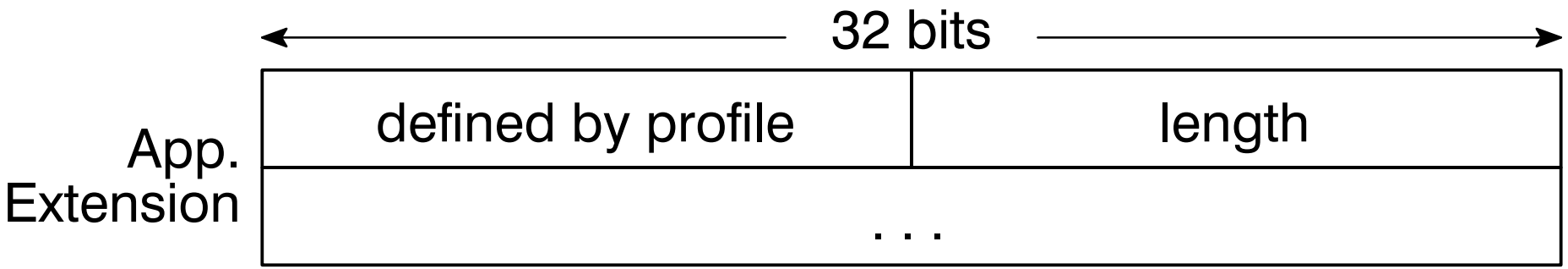
# Real-Time Transport Protocol — Introduction

- Transport for data with an inherent notion of time
  - Provides end-to-end services like seq. #, timestamp
  - Provides scalable multicast distribution feedback
  - Works with UDP to get port numbers and checksum
  - Does not make any delivery guarantees
  - Does not necessarily provide all control info
    *(See MMUSIC WG tomorrow, same time)*
- A new style of protocol, following Application Level Framing and Integrated Layer Processing principles
  - Framework to be modified as needed for applications
- Two parts: RTP (data) and RTCP (control)

# RTP Data Header



# RTP Data Header Extension

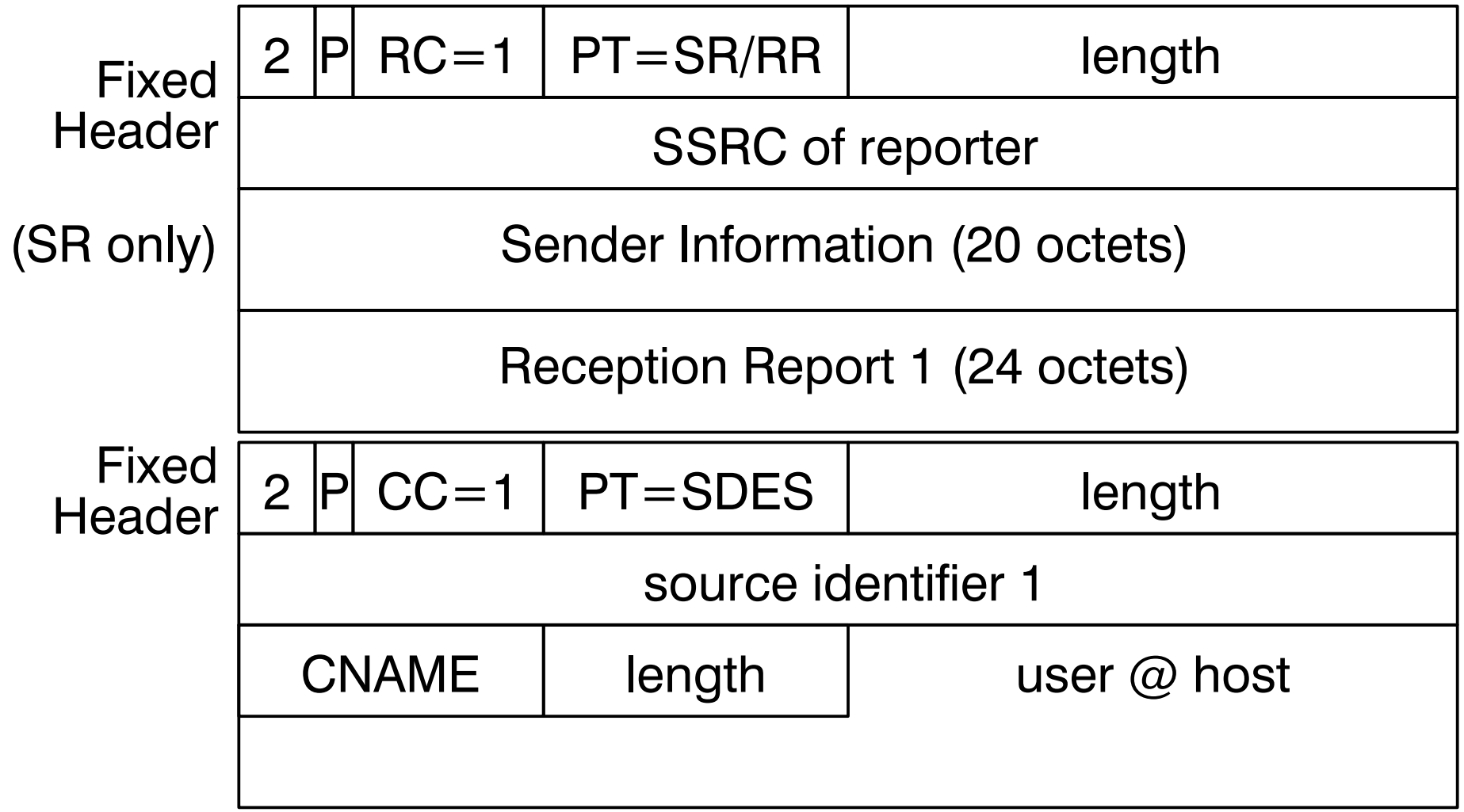



- We don’t expect this to be used!
- Most protocol extensions should be done another way
  - Profiles may define additional fields in fixed header
  - New payload types should define their own headers
- An escape hatch for experiments allowing interoperation with unmodified applications

# RTP Control Protocol (RTCP)

- Sent on separate “port” from data
- Multiple packet types “stackable” in one UDP packet
- Always includes sender/receiver report plus CNAME to provide meaningful binding among media tools, over time
- Additional source description or control information may be sent as needed by the application
- Overall control traffic bandwidth should be limited to a small fraction (1–5%) of the data traffic through the calculation of an appropriate reporting interval

# Typical RTCP Packet Format



# Sender Information

- Sent if data has been transmitted “recently”
  - NTP timestamp when report is sent (zero if unknown)
  - corresponding RTP media timestamp (calculated)
  - sender’s packet count (to calcluate packet rate)
  - sender’s byte count (to calculate data rate)

*(first two items together facilitate intermedia synchronization)*

## Reception Reports

- Reception info from all sources heard since last report, or round robin for as many as will fit in allowed BW:
  - source identifier of source being reported
  - cumulative number of packets received
  - cumulative number of packets expected (from seq.)
  - interarrival jitter (algorithm specified)
  - timestamp from last SR from this source
  - delay time since that SR was received

*(last two items for round-trip propagation delay calculation)*

## Changes Since July Draft (rtp-05)

- TBD's completed:
  - RTCP report interval calculation specified
  - Reception report "jitter" calculation specified
- Data and control ports specified to be even/odd pair
- Decided not to partition RTCP packet type space
- Proposed: renumber RTCP types for header validation
- All length fields are zero-based (including SDES items)
- New SDES items: PHONE, TOOL, PRIV (& renumbered)
- Length field prefixed to optional BYE reason string

# Interarrival Jitter Algorithm

Based on difference of packet spacing at receiver compared to sender:

$$D(i,j) = (R_j - R_i) - (S_j - S_i)$$

Or equivalently, the difference in relative transit times:

$$D(i,j) = (R_j - S_j) - (R_i - S_i)$$

Jitter is defined to be the mean deviation (smoothed absolute value) of this difference:

$$J = J + 1/16 (|D(i-1,i)| - J)$$

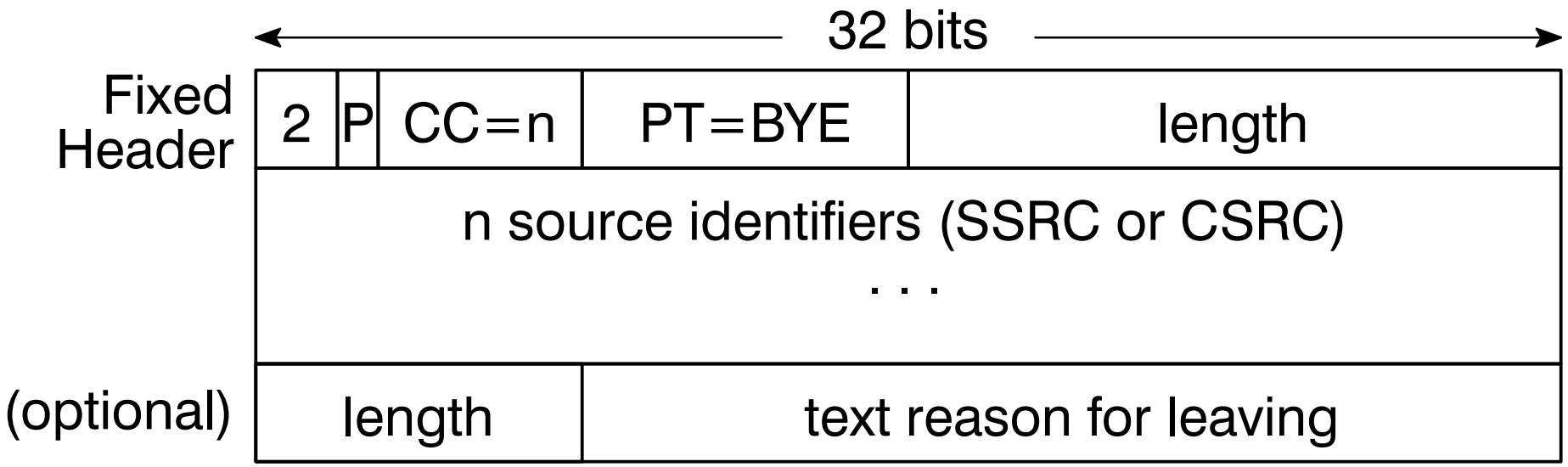
## Some Expanded Explanations

- The name “bridge” has been changed to “mixer”, translators allowed to change encoding
- Document types:
  - “profile” defines default payload types, any protocol extensions needed
  - “payload format” tells how to carry a media encoding in RTP, may be used with multiple profiles
- Use (or not) of data header extension mechanism

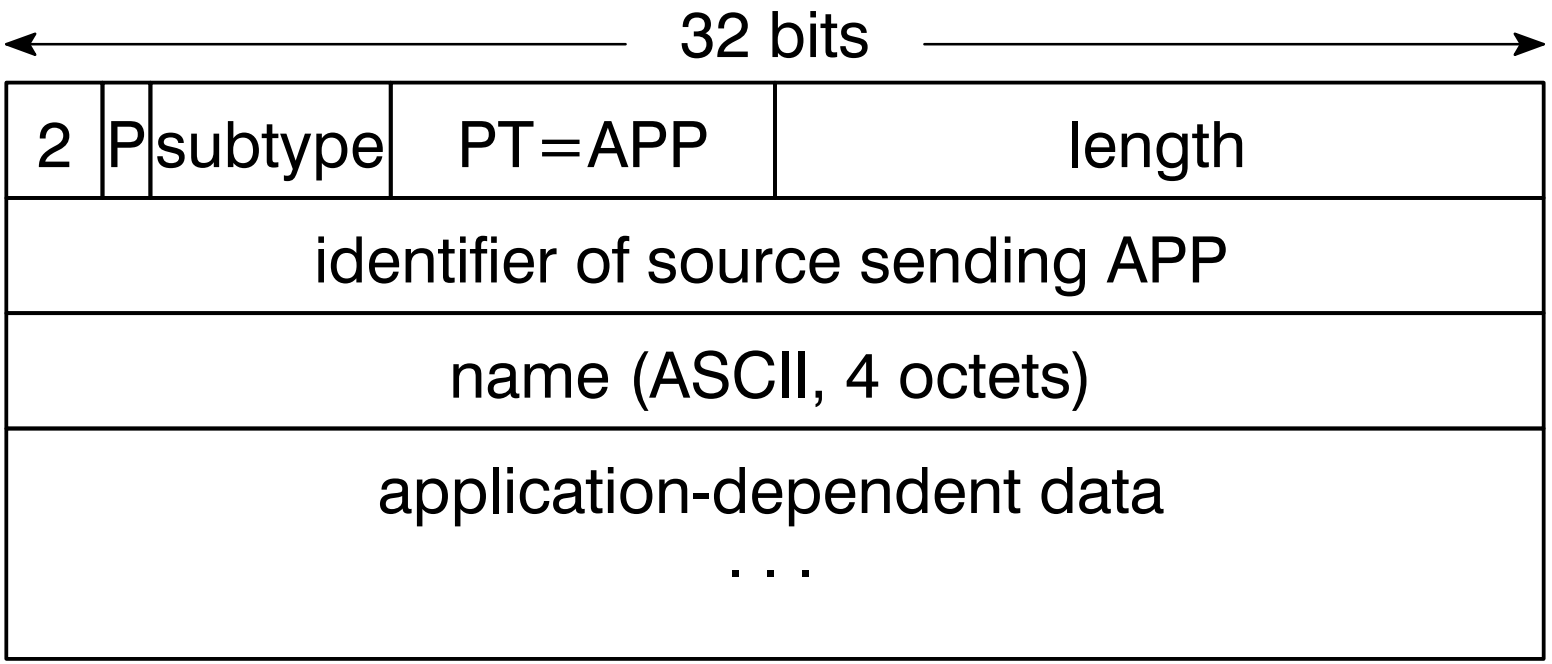
## Still To Be Completed...

- Need to specify algorithm for collision resolution and loop detection
- Need descriptions of algorithm for header validity check
- Finish editing more explanatory text
  - ALF philosophy: how to adapt RTP
  - Typical vs. worst case probability of collision
  - Motivation for use of RTCP reporting mechanism
  - Example use of auxilliary SDES items
  - Semantics for TXT: status notices, not quote of day
  - Mixer and translator functions

# Goodbye (BYE) packet format



# Application-Specific (APP) packet format



## Overview of Changes in RTP version 2

- Control and data on separate ports
- New control packet format; reception reports required
- No data options, CSRC count in fixed header
- Local 16-bit SSRC ID became 32-bit random global ID, always present (header now 12 octets)
- Media-specific timestamps (vs. fixed 65536 Hz rate)
- No Channel ID, use multiplexing in lower layers
- Application-specific sync marker bit
- Encryption covers whole packet; authentication omitted
- Translators can’t do unicast reverse path control packet
- BOS option eliminated, put info in encoding if needed
- Lengths changed to word counts, not counting length