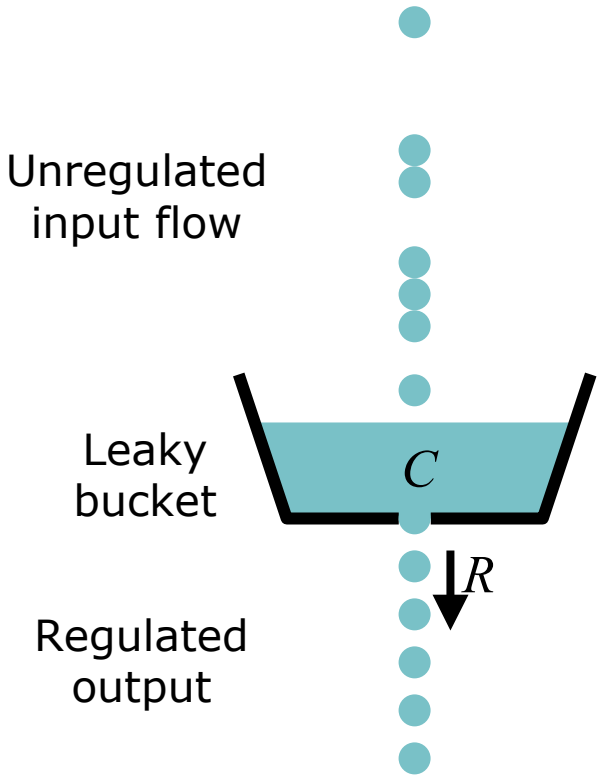
Real-Time Communication on IP Networks

Colin Perkins

http://csperkins.org/teaching/2003-2004/rtes4/lecture16.pdf

UNIVERSITY
of
GLASGOW

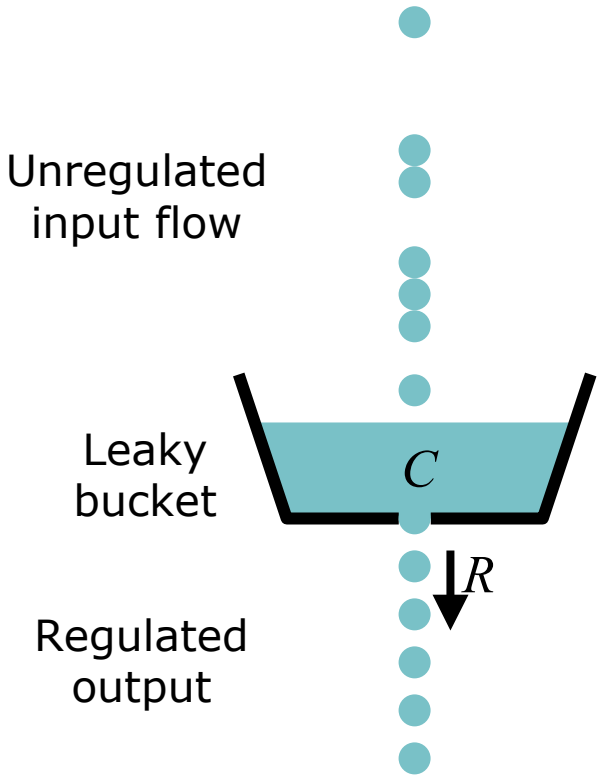
Recap on Leaky Bucket Filters



- Water flows into the bucket at a variable rate.
- The capacity of the bucket is C
- The bucket has a leak and, when non-empty, has constant outflow rate, R
- When the bucket overflows, some water is lost

- An (R, C) **leaky bucket filter** can be used to filter network packets in the same way

Recap on Leaky Bucket Filters



Properties of a leaky bucket filter:

- Output data rate will not exceed R, irrespective of variation in the input rate (may be < R, if input is idle)
- **Average** input rate cannot exceed R although short term bursts can
  - Maximum burst size, exceeding rate R, on the input is C packets
  - When the output is initially idle
- Maximum number of packets that can enter the network in any given time, t, is $Rt + C$

- Useful for turning sporadic flows into periodic flows, if average rate of the sporadic flow $\leq R$, and bursts never cause the bucket to overflow

Lecture Outline

- Have seen how uncontrolled packet networks can disrupt the timing of a network flow, and how enhanced services can be used to provide predictability in packet networks
- Despite their limited suitability, there is much interest in using IP networks to deliver real-time services, such as telephony and streaming video
- This lecture discusses how real-time networked multimedia services are supported on the Internet
  - Much is also applicable to other soft real-time services

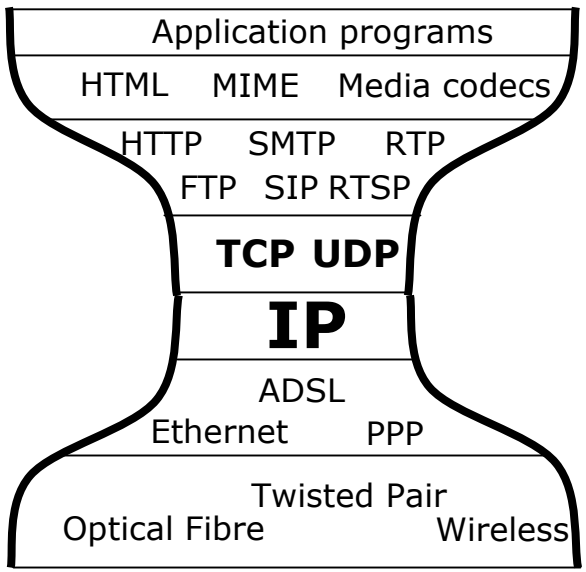
Contents:

- Timing properties of IP networks
- TCP/IP and UDP/IP
- Real-time Transport Protocol (RTP)

Networked Multimedia on the Internet

- Internet multimedia has long history:
  - First audio experiments in 1973
    - RFC 741, "Network Voice Protocol", 1977
    - http://www.ietf.org/rfc/rfc741.txt
  - First video experiments in the early 1980s
- Modern standards development began in 1992:
  - Developing from teleconferencing systems
  - Audiocast of IETF meetings
    - 20 sites on 3 continents
    - Precursors to RTP and the present standards
  - Standardized RTP in 1996
- Widespread availability of suitable networks in the last couple of years

The IP Protocol Stack



- IP provides an abstraction layer
  - Applications, transport protocols above
  - Assorted link technologies below
- Applications can't see the link layers
  - Just see IP layer performance
  - The IP routers can provide enhanced packet delivery service, but often don’t
  - Assume lowest common denominator behaviour, unless you control the entire system
- Link layer can't tell the needs of the application
  - Just see a series of packets
  - Optimisations for particular traffic classes are risky (e.g. 802.11 rexmit)
  - Is the traffic really what you think?
- Real-time on IP is about decoupling applications from the network

The IP Protocol Stack

- Performance not guaranteed
- Packets can be…
  - lost
  - delayed
  - reordered
  - duplicated
  - corrupted

  …and the transport protocol must compensate

- Many causes of problems:
  - Congestion may cause loss and queuing delays
  - Packet corruption may cause loss
  - Route changeover may cause loss and change the path latency
  - Propagation and queuing delay
  - Multi-path routing varies latency and may reorder
  - Link-layer striping may reorder
  - Spurious retransmission and router bugs cause duplicates

Assumption: significant packet loss, latency and jitter can be observed on a best effort IP network (remember IP performance graphs from lecture 14)

Real-Time on IP

- Performance *can* be bad
  - Applications should be prepared to compensate, isolating their timing behaviour and reliability from that of the network
- Packet loss, latency and jitter can be kept small through careful engineering and over-provisioning
  - Most backbone networks have very good performance
    - Essentially no loss
    - Very little queuing delay
  - Interconnects and customer LANs are currently the main trouble spots
  - Enhanced service networks can be used, if necessary
- Good enough for soft real-time, in many cases

Transport Protocols

- The IP service, by itself, is very limited
  - Just (tries to) deliver packets
- Always augmented by a transport protocol
  - UDP/IP
  - TCP/IP
  - (others in development)

UDP/IP

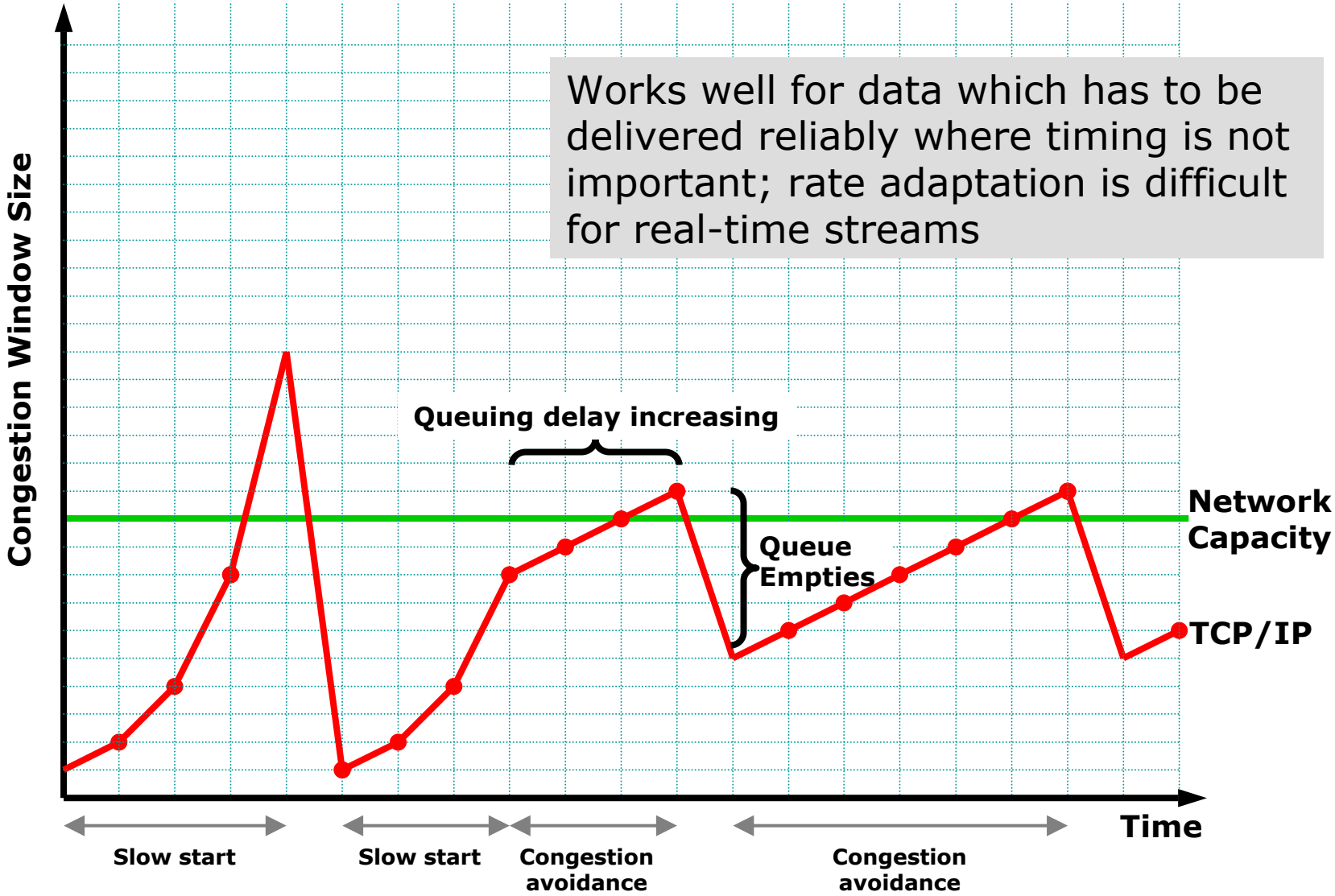
- Exposes the IP datagram service to applications
  - Best effort (unreliable) packet delivery
  - Connectionless
  - Unicast and multicast

- Can have all the problems we discussed in lecture 14:
  - Packet loss
  - Variable throughput
  - Jitter
- Uncontrolled timing, unless running on an enhanced service network, but no worse than the timing of IP

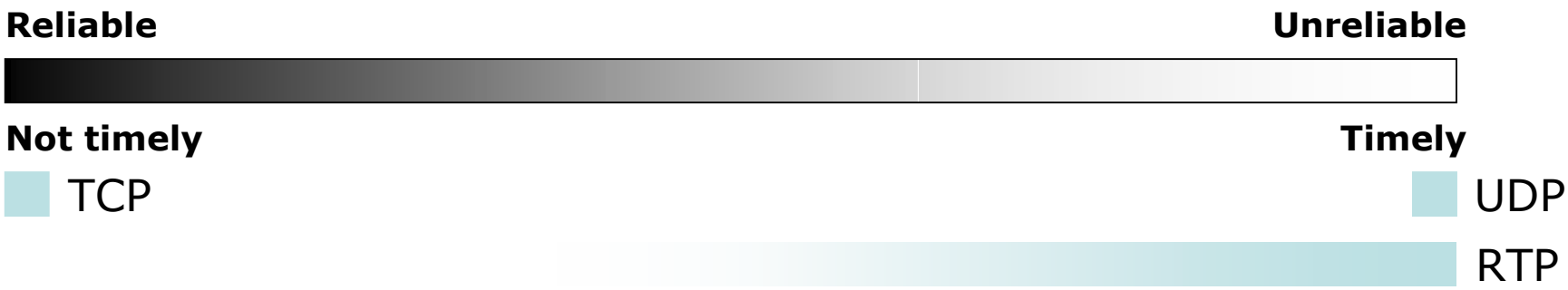
TCP/IP

- Connection oriented, reliable and rate adaptive protocol built above IP
  - Each packet contains a sequence number
  - Acknowledgements sent as packets arrive
  - Sender retransmits any lost packets
  - Receiver buffers data until all preceding packets have arrived, and presents to the application in order
- Adapts transmission rate to match network capacity
  - High link utilization
  - Approximately fair share between flows
    - No prioritisation
- Combination of retransmission and rate adaptation result in significant timing variation
  - Affected by network dynamics, not controlled by application
  - Largely unusable by real-time traffic

TCP/IP Rate Adaptation



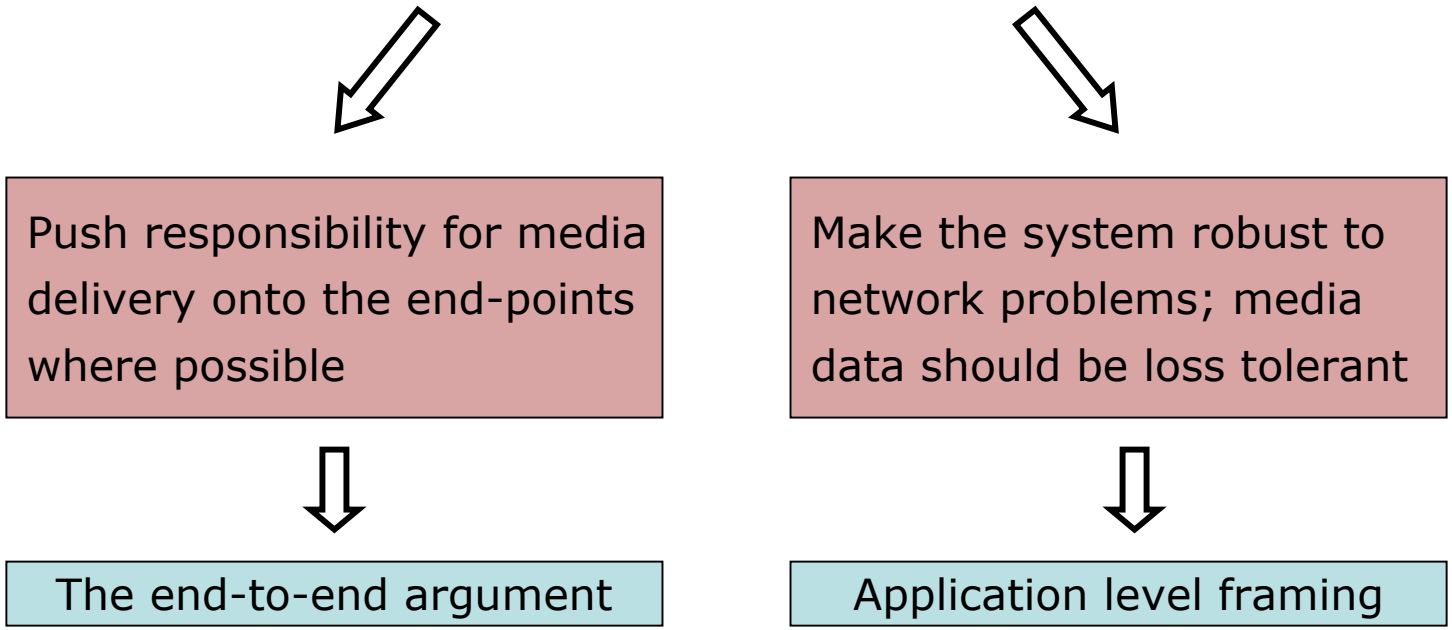
Reliability/Timeliness Trade-off



- Protocols built on uncontrolled packet networks must make a fundamental trade-off:
  - Unreliable, accepting (mostly) timely behaviour of the network
  - Reliable, accepting that error correction will worsen the timing
- TCP is at one extreme, UDP the other
  - Application level protocols can blur the boundary
- Real-time systems choose their transport carefully:
  - TCP for control
  - UDP for data, aided by the application

Real-Time on UDP/IP Networks

- The challenge:
  - Build a mechanism for robust, real-time media delivery above an unreliable and unpredictable transport layer
  - Without changing the transport layer
    - If you can change the transport layer, would just use an enhanced service network, and avoid these problems



The End-to-End Argument

- Two options for ensuring reliability
  - Pass responsibility hop-by-hop, along with the data
    - Email
  - Responsibility remains with the end points, which ensure delivery even if the intermediate steps are unreliable
- Most Internet protocols take the second approach

- Consequences:
  - Intelligence tends to "bubble-up" the protocol stack to the end points
  - The intermediate systems can be simple, and need not be robust
    - They can simply discard data they cannot deliver, since it will be recovered end-to-end

- The network is dumb, but end-points are smart

Application Level Framing

- Only the application has sufficient knowledge of its data to make an informed decision about how that data should be transported
- Implications:
  - The transport protocol should accept data in application meaningful chunks ("ADUs")
    - The application must understand the data,
    - The application must be able to process ADUs independently, in arbitrary order, and in the presence of loss
  - The transport protocol should expose details of delivery, allowing the applications to react intelligently if there are problems
    - The application can monitor delivery times, and adjust data use rates to match
    - Blind retransmission is not always appropriate
    - Maybe the data is stale, and an updated version can be sent
    - Maybe the data is obsolete, and doesn't need to be resent
    - Maybe an alternate representation of the data can be sent

Real-Time on IP Networks

- This philosophy implies smart, network-aware, applications that are capable of reacting to problems end-to-end.
  - Both sender and receiver are intelligent
  - The network is dumb and can be unreliable
- Use a network protocol designed to work with applications, and to expose timing and reliability of the network

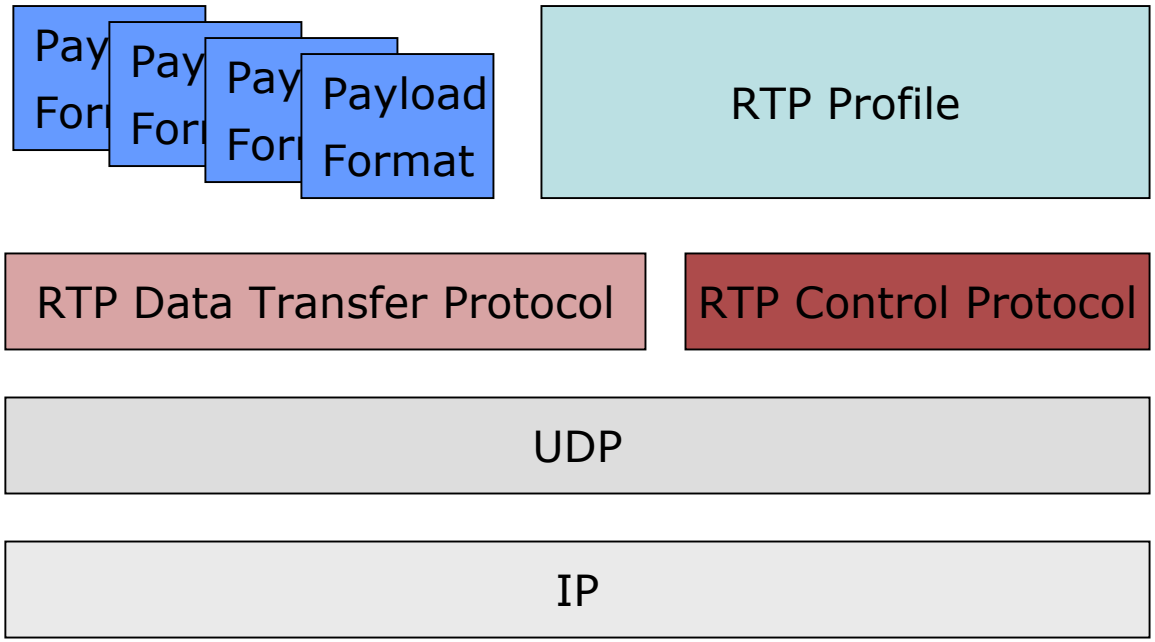
- Fits well with the IP service
- Contrast with traditional real-time networked applications:
  - Telephone network is smart, end-points are dumb
  - TV distribution: MPEG sender is smart, receiver relatively dumb

RTP: Real-time Transport Protocol

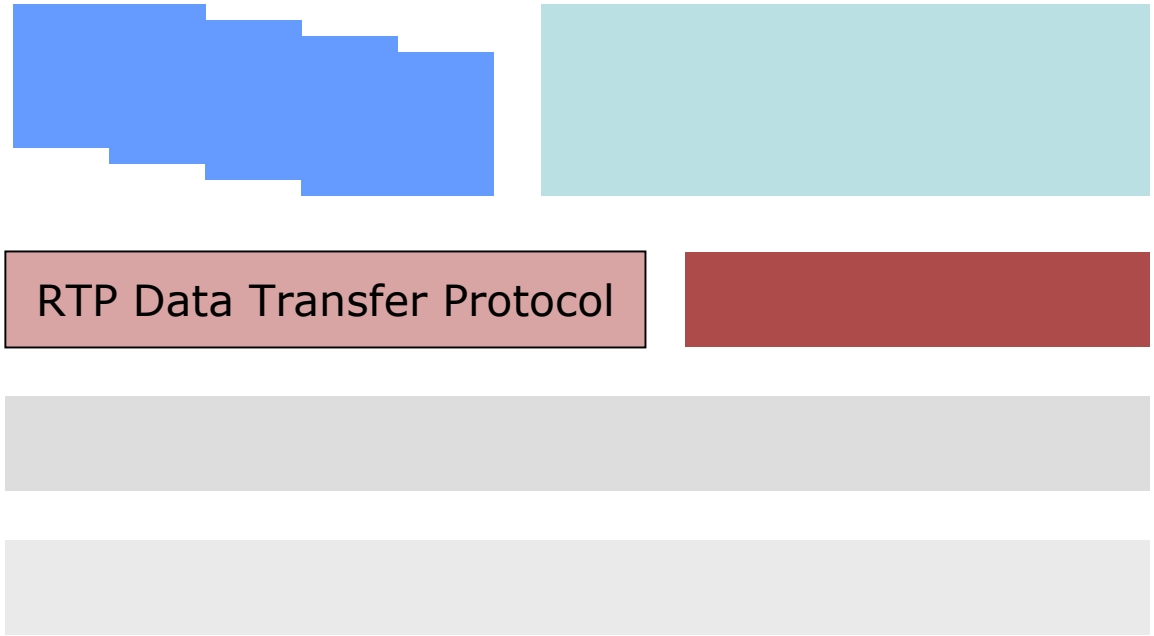
- The standard for real-time transport over IP networks
  - Streaming audio and video
  - Voice over IP
  - Sensor data

- Published as an IETF draft standard RFC
  - First version in 1996, updated in 2003 as RFCs 3550 and 3551
  - Adopted by ITU as part of H.323
  - Adopted by 3GPP for next generation cellular telephony
  - Widespread use in streaming: QuickTime, Real, Microsoft

Protocol Components

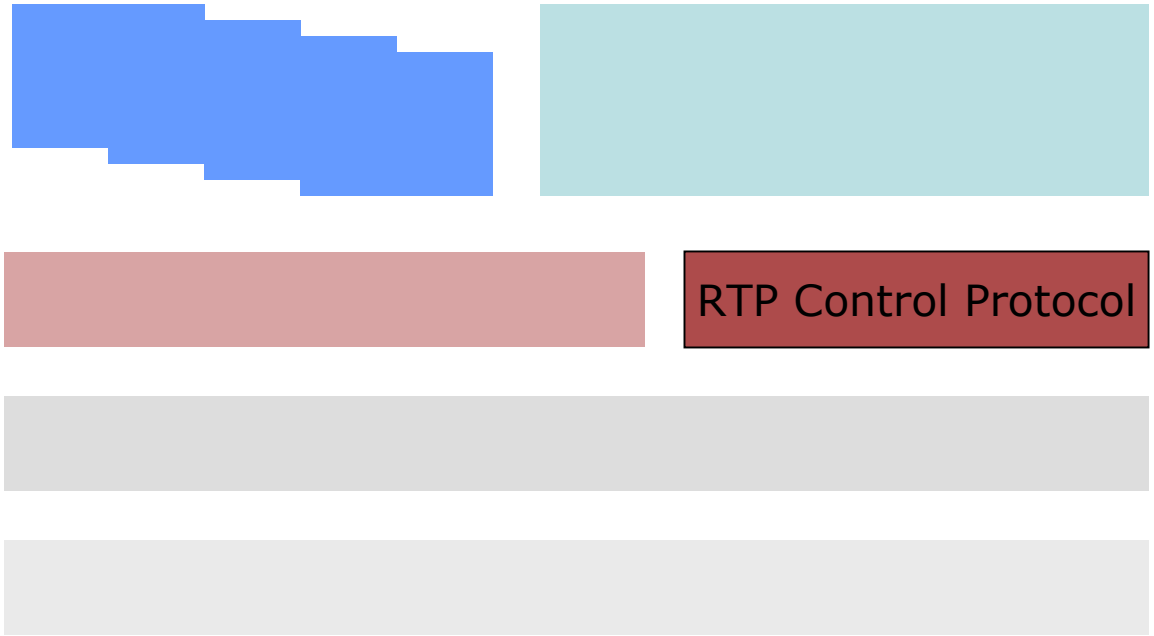


Protocol Components



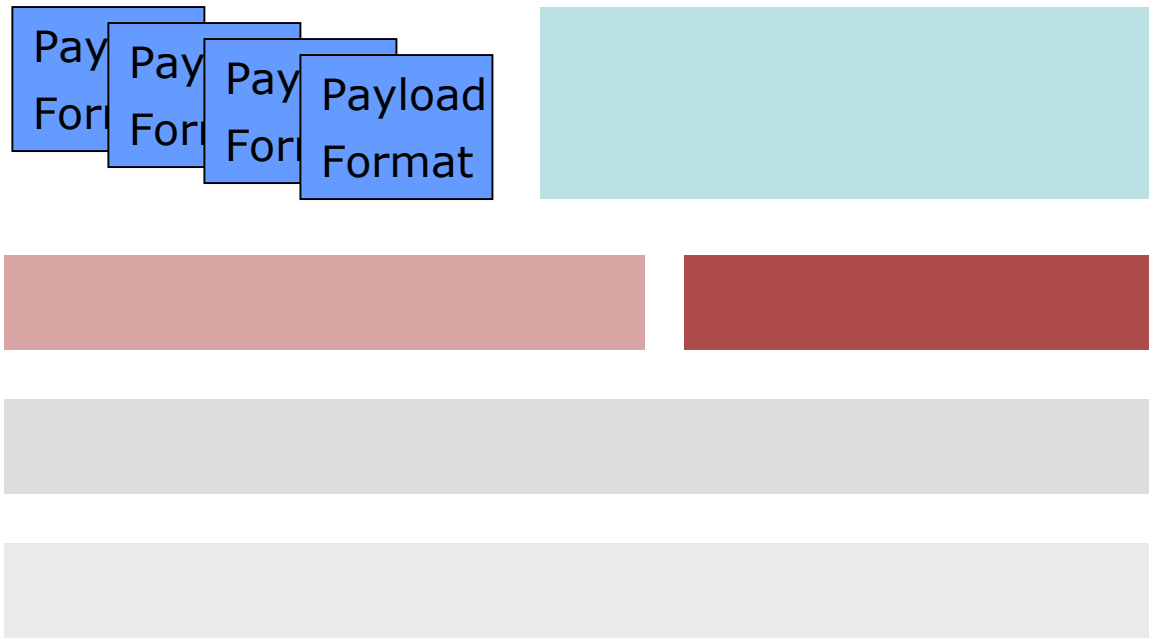
- Transports application data units
- Delivers a single real time data stream from sender to one, or more, receivers
  - Few assumptions about the underlying transport
  - Usually runs over UDP/IP
- Provides:
  - Source identification
  - Data format identification
  - Sequencing
  - Timing recovery
- Typically implemented in an application or as a library

Protocol Components



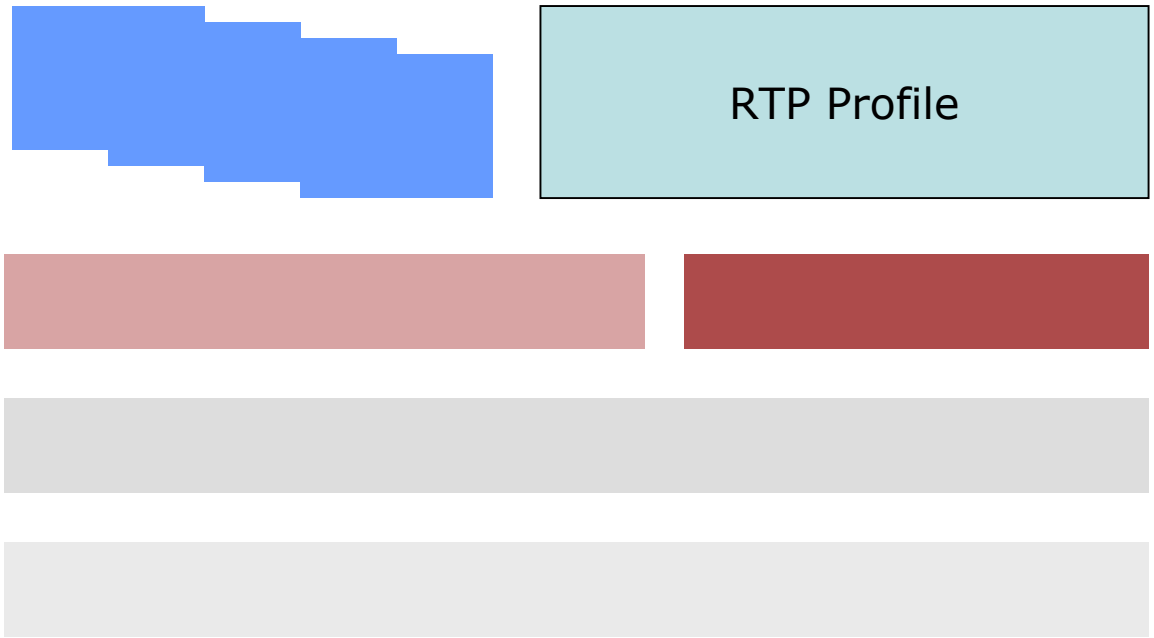
- Reception quality feedback
  - Packet loss fraction
  - Average timing jitter
- Optional source description
  - Name, location, email address, phone number
- Mapping from data clock to external time-base
  - E.g. for lip synchronization
- Loosely coupled membership management

Protocol Components



- Provide the adaptation layer between a particular data format and RTP
- Many payload formats exist, with more being developed

Protocol Components

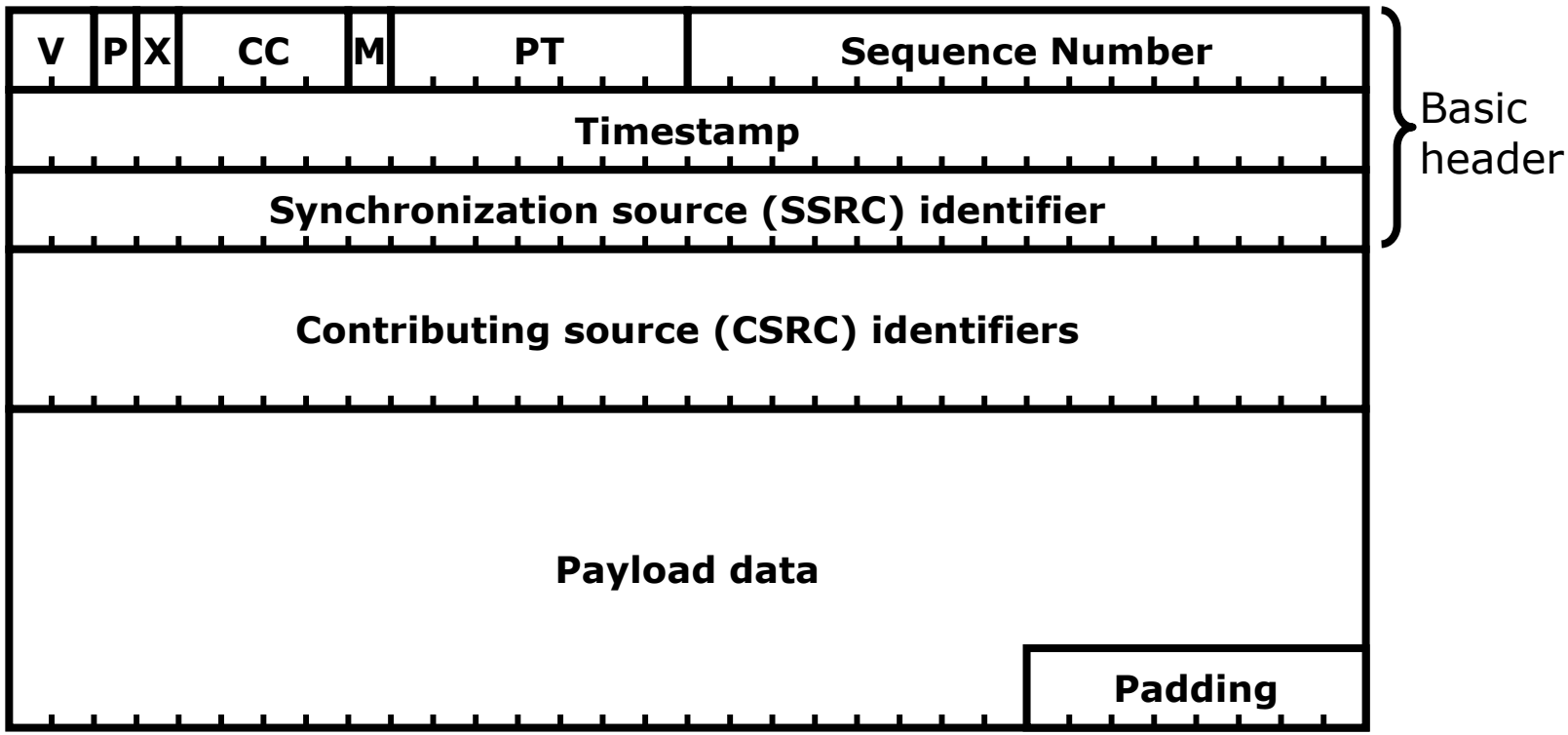


- Define use of RTP in particular application scenarios
  - "Reasonable defaults"
  - Adaptation to unusual conditions
    - Single source multicast
    - Operation without back channel
    - Authenticated and secure operation
- Provides a namespace for payload formats

Combining the Pieces

- Real-time data is transmitted by an application, within RTP
- Each RTP session:
  - Implements a particular RTP profile
    - E.g. profile for video conferencing; profile for sensor data
  - Includes an RTP data flow
    - Transporting a single data type according to one or more payload formats
      - E.g. Audio switching between G.729 and DTMF
      - E.g. Video using MPEG
  - Includes an RTP control protocol flow
    - Providing reception quality feedback, etc.
  - Is defined by:
    - Source and destination IP addresses
    - A pair of UDP ports: one for RTP, one for RTCP

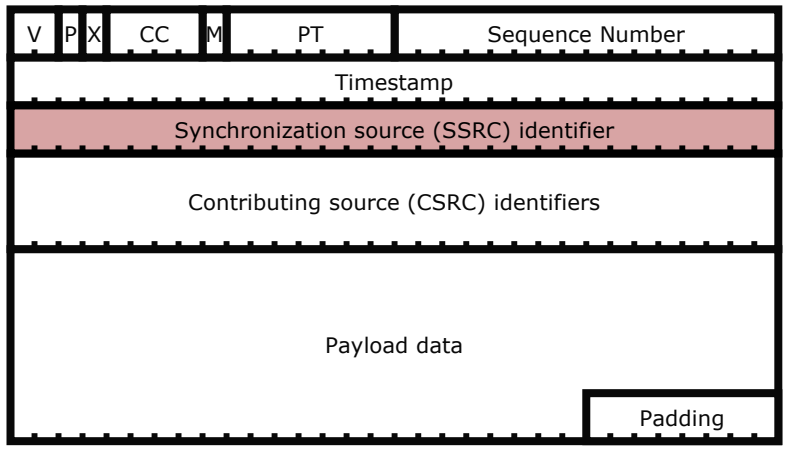
RTP Data Packet Format



- Usual header size is 12 bytes, extended in special cases
  - Header compression can be used to reduce this on low bandwidth links
- Usually sent on UDP port 5004, but may be dynamic

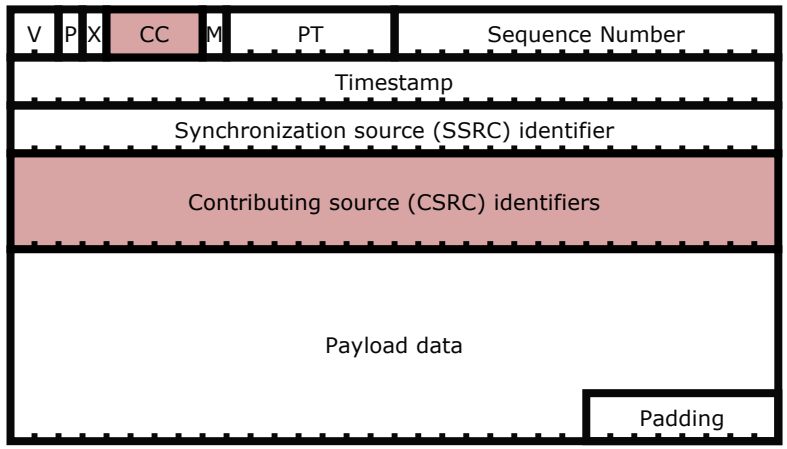
Source Identification

- Each packet carries a 32 bit synchronization source
  - Randomly chosen at start-up, with collision detection
- Provides a transport layer independent identifier
  - Supports gateways
  - IPv4, IPv6, ATM
- Identifies a single time-synchronized data flow
  - Mapped to a persistent identifier using RTCP



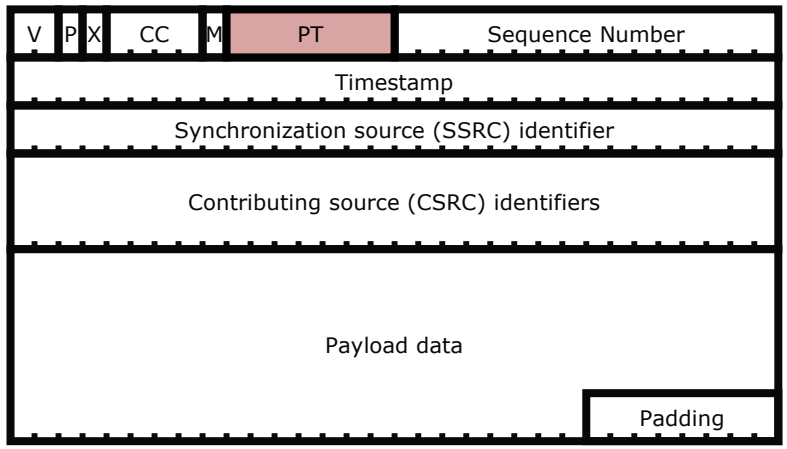
Source Identification

- Each packet may include a list of contributing sources
  - Usually empty, but allows data from up to 16 sources to be identified
- Allows RTP to support mixers and translators
  - Mixers combine several flows into one
  - Translators change the format of a flow, or gateway between different networks



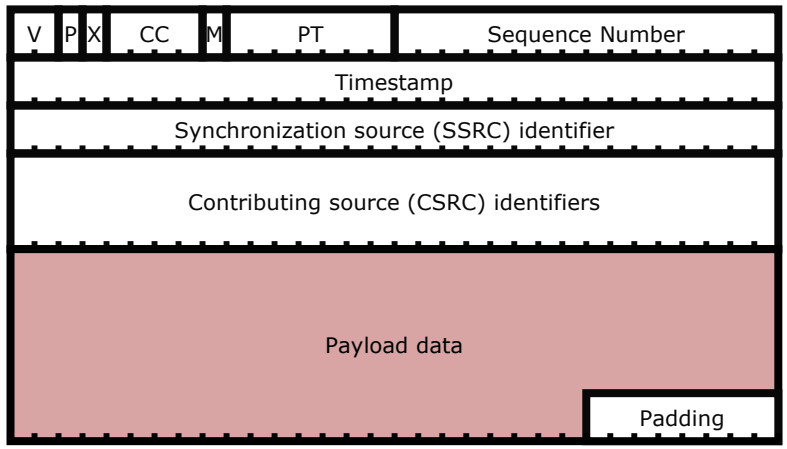
Data Format Identification

- Each packet carries a 7 bit payload type field
- Mapped to a **payload format** during session setup
  - Allows flexible signalling of data type and parameters
- Each flow carries only one type of data
- The payload type allows the sender to switch between a set of payload formats



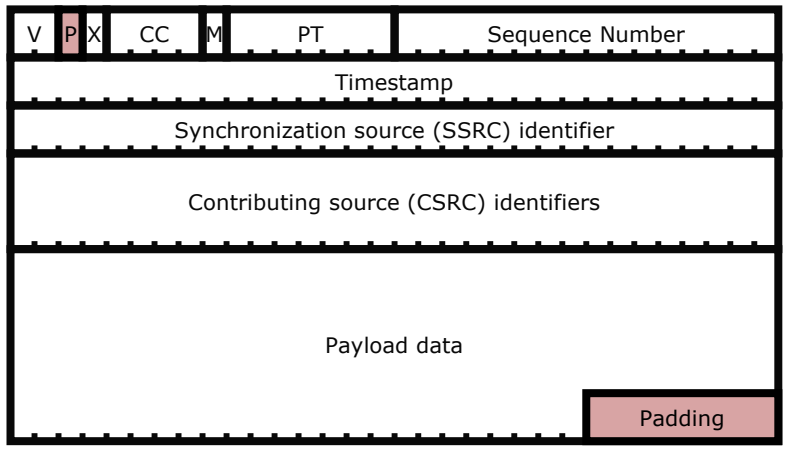
Media Transport and Payload Formats

- Packets contain a block of payload data, described by a payload format
  - E.g. G.729 generates 10 bytes of data every 10ms
- Payload formats describe the mapping between data format and RTP packets
  - Chosen so that each packet is independently decodable
  - Application level framing
- The data typically includes a **payload header** to ease parsing



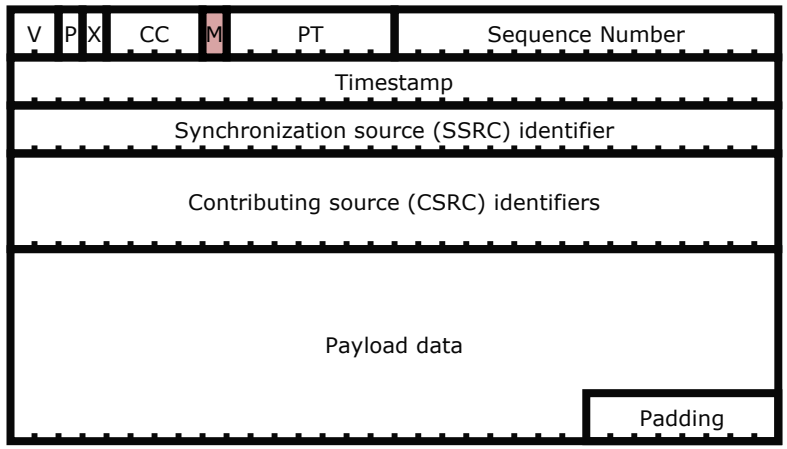
Media Transport: Padding

- Each packet may be padded beyond its natural size
- Rarely used, but needed by some encryption algorithms
  - DES in CBC modes operates on 64 bit blocks



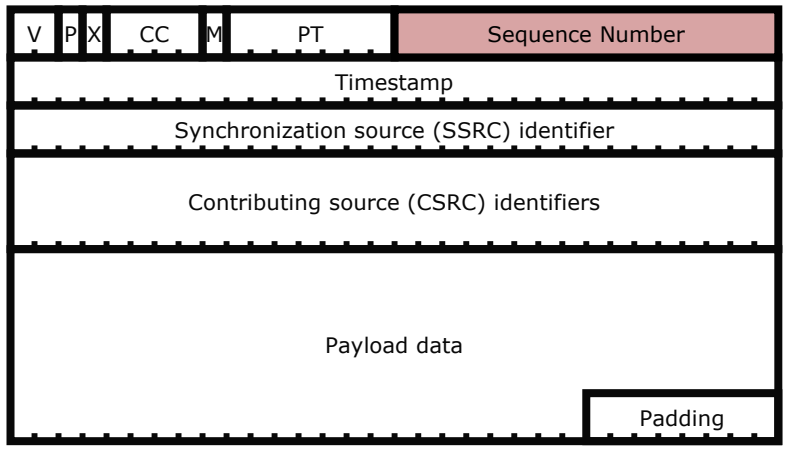
Media Transport: Marker

- Each packet includes a bit to mark significant events
- A hint that special processing may be required
  - e.g. last packet in a video frame



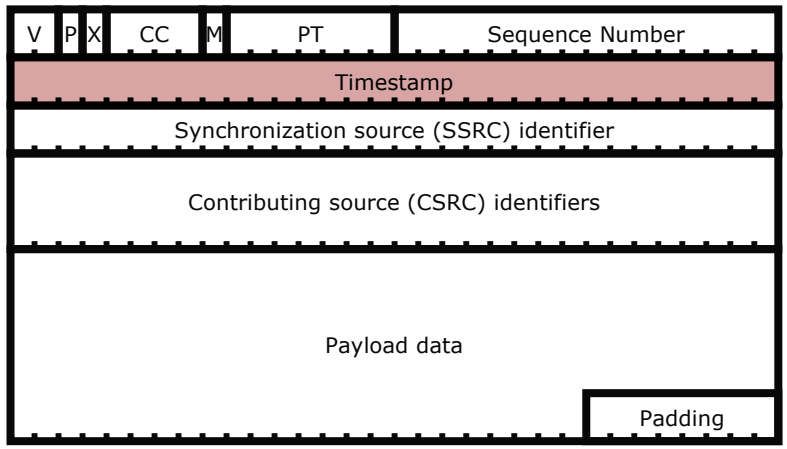
Sequencing

- Each packet contains a 16 bit sequence number
  - Random initial value
  - Increments monotonically with each packet sent
  - Wraps around to zero when the limit is reached
- Used to detect packet loss
  - Is *not* used to determine playout order
- Basic RTP does not provide error correction
  - The receiver is expected to conceal the error, and to continue processing
  - Extensions provide forward error correction and limited retransmission



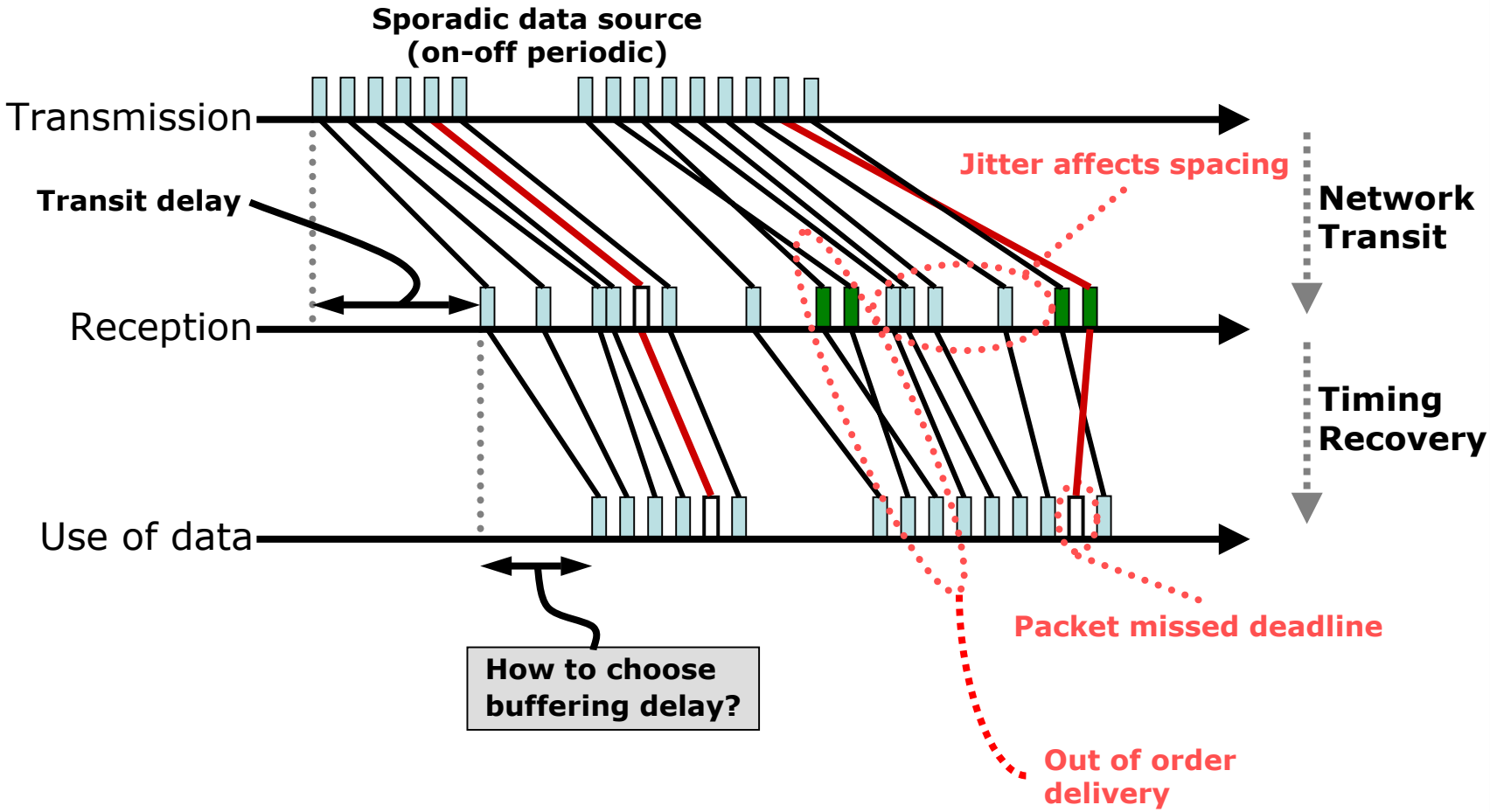
Timing Recovery

- Each packet contains a 32 bit timestamp
  - Indicates the sampling instant of the oldest payload data
  - Determines playout order
- The clock rate is defined by the payload format:
  - Audio clock is sampling rate
  - Video clock is 90kHz, indicating the frame time
- RTP places no requirement on stability or accuracy of clock



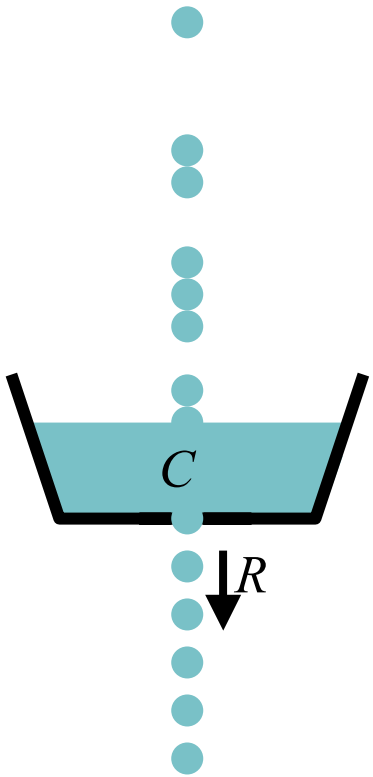
- Used by the receiver to reconstruct the timing of the data

Buffering for Timing Recovery



Buffering for Timing Recovery

- Consider a periodic flow (e.g. packet voice)
  - Timing has been bound by a leaky bucket filter at the sender
  - Then disrupted by the network
  - Data consumed at a fixed rate
- Model the buffering as another leaky bucket, with a plug

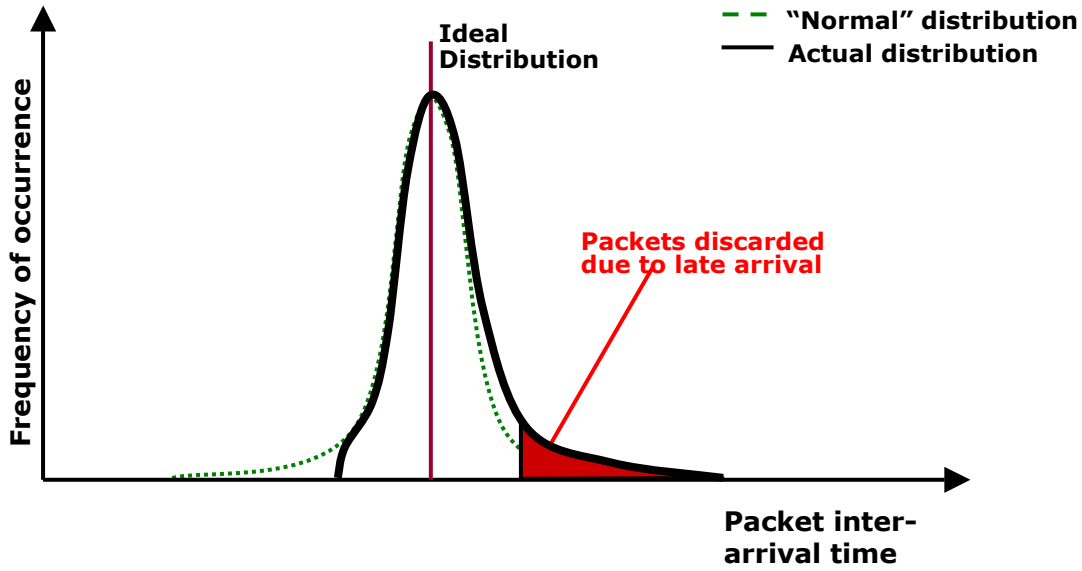


- Data begins to arrive, and accumulates in the timing recovery buffer
- After some buffering delay, data begins to be consumed
- The buffering delay is chosen based on the statistics of packet arrival times, so that the bucket rarely becomes empty while keeping within any latency bound
  - Empty bucket $\Rightarrow$ missed deadline

How Much Buffering Delay?

- Depends on jitter statistics
- Assume a normal distribution, and calculate standard deviation σ of inter-arrival times

$\Rightarrow$ 99.7% within 3σ of the mean



- Buffer for 3 times the standard deviation of the inter-arrival times and hope this missing ~0.3% of deadline is acceptable

Is a normal distribution a valid assumption?

Absolutely not…

…but close enough for many soft real-time applications

(engineering rule of thumb: assume, approximate, test)

…don't even think of doing hard real time on the Internet!

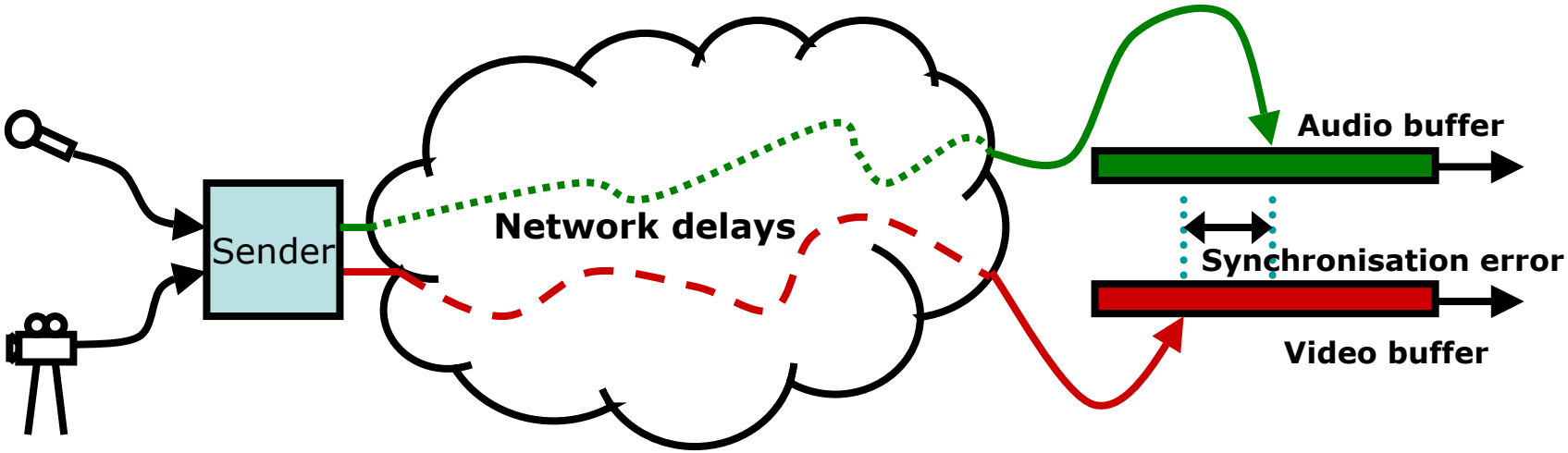
Timing Recovery

- RTP does not specify standard buffering and timing recovery algorithms
  - The necessary information is provided
  - Implementations choose how to recovery timing, based on their needed accuracy
- Many trade-offs to consider:
  - latency versus quality
  - speed of reaction to change
  - buffering ability
- Typical design:
  - Streaming applications use large delay (10+ seconds)
  - Interactive applications try to keep delay low (tens of milliseconds)

RTP Control Protocol (RTCP)

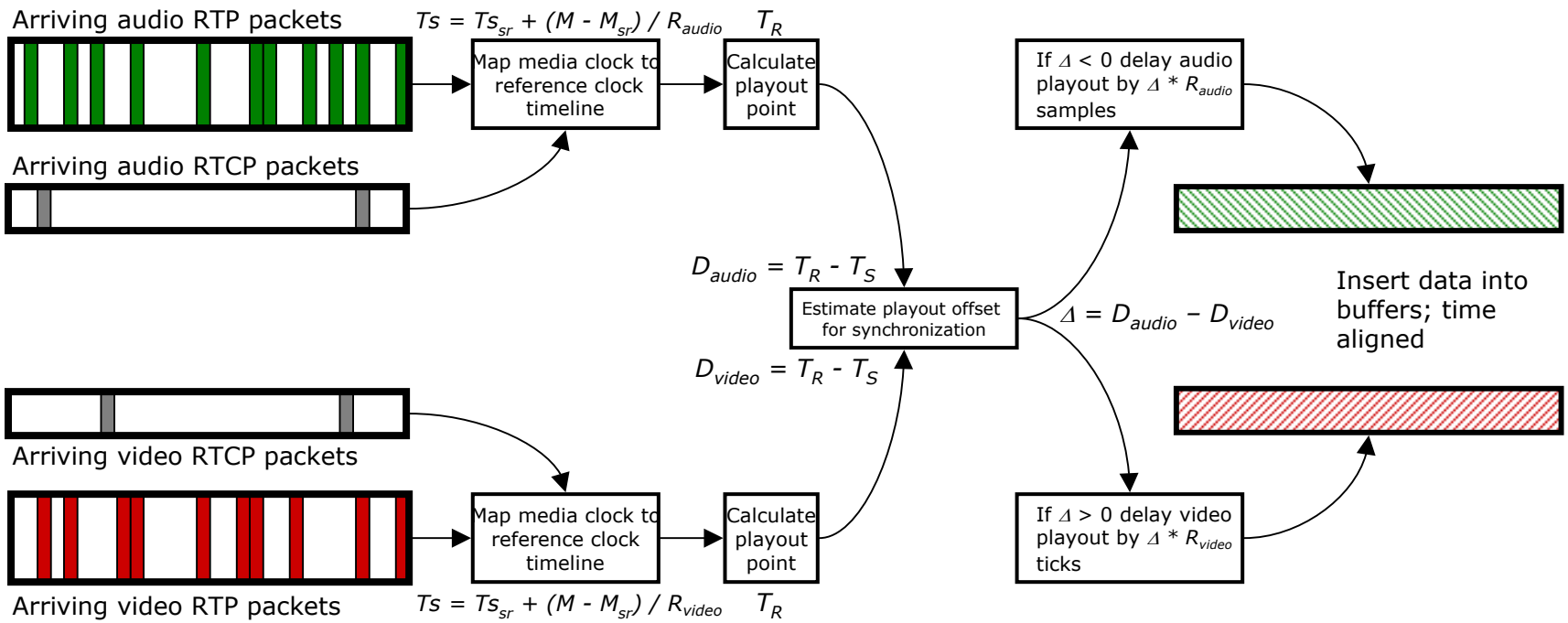
- Each RTP data flow has an associated control flow
- The control flow provides:
  - Time-base management
  - Quality of service feedback
  - Member identification and management

Synchronization and Time Management



- RTCP packets contain timestamps to map between the RTP timeline and NTP "wall-clock" time
  - Provides the information needed for a receiver to synchronize data sent as different flows, with different clocks
- Also allows receivers to estimate data/packet rate and possibly clock skew

Synchronization and Time Management



- Use information in control packets to map data clocks to a common timeline
- Estimate offset and skew between clocks
- Delay use of one set of data to align with the other set

Reception Quality Reporting

- Quality of service feedback from each receiver:
  - Loss fraction
  - Cumulative number of packets lost
  - Highest sequence number received
  - Inter-arrival jitter
  - Round-trip time
- Many uses:
  - Loss rate can be used to select amount of FEC to employ
  - Jitter gives estimate of playout buffer delay at receiver

Membership Management

- RTCP provides a canonical name, mapping SSRC to a persistent identifier
  - Used to associate streams for synchronisation
- RTCP can optionally deliver source description data:
  - Name
  - Email address
  - Phone number
  - Location
  - (extend with metadata)
- Provides loosely coupled presence information
  - Explicit leave message
- Augments the membership management provided by the signalling protocol
  - Primarily using the explicit leave indication

RTCP Reporting Interval

- RTCP is a low-rate reporting protocol
  - Not intended for uses that require instant feedback
  - Scalable to very large sessions
    - Statistical summary of group conditions
- Packets are sent periodically
  - The interval between packets is adjusted to limit RTCP to once per 5 seconds, or 5% of the data rate
  - Randomized ±50% to avoid synchronization

Summary

RTP provides:

- Flexible and extensible real time data transfer protocol
  - Supports a range of data type
  - Allows detection of network problems
  - Allows recovery of media timing
- Associated, low rate, reporting of reception quality, time-base, and presence information

- The building blocks to let **soft real-time** applications adapt to the vagaries of an IP network

Summary

By now, you should know…

- Timing properties of IP networks
- Use of TCP/IP and UDP/IP for real-time traffic
- Overview of RTP
- Understanding that real-time on IP networks is limited to soft real-time, with flexible applications

Next Lectures:

- No lectures next week – slots to work on programming assignment
- Q&A session on 9th March
- Next lectures on 10th and 11th March