MANCHESTER 1824
The University of Manchester

COMP28411 Computer Networks

Nick Filer

Multimedia – 3

Some material from:

Kurose & Rose – Chapter 7 + Slides

Halsall – Multimedia Communications

MANCHESTER 1824
The University of Manchester

Overview

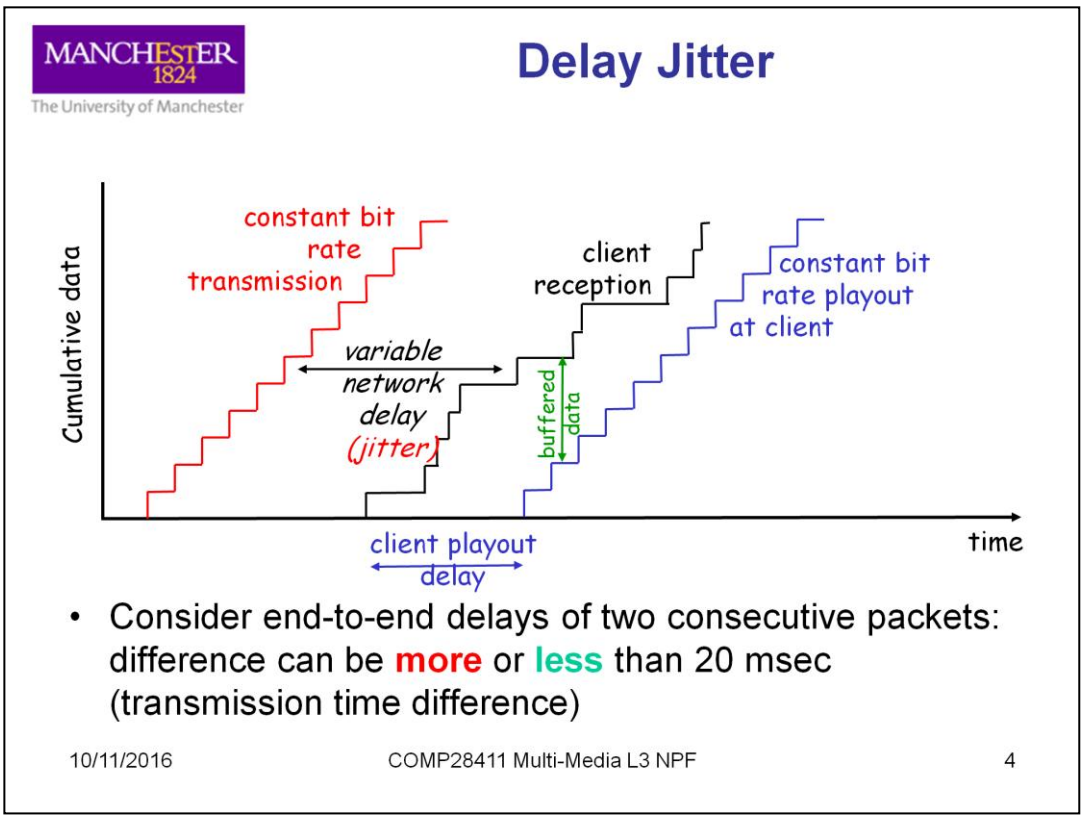
- Setting up Voice over IP (VoIP) telephone calls and other media exchanges.
- RTP + RTCP

MANCHESTER 1824
The University of Manchester

Problem ????

REAL-TIME CONVERSATIONS?

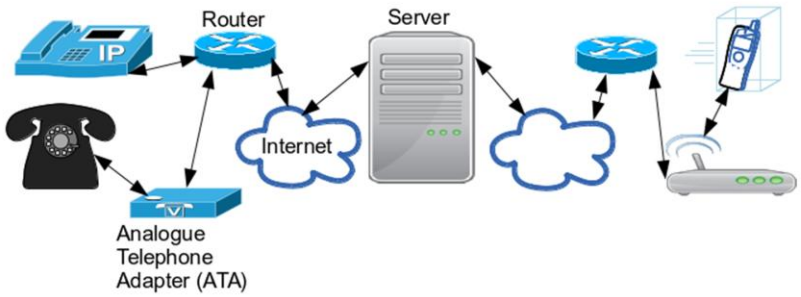
- Setup conference sessions
  - Session Initialization Protocol (SIP)
- Real Time Protocol (RTP) &
  - Real Time Control Protocol (RTCP)
- Making TCP work for streaming and real-time



You have seen this type of display of packet delays for a stream of packets before. Here we show how a constant packet transmission rate at a source typical of applications such as sound and video results in varying delays at the receiver which we call jitter or more clearly “varying network delay”. When there is jitter and the destination wants all or most of the data to available for use in the correct order it is necessary to buffer and delay the use of arriving data at the destination to allow for the delay variation.

How do today's MM sessions work?

- SIP – Session Initiation Protocol
  - Inter-Asterisk-eXchange (IAX) or now IAX2 competes with SIP. IAX2 only uses 1 port, less bandwidth for control, binary not text. But it is not yet fully standard.
  - Application layer , not a service provider, other protocols provide services e.g. Real Time Protocol (RTP) to carry MM traffic.
  - Uses a client server model.

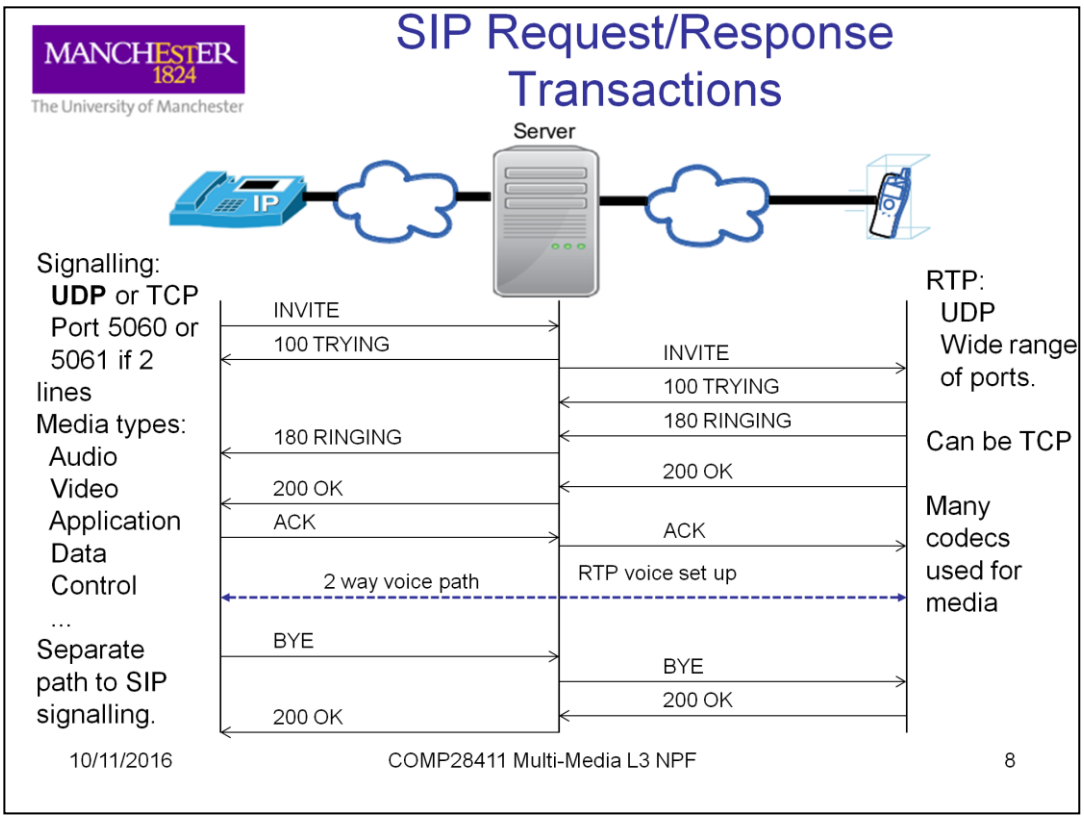


10/11/2016 COMP28411 Multi-Media L3 NPF 7

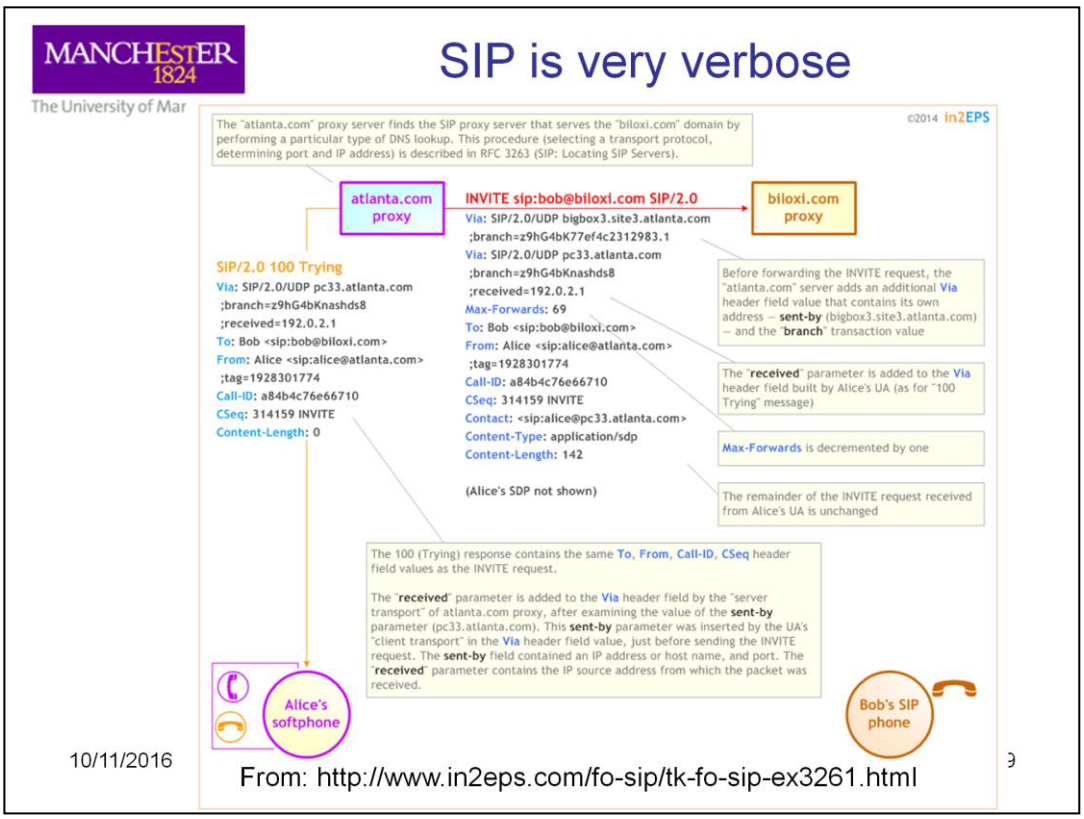
SIP can be used for one way streaming or for two way multi-media conversations. There can be 2 or more participants and there may be many separate streams of media carrying, for example voice and pictures together or separately between participants. SIP does not handle the actual media, just the configuration of the links.

SIP is a client server architecture. Each participant connects via a server and destinations are selected by SIP servers talking to one another. Hence SIP addresses normally explicitly state the name of the destination server and end up looking like email addresses. My SIP address could be nickfiler@some.sip.server.com or similar. There are quite a lot of different SIP providers.

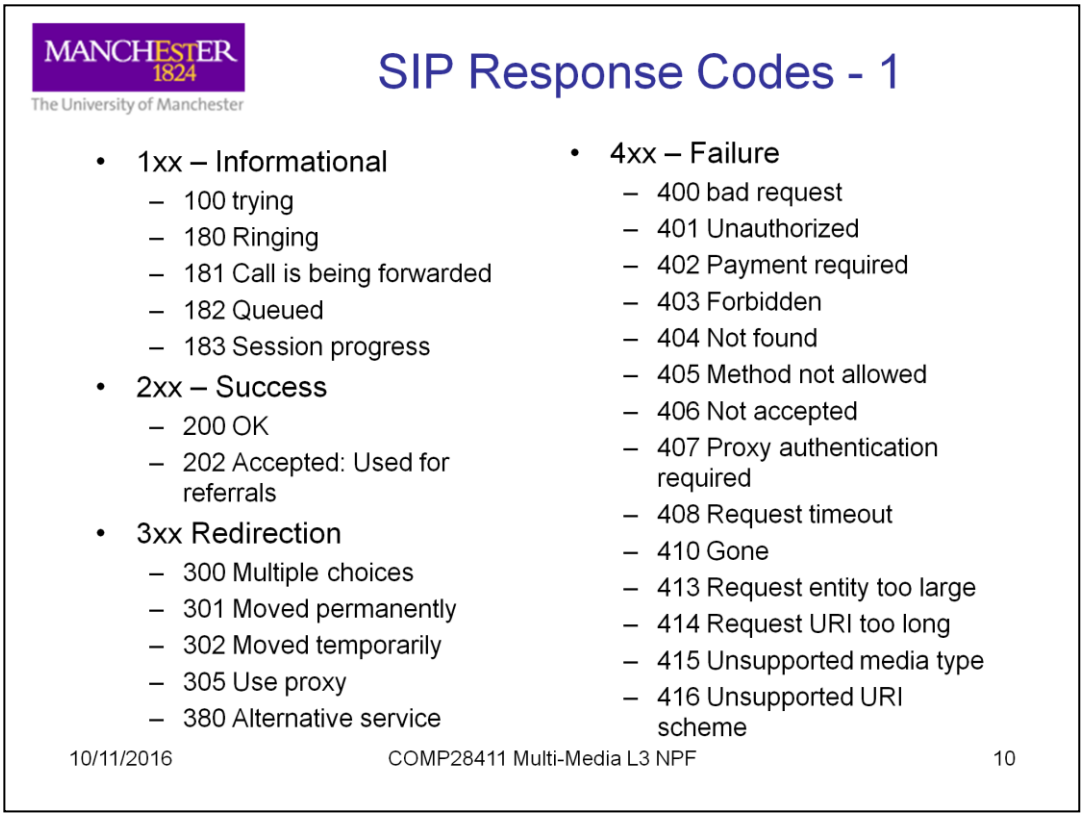
The main competition for SIP has been from IAX and IAX2 which are part of the open-source Asterisk (http://www.asterisk.org/) system. But even Asterisk now promotes SIP.



SIP runs in real-time by passing messages between devices. The slide shows a sequence that abstractly represent a telephone conversation over SIP. Of course, SIP can handle both telephone and VoIP and other media conversations. The messages are a mixture of commands and status responses. As you might expect there are many different status responses to handle all the different outcomes and problems the media exchange system can have.



The example shows Alice trying to call Bob's SIP phone. Note that what might be a simple message is expanded with lots of fields which are necessary to uniquely configure, maintain and charge for the call!



As I said there are lots and lots of SIP response codes to handle common interactions and problems in setting up conversations, maintaining them and later tearing them down.

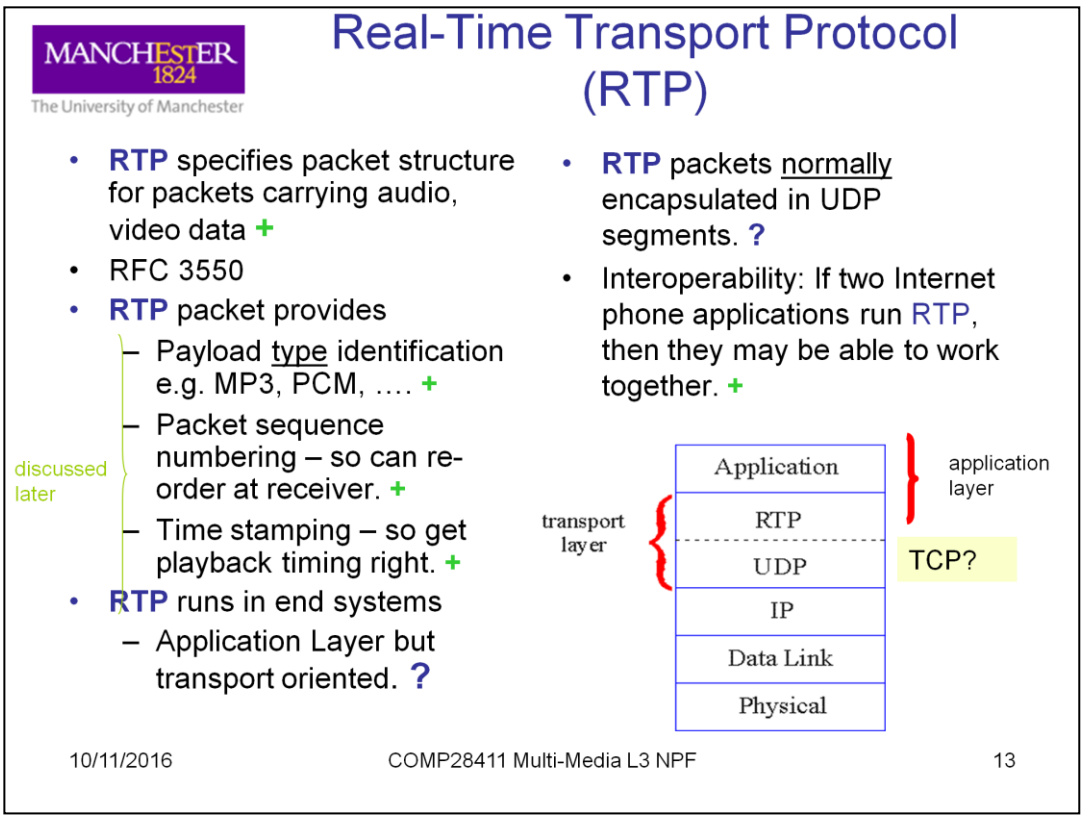
SIP Response Codes - 2

- 4xx – Failure (continued)
  - 420 Bad extension
  - 421 Extension required
  - 423 Interval too brief
  - 480 Temporarily unavailable
  - 481 Call/transaction does not exist
  - 482 Loop detected
  - 483 Too many hops
  - 484 Address incomplete
  - 485 Ambiguous
  - 486 Busy here
  - 487 Request terminated
  - 488 Not accepted here
  - 491 Request pending
  - 493 Undecipherable
- 5xx - Server errors
  - 500 Server internal error
  - 501 Not implemented
  - 502 Bad gateway
  - 503 Service unavailable
  - 504 Server timeout
  - 505 Version not supported
  - 513 Message too large
- 6xx Global failures
  - 600 Busy everywhere
  - 603 decline
  - 604 Does not exist anywhere
  - 606 Not acceptable

MANCHESTER 1824
The University of Manchester

Problem ????

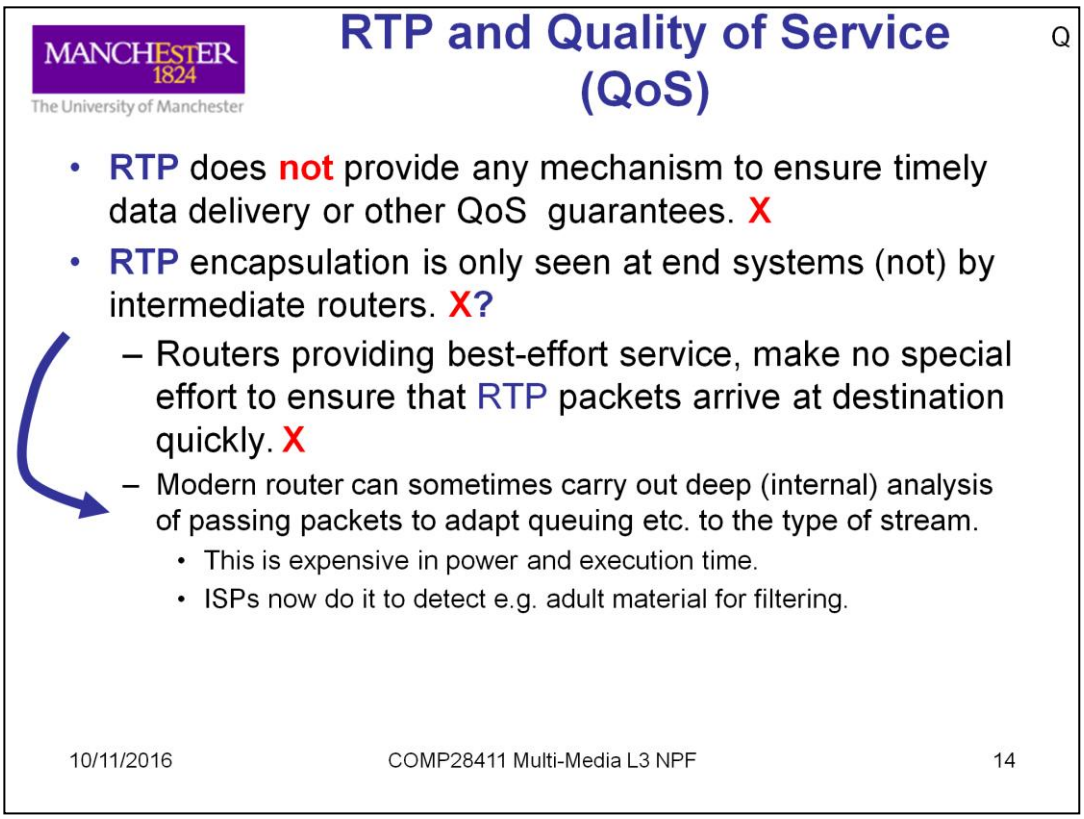
REAL-TIME PROTOCOL



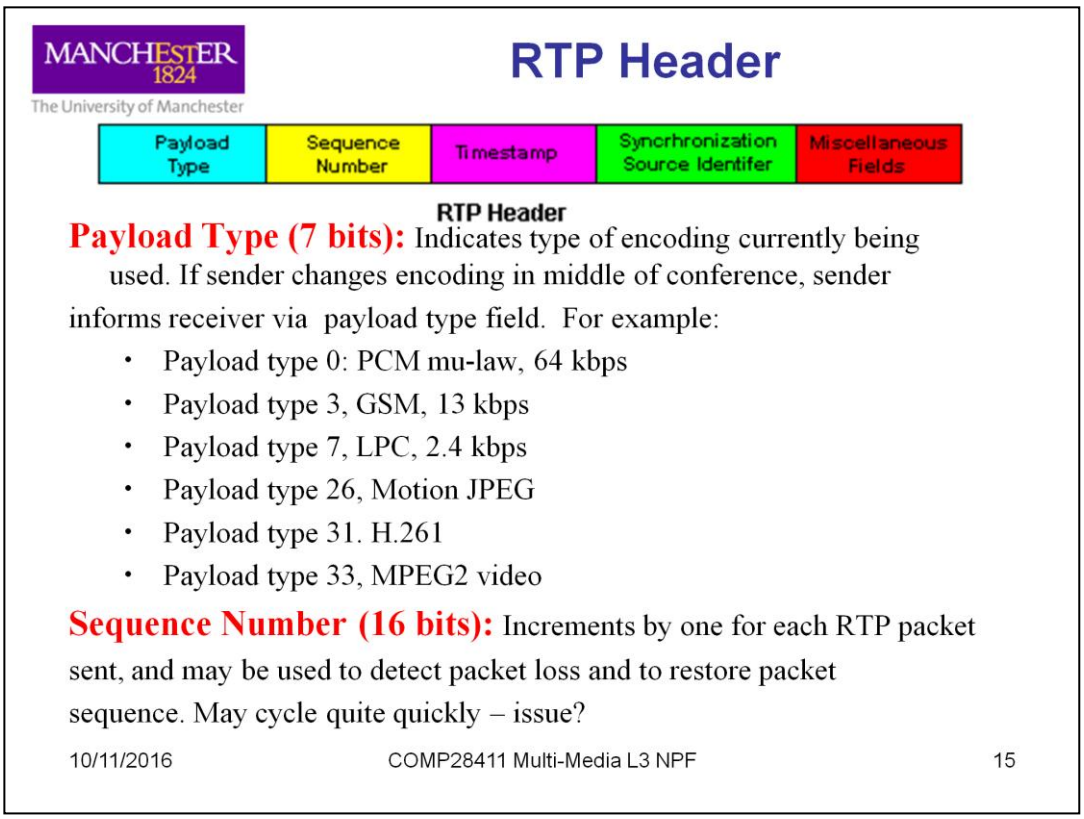
We have no doubt mentioned the RTP several times already. RTP is a protocol used to extend, in particular UDP and similar transport protocols to carry real-time media in a form whereby it can be, at the destination and provided it arrives on time, played back in the correct sequence with the correct inter packet timing so that a media stream sounds and looks OK. In addition, the protocol allows the exact format of the media data to be specified so that the destination can correctly interpret it, even if it changes from packet to packet.

Typically, destinations open a port for receiving incoming RTP packets. In a conference environment or where several applications are all generating RTP at the same time there can be multiple streams of media arriving and being processed at the same time. This requires a per media stream index as an addition field in each packet to indicate the source from where this packet stream arrived so that it can be processed correctly relative to the packets just before and after it from the same stream and from other streams.

For example, you might be in a video conference with several people. From each person's computer you receive an audio stream and a video stream separately. The packet sequence numbers and timestamps are not enough to play the data back correctly. Each stream needs to be processed separately then combined for playback.



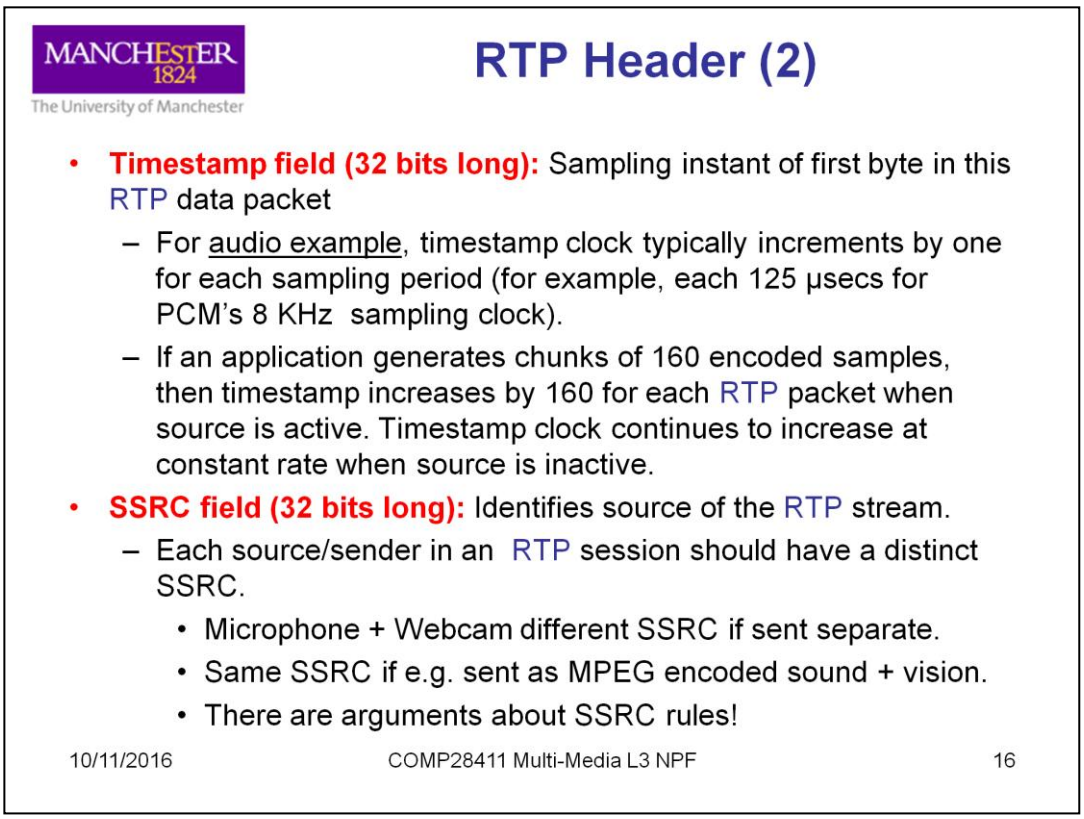
The Real Time Protocol (RTP) is much more limited in capability than you might at first expect. However, other protocols can be used alongside RTP to expand and control what it is able to do and be used for. Firstly RTP carries timestamps from place to place associated with media objects to which the timestamp applies but it does not do anything about making sure the media is delivered on time. Partly this is because RTP is an application layer protocol which used end-to-end transport such as UDP or TCP to move packets of data from source to destinations. In theory routers are unaware of RTP but modern routers using deep packet analysis may be able to differentiate RTP packets from other protocols and types. You have seen this type of filtering in the laboratory using Wireshark.



An RTP packet wraps around a block of media data which is then sent for transport across a network. The header part of the wrapper has various fields. The fields like many protocols have some pre-defined fixed values but in many cases lots of undefined values are left that users might grab and use for their own, in the case of the payload type, codecs and data formats.

The sequence number is 16 bits and is used as a packet counter with 64K values. It is used to ensure that packets arriving at a destination can be put back into the order they were sent if they arrive out of order. This is similar to the sequence number in TCP and other protocols. RTP could use TCP for transport but it can also be used with protocols such as the much simpler UDP that has no sequence number. In order play-back is very important to media so a field is used for this ordering information.

The small size of the sequence counter is possibly a problem. If the jitter on a network is large and the network is transferring very large numbers of RTP packets in a very short time then the cycle time for sequence numbers may be too fast such that two packets with the same sequence number are in the network at the same time. So far, this has not been a major issue. Because, in theory a UDP packet can carry 64K of data in one chunk the sequence number problem could be solved by using larger packets. However, because the ubiquitous Ethernets allow only small MTU values (see TCP notes) such large packets may end up being fragmented. Hence most media packets are relatively small and in some cases very small which is an efficiency issue because lots of packets each carrying their header and tail data and tiny amounts of actual data are much less efficient resource users than fewer large packets such as typical TCP ones fitted to the common Ethernet MTU of 1500 bytes/Octets.



The timestamp is recorded at the source as the first sample (magnitude of sound, frame of video ….) sent in the packet is collected by the source device. Using the audio G711 CODEC 8,000 samples per second are taken at 125µsecs intervals. Note a µsec is 1 millionth of a second or 10^{-6}. The timestamp is another counter that increments in sample time units whether data is being collected or not. Therefore, it accurately reflects the separation in time between samples in adjacent or distantly separated (in time) packets. This separation in time allows the play-back to align the samples exactly the same distance apart in time for play back as they were when originally captured.

Because RTP sessions may have many different streams of media the final field discussed is an identifier for these streams. For example, imagine a studio recording with 32 different tracks or musical instruments to be merged into a final audio recording for sale to the public, each track may have an individual identifier to allow the destination to accurately reconstruct the original data. Destinations may receive data from many locations and with many sources and in different formats all at the same time. The SSRC and the Payload Type allow these various sources to be properly processed by the destination.

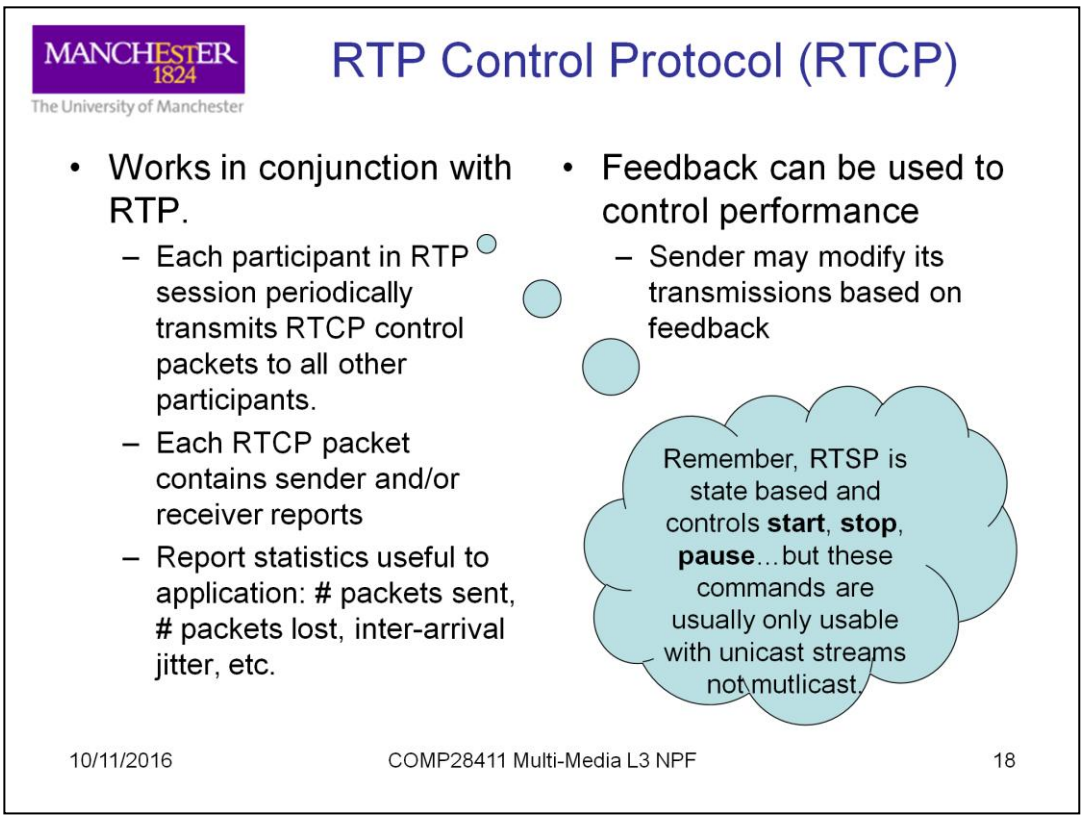
.

SSRC – Synchronization Source – A channel per source or grouped source (MPEG).

MANCHESTER 1824
The University of Manchester

Problem ????

REAL-TIME CONTROL PROTOCOL

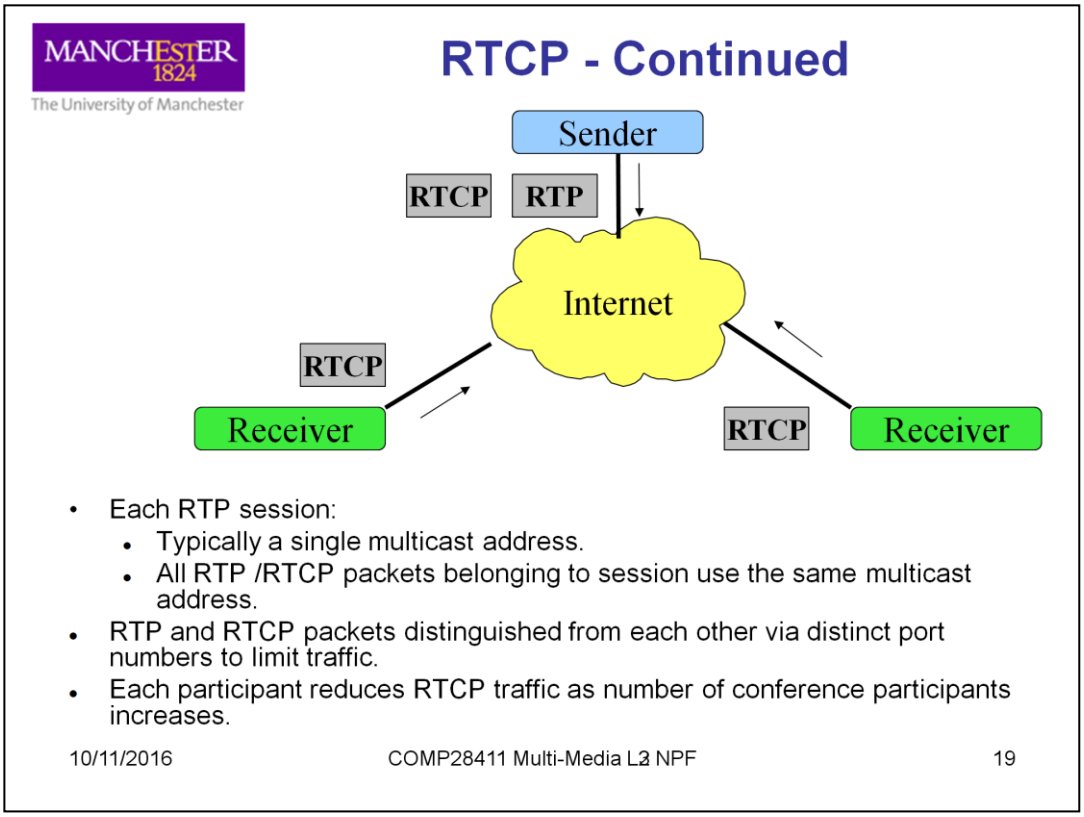


As we have already seen, RTP carries a minimal amount of information just sufficient to allow received data to be played back in the correct order and with the correct time distribution. You might query why there is both sequence number and a timestamp as the sequence number is redundant information. Why is the time stamp NOT redundant but the sequence number is?

RTP is used to distribute media in many different ways. It might be unicast from 1 source to 1 destination, from 1 source to many different unicast destinations or from 1 source to an unknown number of multicast destinations. Or even broadcast to everybody. It is useful for the source to know if the destinations are getting the data and whether the data is arriving on-time or late, how the inter-packet gap is varying (jitter) and how many packets never arrive (missing sequence number). In a unicast situation this data can be sent back using RTCP every so often and used to adjust the source's choice of CODEC and rate of sending packets to try and minimise congestion and maximise the quality of the media play back.

RTCP is purely for the control of the media sent using RTP. Another protocol such as RTSP the Real-time Streaming Protocol is used to carry out tape-recorder type operations such as start, stop, seek, rewind, fast-forward on the stream. RTSP commands only make sense with unicast streams and in fact most streaming uses unicast to facilitate this and also so as to allow people to start and stop watching or listening whenever they like. Multicast streams make sense for live media and for some bulk distribution of streaming data to content delivery systems nearer the locations where the data will be played back. These content delivery systems are used to reduce network load on servers and on the network interconnections via distribution.

With multicast, the number of destinations may vary with time. Therefore mechanisms to limit RTCP feedback from destination nodes are require so as not to overload the media source with RTCP traffic.



Multicast play-back is all sent to a single network address. Destinations must register their wish to receive the multicast with their local router which in turn registers the path with router between the source and the current router. Multicast allows a single packet to be sent and duplicated on need to deliver to many destinations. RTP and RTCP packets all use the same multicast IP address so they use the same routers and destinations. Using different usually adjacent ports is enough to separate the two streams of packets from each other. However, as more destinations register for the multicast, existing destinations must reduce the amount of feedback traffic they generate based on being told via RTCP the total count of destinations for the multicast.

RTCP Packets

Receiver report packets:

- Fraction of packets lost,
- Last sequence number,
- Average inter-arrival jitter

Sender report packets:

- SSRC of RTP stream.
- Current time.
- Number of packets sent.
- Number of bytes sent

Source Description Packets:

- E-mail address of sender.
- Sender's name.
- SSRC of associated RTP stream
  - Provide mapping between the SSRC and the user/host name

The content of the RTCP packets is fairly obvious and easy to guess. You can look up the exact fields and formats used on the Internet. The data is almost all metrics or statistics on the media transfer.

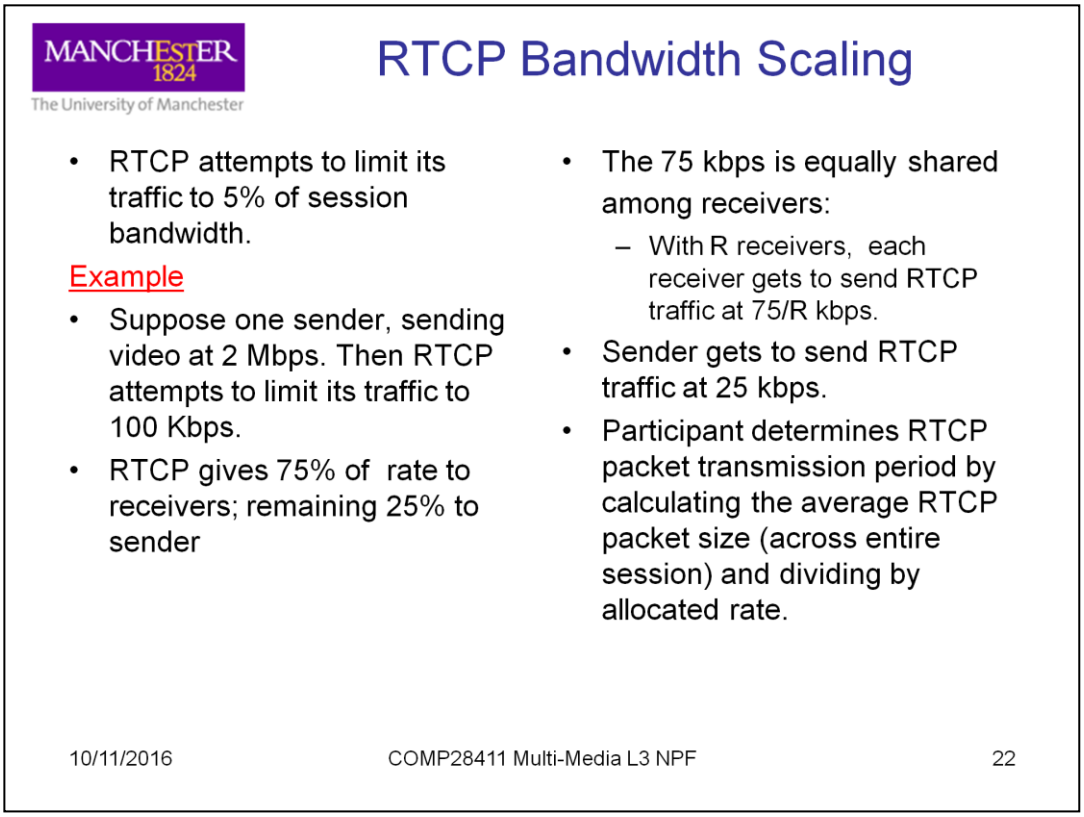
MANCHESTER 1824
The University of Manchester

Synchronization of Streams

- RTCP can synchronize different media streams within an RTP session
- Consider a video-conferencing application for which each sender generates separate RTP streams for video and for audio.
- Timestamps in RTP packets tied to the video or audio sampling clocks.
  - **Not** tied to wall-clock time
- Each RTCP sender-report packet contains (for most recently generated packet in associated RTP stream):
  - Time-stamp of RTP packet
  - Wall-clock time for when packet was created.
- Receivers use association to synchronize play-out of audio and video.

10/11/2016 COMP28411 Multi-Media L3 NPF 21

By using the same multicast address as the RTP uses for RTCP packets, these packets go to everybody in the session. This is great for conferencing as sender reports contain sequence number, timestamp and wall-clock time associated with the last packet sent. This then allows fairly exact synchronization of media coming from many different sources to be synchronized at destinations so that fully interactive sharing of what appears to be a single media stream can be achieved. The various streams ae mixed together for play back each obeying its own time stamp constraints with the local destination buffering able to adjust the different streams to align them at least roughly in time. Clearly this is not sufficient for high fidelity mixing of e.g. a distributed classical music orchestra or choir. Such an application would need much tighter constraints so that all instruments are heard to start at the same time. However, they still do not need to start exactly in synchronization as our ears are used to some sound distribution in large concert halls of many milliseconds (sound does only about 343m/s).



This is a simple example of how RTCP controls its amount of traffic in a distributed environment. The only issue left is how the value R is found. I think it is simple enough – and there are hints in my earlier notes. Think about who hears each multicast transmission.

MANCHESTER 1824
The University of Manchester

Some Live Media Requirements

- Easy to distribute to one or many destinations.
- Quick start/stop.
- Predictable short delays.
- Low error rate (note: not zero).

- Need to know:
  - Order of events/frames/samples.
  - Event/frame/sample timing.
  - What has arrived and not arrived.
  - What the content is.
  - When/how it synchronizes with other streams of events/frames/samples.

Note

Forward jumping is impossible for live media. Backwards jumping or time shifting can be done via cache and re-play. If multicast is used, lost packets must be very small as usually cannot request re-transmission.

Later streaming may use different methods e.g. download all before or while watching to allow back and forward jumps. For streaming, one or many (unicast) TCP streams are often used. Therefore, no lost packets, but sometimes have to wait due to congestion.

MANCHESTER 1824
The University of Manchester

Summary

- Delivering media is a continually developing solution.
- Demand, location, time of day, density of users all effect choice of method.
- We have:
  - TCP – End-to-end unicast, reliable but subject to delays, duplication, bottlenecks (hot-spots).
  - UDP - End to end unicast, unreliable, less delay problems due to accepting losses. Avoid duplication partially via multicast.
  - By sending multiple TCP streams side by side we can increase throughput or add redundancy to improve some Quality of Service (QoS) except in really bad congestion.
- **Next time:**
  - Making TCP work for multi-media, then UDP Multicast and Content Delivery …