Application-level QoS: Improving Video Conferencing Quality through Sending the Best Packet Next

Ian McDonald, Richard Nelson
Introduction

• Aim is to improve audio reception over video
• Why send packets when we know they will be of no use?
• Why send packets in order at the transport level?
• Use the information of priority and RTT at the transport level to send the best packet next
Use of DCCP

• Standards track RFC
• Has unreliable sessions and pluggable congestion control
• Designed to replace UDP for multimedia apps
• Available in Linux
• Use CCID3 which is TFRC (TCP Friendly Rate Control)
Testing setup

• Captured traces from Ekiga, Skype, MSN. Traffic was nearly all UDP. Applications used congestion control on top of UDP.

• Created model based on these and wrote a program to play/record this model using DCCP to compare standard vs send best packet next (not trying to model congestion control in Skype/Ekiga/MSN)

• Created programs to track time differences between machines and analyse results
Packet timing

200 msecs

P Q N P

P Q N P

✓

✗
SBPN1 algorithm

- Expiry time is packet creation time plus allowed delay (200 ms in testing).
- Packet has a priority – 2 is audio, 3 is video
- Check expiry time on transmit, taking into account half of rtt. Discard if it has expired, or will expire in transit.
Linux kernel implementation

- Linux 2.6.20 plus patches to meet TFRC performance
- Control data sent through sendmsg – expiry time, priority and method
- Kernel stores information in priority queue and DCCP already tracks RTT
- At time of transmission check for best packet to send and discard any expired packets
Testing setup

client  netem  server
Testing with loss - 80 ms RTT, queue length 5
Timing of packets

Packet transmission for video conference

- Video
- Audio

Bytes

Time
Being penalised for not sending

• Sending rate $X = \max(\min(X_{\text{calc}}, 2 \times X_{\text{recv}}), \frac{s}{t_{\text{mbi}}})$ ($s = \text{packet size, } t_{\text{mbi}} = 64$)

• $X_{\text{recv}}$ is updated at least once per RTT

• If no sending for a period, then $X$ plummets

• Need to transmit whenever possible

• sbpn2 sends a packet if expired if it is the only one in queue
Testing with loss - SBPN1 vs SBPN2 - 80 ms RTT, queue length 5
Difference in on time arrival - 80 ms RTT, queue length 5
Unmodified DCCP vs SBPN2 - 80 ms RTT, queue length 5
Unmodified DCCP vs SBPN2 - 80 ms RTT, queue length 32
Unmodified DCCP vs SBPN2 - 30 ms RTT, queue length 5
Unmodified DCCP vs SBPN2 - 150 ms RTT, queue length 5
Unmodified DCCP Faster Restart - 80 ms RTT, queue length 5
Future work/Conclusion

• Conclusion – a real improvement in on time audio reception and can be implemented on server only

• Tradeoff between bandwidth and improvement – penalised for not sending expired data

• Future – more complex algorithms, increased sending of expired packets to use bandwidth

• Look at time of bandwidth changing