A brief look at using multiple TCP connections for real-time flows (TCP F4RT!)

Piers O’Hanlon
Why?

• Many Firewalls and NATs only let TCP or unidirectional flows through
• Single TCP flow backs up as soon as a loss occurs
  – Connection waits for ACK time-out
  – Packet is only then retransmitted
  – > Use multiple TCP connections!
• See how well it actually works
Side step congestion events
How?

- Stripe each RTP packet across multiple TCP connections
  - Use IETF draft for RTP-TCP
    - \{2 Byte RTP Len\} + \{RTPkt\}
  - Send separate packet on each connection
    - Round-robin style
  - Receive when socket ready to read (i.e \textit{select()})
  - Setsockopt TCP\_NO\_DELAY – avoid Nagle

- Implemented in \texttt{vic as tcp net module using “layers” for multiple TCP connections}
  - “Client” or “server” mode plus cmd-line layer
Why not?

• Should be using UDP (but can’t)
• Not ideal use of TCP!
  – Attempts to circumventing most TCP control mechanisms
    • congestion control, reliability etc
• Performance may vary according to the multitude of TCP variants
• It makes the TCP people unhappy(;
Test scenario
Single TCP connection
Additional thoughts

• Relate number of connections to RTT?
• Check send buffers before sending
  – On Linux could tcp_diag to obtain details
• Open additional connections when old ones block.
• Could just replace UDP IP header type with TCP!
• Don’t do it?!