Observations on Internet Transport

Colin Perkins
Best Effort Is Grease For Robustness

• Best effort **forces** transport protocols and applications to adapt:
  • To variation in quality of service
  • To different link layers
  • To potentially suboptimal performance
  • To the presence of ossification

• In a global internetwork, performance cannot be guaranteed – the end points **must** be robust to network variability
  • Quality of service comes from robustness and adaptability

• That internet applications and services were able to adapt to the shift in traffic patterns and use due to the pandemic shows the importance of this
Signalling is Mandatory

• The network is increasingly heterogeneous and ossified
  • Once introduced nothing goes away – clean slate designs add to the mix
  • Can’t assume any particular protocol, feature, or option will work

• Probing, in the style of happy eyeballs or ICE, is increasingly critical

• APIs like Transport Services help because they automate complexity
  • Allow applications to ask for what they need
  • Give network flexibility to probe what works to meet those needs
Transport Protocols Are Harder Than You Think

• A key lesson from the development of QUIC: it’s hard to beat TCP
  • In performance
  • In adaptability
  • In robustness

• Corollary: it will be hard to beat DASH and RTP for their domains
  • **Not** because these are especially good or elegant, but because they have years of development, optimisation, adaptation, and have found the corner cases
  • This introduces complexity – effective protocols are not elegant
Good Specifications Are Vital

• New protocols only succeed if the offer compelling benefit over latest version of the old **and are easy to deploy**
  • Deployable protocols are implementable in user space, run over UDP, and actively fight ossification
  • If easy to implement, people will implement – already **many** more QUIC implementations than TCP stacks

• Widespread implementation requires clear specification
  • Protocols with 3-or-4 implementations can be poorly specified and understood by word-of-mouth
  • Supporting dozens of robust, secure, interoperable implementations requires heroic testing, high-quality machine-readable specifications that can be tested and tested against, or both
Implications for Future Transport

• We’re good at designing transport protocols – despite their complexity

• We’re **bad** at writing specifications that can be automatically tested or used to automatically derive (partial) implementations
  • Who will implement? How much expertise is, and should be, needed to build a correct implementation from the spec?
  • How can we expect to build secure, performant, and robust implementations from imprecise, inconsistent, poorly-tested specifications?

• Don’t just design transport protocols – design transport protocols that can readily be implemented and tested