What’s Missing with QoS?

A not-yet-draft comment on the state of IETF QoS architectures

QoS Elements

QoS is:

- Service Response Mechanisms
- Control structures

Today:

- Mechanisms appear to be well understood
  - filtering, conditioning, metering, queuing, discard
- Control mechanisms appear to be weakly understood

Control Structures

Signaling:

- e-2-e application signaling
- application - network signaling
- application - network - policy signaling

Management:

- a coherent view of the network operating state
- a coherent view of network resource allocation
- management of load to match operating capability

Where we are...

IntServ

- application-centric view of the QoS world
- pre-emptive reservation imposed upon the network

DiffServ

- network boundary-centric view of the QoS world
- no associated service delivery undertaking
- it scales, because it actually does nothing!

1. Application modification?

- Is QoS an application request to qualify the transport stack?
- Is QoS a policy-driven transport option that is transparent to the app?
- For IntServ
  - the application MUST be altered to be able to predict its load profile and negotiate this with the network and remote end
- For DiffServ
  - not applicable in either direction(!)

2. The Service Platform

- There appears to be no single service environment that possesses both service response accuracy and scaling properties
- IntServ attempts accuracy of e-2-e service but at the cost of per hop state
- DiffServ attempts to scale service response without any attempt at signaled service accuracy
  - no signalling from core to boundary
  - no signalling from app to boundary

3. QoS dynamic discovery?

- How can an application pin down a service-qualified path to an arbitrary destination?
  - DiffServ does not attempt to even come close to answering this question
  - IntServ is intended to achieve this, but there is the problem of scale of state in the core
  - hybrid systems appear to be gaining ground here

4. We need QoS Routing

- More accurately, there appears to be a need for the interior to signal to the boundary the current conditions of the core
- this implies the ingress TCBs to meter on a per-path basis in order to ensure the integrity of the boundary ingress actions
  - maybe this is itself a weakness, in that boundary conditions are assumed to operate with definitive integrity and the interior nodes configure according to boundary conditions
- routing in this case is a signaling process of core to boundary

5. TCP and QoS

- token buckets are TCP hostile
- TCP requires some level of ACK QoS symmetry

6. Aggregated Flow services

- does this make QoS sense at all?
  - Flow shaping of an aggregated flow looses application signalling
- This is perhaps a TE issue and not a QoS service issue

7. Too much choice

- for vendor and inter provider interoperability and end-to-end coherency, some group, somewhere will need to make a few choices and promote these as a grouped interoperable profile.

8. Deployment

- deployment will have visible operational cost.
- Without customers with deployed requirements this will not work
- But without deployed services there is no impetus to deploy the application and host signalling set

9. Service Performance Measurement

- How do I know that it works?
- How do I know that it works better than no QoS at all?
  - I = network operator
  - I = customer

10. No common accounting model

- this could be a real show stopper - as it is likely that every operator will want to extract the marginal costs of supporting this stuff from the punters who want to use it. Call me old fashioned if you want, but I matches the regular old model of cost appropriation!

11. Interprovider QoS

- This breaks down into two areas:
  - the technology uniformity to allow a QoS service inside one domain to cleanly map to the same service in another domain
  - the economicmodel of retail and settlements over unidirectional e-2-e services
- both are really furry uncertainties at the moment

12. Coping with disconnected islands of QoS

- any ngtrans veteran will look at this and laugh hysterically, especially as this cannot tolerate tunnels to bridge between QoS islands

13. What we have is a few parts: mechanisms, PHBs

- And what we really want is a deliverable SLA