Engineering the Internet with QoS Support

Lixia Zhang
University of California, Irvine

As more and more real-time applications, such as real-time video, are getting deployed over the Internet, the question of how to assure quality of service (QoS) is becoming increasingly important. Not only may real-time applications have stringent requirements on data delivery, but different applications may also have diverse requirements on delivery performance as measured by data delivery throughput, latency, jitter (variation in latency), and packet losses.

Building a large-scale network, such as today's Internet, is a great engineering challenge; providing QoS support over the Internet is also a great engineering challenge. Given engineering designs are all about tradeoffs, what is the best way to engineer the Internet to meet various QoS requirements imposed by different applications? Our answer to this question has been evolving over time as the Internet continues to grow rapidly, networking technologies continue to advance rapidly, and we continue to deepen our understanding about the relations among various design tradeoffs.

For over a decade the Internet engineering and research community has debated, designed, and ignored IP quality of service tools and techniques. Tremendous research efforts were devoted to network QoS support, resulting in a rich literature of various QoS supporting mechanisms and protocols. On the other hand, the primary operational response to the problem of data delivery quality has been *provisioning* the network with sufficient headroom so that congestion becomes unlikely. The gap between research and reality taught us that providing QoS support is not simply solving packet scheduling problems but is a challenging engineering decision.

Like all other engineering designs, there exist multiple alternative approaches that can all satisfy the same QoS requirements. Considering adding QoS support as an engineering design, one must fully explore the entire design space, and carefully evaluate alternative approaches. Generally speaking, an engineering design is constrained by multiple, and often conflicting, design goals, such as functionality, cost, complexity, reliability, and manageability, to name a few. For example, a more complex design may be able to meet the QoS requirements by using less link bandwidth; however, the increased complexity may lead to higher management cost as well as reduced reliability. Furthermore, different communities on the Internet may also view the tradeoffs differently. For example a corporate intranet may decide on its own to deploy advanced QoS supporting mechanisms internally, while a global Internet service provider would be more concerned with scalability and manageability of the same mechanisms. Thus, one architectural direction that appears to offer promising outcomes for QoS support is not one of universal adoption of the same supporting mechanisms, but is instead a tailored approach where a simple approach is used in the network backbone where scalability is a major design objective, and more sophisticated mechanisms can be deployed at the edge of the network where fine granularity of control and accuracy of the QoS measurement is desired. Architecturally, this points to a set of QoS-enabling mechanisms, including brute-force provisioning, and a number of ways these mechanisms can be configured to interoperate in a stable and consistent fashion.

Keywords:
QoS: Quality of network data delivery service. Generally speaking it includes the measure of data throughput, error rate, latency, and jitter.

Throughput: The rate at which user's data is delivered to the destination, usually measured in bits per second (bps).

Latency: The time it takes the network to deliver the data from a source to a destination.

Jitter: The measure of variation in latency. Due to statistical multiplexing used in a packet switched network, individual packets tend to experience different latencies crossing the network.