Latency and IP

Geoff Huston
August 2000

In many parts of the world Internet services are based on satellite access. Often these satellite-based services compete in the market with services based on terrestrial cable. What's the difference?

The essential difference is "latency", or the time taken for a packet to traverse the network. Latency is a major factor in IP performance, and reducing or mitigating the effects of latency is one of the major aspects of network performance engineering.

In order to understand why this is the case it is appropriate to start with the IP protocol itself. IP is a 'datagram' protocol. For data to be transferred across an IP network the data is first segmented into packets. Prepended to each packet is an IP protocol header. IP itself makes remarkably few assumptions about the characteristics of the underlying transmission system. IP packets can be discarded, reordered or fragmented into a number of smaller IP packets and remain within the scope of the protocol (ie. the protocol does not assume any particular network service quality, bandwidth, reliability, jitter or error
rate).

Layered above IP are two transport protocols, UDP and TCP. UDP is a lightweight extension to IP, implementing an unreliable packet transport protocol. UDP is used as the transport platform real time applications. UDP traffic typically forms some 8% of the total volume of traffic on a public Internet network, and the majority of this traffic is attributable to Domain Name queries - real time traffic, such as voice and video, is still a relatively small proportion of total traffic volumes. UDP traffic is non-adaptive, so that a UDP protocol stack will not attempt to adjust its sending rate to fit within available network capacity.

Latency most effects TCP. TCP is a reliable data transfer protocol. The requirements for reliability in the data transfer implies that the protocol will detect any form of data corruption on the part of the network and retransmit until the data is transferred successfully. This 'stop and retransmit' implies that there is no fixed rate for data transfer, nor will any implicit timing of packets be preserved by TCP. TCP is not a real-time protocol. TCP attempts to maximise its data transfer rate through dynamic rate adjustment. The way TCP achieves this is to continually test the network to see if a higher data transfer rate can be supported. When TCP encounters packet loss, it assumes that the loss is due to network congestion, and the protocol immediately reduces its data transfer rate.

TCP uses a 'sliding window control' to send data. By this, it is meant that the sender sends a sequence of packets (a 'window'), and then holds a copy of these packets while awaiting an acknowledgement (ACK) from the receiver that the packets have arrived. Each time an ACK for new data is received, the window is advanced by one packet, allowing the sender to send the next packet into the network.

The way in which TCP adjusts its rate is by increasing the window each time an ACK for new data is received, and reducing the window size when the sender believes that a packet has been discarded. Typically, in a steady state, the sender sends a burst of packets, and then waits for a corresponding burst of ACKs before sending any further data. When starting a TCP session a control method called "slow start" is used, where the window is increased by one packet each time an ACK is received. Across high latency paths, such as satellite paths, this function can result in packet bursts, where the sender sends a string of packets and then sits

idle, waiting for the corresponding string of ACK packets. With TCP, these packet bursts are injected into the network at twice the rate that is available on the data path. The network must perform rate adaptation by using queues to adapt the sending rate to the bottleneck rate. The larger the RTT, or, in other words the greater the latency, the greater this burstiness of the data.

A sender cannot continuously increase its sending rate without limit. At some stage the receiver will signal that its receiving buffer is saturated, or the sender will exhaust its sending buffer, or a network queue resource will become saturated. In the last case this network queue saturation will result in packet loss. When TCP experiences packet loss the TCP sender will immediately halve its sending rate and then enter "congestion avoidance" mode. In congestion avoidance mode the TCP sender will increase its sending rate by one packet every RTT interval. Paradoxically, this is somewhat slower than the "slow start" rate.

For TCP, the critical network characteristic is the latency. The longer the latency, the more insensitive TCP becomes in its efforts to adapt to the network state. As the latency increases, TCP's rate increase becomes slower, and the traffic pattern becomes more bursty in nature. These two factors combine to reduce the efficiency of the protocol and hence the efficiency of the network to carry data. This leads to the observation that, from a performance perspective and from a network efficiency perspective, it is always a desirable objective to reduce network latency.

If you cannot reduce latency, what can you do? The basic answer is: 'use very large buffers'. Tuning a TCP stack to support window scaling, selective acknowledgements and large TCP buffers is a very effective means of limiting the worst effects of latency. Setting up large buffers on the routers that connect to the satellite system also has an positive effect on TCP. You cannot always avoid latency, but you can reduce its most harmful effects.

Further Reading:

Tuning TCP: http://www.psc.edu/networking/perf_tune.html

RFC 2488, "Enhancing TCP over Satellite Systems using Standard Methods", http://roland.grc.nasa.gov/~mallman/papers/rfc2488.txt