Title: FAST (Fast AQM (Active Queue Management) Scalable TCP)

Contacts: Steven Low (slow@caltech.edu)

URLs / RFCs / Papers:

- "FAST TCP: From Theory to Experiments", C. Jin, D. Wei, S. H. Low, G. Buhrmaster, J. Bunn, D. H. Choe, R. L. A. Cottrell, J. C. Doyle, W. Feng, O. Martin, H. Newman, F. Paganini, S. Ravot, S. Singh; submitted to IEEE Communications Magazine, April 1, 2003
- http://netlab.caltech.edu/FAST/

Principle / Description of Operation

FAST is built on the idea of TCP Vegas. TCP Vegas was introduced as an alternative to the standard TCP (TCP Reno). Vegas does not involve any changes to TCP specification. It is merely an alternative implementation of TCP and all the changes are confined to the sending side. In contrast to the standard TCP, which uses packet loss as the measure of congestion, Vegas source anticipates the onset of congestion by monitoring the difference between the rate it is expecting to see and the rate it is actually realizing. Vegas's strategy is to adjust the source's sending rate in an attempt to keep a small number of packets buffered in the routers along the path.
Although experimental results show that Vegas achieves better throughput and fewer losses than standard TCP, Vegas lacked a theoretical explanation of why it works. Here they develop a model of Vegas and show that Vegas can potentially scale to high bandwidth network in stark contrast to the standard TCP. Further, they show that Vegas can become unstable at large delay. Also error in RTT estimation can distort Vegas and can lead to persistent queues and unfair rate allocation. They show that by augmenting Vegas with appropriate Active Queue Management algorithm like Random Exponential Marking (which requires modification in the router), it is possible to avoid the above-mentioned problems. FAST TCP aims at solving those problems by modifying just the TCP kernel at the sending hosts. Detailed description of the algorithm and implementation of FAST TCP is yet to be published (as on 6/1/03, should be out in a couple of months)

Supported operation mode:
Memory to memory (general transport)

Authentication: No

Implementations / API:
Not available yet (as on 6/1/03)

Congestion Control Algorithms:
The congestion control algorithm in FAST TCP is built on the algorithm used in TCP Vegas

Fairness:
Fair bandwidth allocation is one of the main objectives of FAST TCP but the detail about the mechanism used is yet to be published.

TCP Friendly:
Still under study

Predictable Performance Model:
This work was motivated by their earlier work, which developed a TCP/AQM congestion control system to achieve high utilization, low delay and dynamic stability at the level of fluid-flow models. But the algorithm used in FAST TCP and the theoretical explanation of why it should work is not published yet.

Results:
FAST TCP was demonstrated in a series of experiments conducted during the SuperComputing conference (SC2002). The demonstrations used an OC192 (10Gbps) link between StarLight (Chicago) and Sunnyvale, the DataTAG 2.5 Gbps link between Starlight and CERN (Geneva), an OC192 link connecting the SC2002 show floor in Baltimore and the TeraGrid router in StarLight Chicago and Abilene backbone of Internet2.
Using default device queue size (txqueuelen = 100 packets) at the network interface card and the standard MTU of 1500 bytes, the default Linux TCP (2.4.18), without any tuning on the AIMD parameters, achieved an average throughput of 185 Mbps, averaged over an hour, with a single TCP flow between Sunnyvale in California and CERN in Geneva via StarLight in Chicago with a minimum round trip delay of 180 ms. This is out of a possible maximum of 973 Mbps to the application, excluding TCP/IP overhead, limited by the gigabit Ethernet card, and represents a utilization of just 19%. Under the same experimental conditions, using the default device queue size (txqueuelen = 100 packets) and the standard MTU of 1500 bytes, FAST TCP achieved an average throughput of 925 Mbps (Utilization 95%), averaged over an hour. Even with a device queue size of 10,000 packets, the standard TCP was able to achieve a throughput of only 266 Mbps (Utilization 27%). With 2 TCP flows sharing the path, standard TCP was able to achieve 48% utilization (txqueuelen = 10,000 packets) whereas FAST TCP was able to achieve 92% utilization. With 10 flows, FAST TCP achieved a throughput of 8,609 Mbps (utilization 88%), averaged over a 6-hour period, over a routed path between Sunnyvale and Baltimore, using the standard MTU. The results using the standard Linux TCP implementation for 10 flows are not shown.
In all the experiments described above, the bottleneck was either the gigabit Ethernet card or the transatlantic OC48 link. The experiments conducted using Intel's pre-release experimental 10-gigabit Ethernet card on a single flow from Sunnyvale to Chicago using standard MTU, FAST TCP sustained just 1.3 Gbps. They claim this was due to the limitation in the CPU power at the sending and receiving systems.

Target Usage Scenario: Though it is intended to solve TCP's limitation in high-bandwidth large-delay environments, it is expected to perform well in conventional environments too.