Template for Alternative Transport Protocols

Title: GTP: Group Transport Protocol for Lambda-grids

Contacts (Include institutions):

Ryan X. Wu (xwu@ucsd.edu); Andrew A. Chien (achien@ucsd.edu)

URLs / RFCs / Papers

Ryan X. Wu, and Andrew Chien, "*GTP: Group Transport Protocol for Lambda-Grids*", in Proceedings of the 4th IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGrid), April 2004.

Principle / Description of Operation

Group Transport Protocol (GTP) [1] [2] is designed to provide receiver-based efficient multipoint-to-point data transfer on high speed links while achieving low loss and max-min fairness among network flows. This receiver-centric approach is especially useful to manage data transfer flows on Lambda-grids, where multiple wavelengths terminate at a single receiver, aggregating a much higher capacity than the receiver can handle.

GTP adopts a *receiver-driven response-request* model, in which the receiver explicitly specifies the desired rate along with data requests; and the sender sends back the requested data with receiver-specified rate. In order to support multi-flow management, enable efficient and fair utilization of the receiver capacity, GTP uses a receiver-driven centralized rate allocation scheme. In this approach, receivers actively measure progress (and loss) of each flow, estimate the actual capacity for each flow, and then allocate the available receiver capacity fairly across the flows. Because GTP's receiver-centric rate-based approach can manage all senders of a receiver, it enables rapid adaptation to flow dynamics, adjusting seamlessly when flows join or terminate.

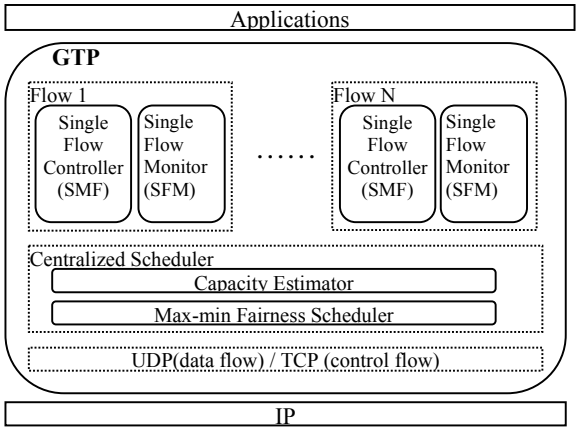

Figure: GTP Framework (Receiver)

Supported operation mode:

Memory to memory (general transport). Other operation modes (e.g. disk to disk) can be provided through XIO interface.

Authentication: None.

Implementations / API: Both C and XIO API will be available

Congestion Control Algorithms:

GTP implements two levels of flow control. For each individual flow, the receiver explicitly controls the sender's transmission rate (by sending rate requests to senders). This allows the flow's rate to be adjusted quickly in response to packet loss (detected at the receiver side). Ideally any efficient rate-based point-to-point flow control scheme could be 'plugged in'. As another lever of flow control, across the incoming flows at each receiver, there is a centralized scheduler. This structure exploits the insight that in lambda-Grids, congestion usually occurs at the end systems, especially the receivers. The scheduler at the receiver manages across multiple flows, dealing with any congestion or contention and performing max-min rate amongst them. The receiver actively measures per-flow throughput, loss rate, and uses it to estimate bandwidth capacity. It then allocates the available receiver capacity (can be limited by resource or the final link) across flows. This allocation is done once for each control interval in Max-min fair manner. Correspondingly, the senders adjust to transmit at the revised rates.

Fairness: Fair for contending flows at each receiver. Formal proof of global and local fairness properties are currently under study.

TCP Friendly: TCP Friendliness could be achieved by reserve certain bandwidth share for TCP traffic. As an ultimate solution, we consider to allocate TCP bandwidth share together with other GTP flows from the same centralized scheduler, and adjust TCP's window size according.

Predictable Performance Model: Analytical models are currently under study.

Results:

Preliminary results from an implementation of GTP show that for point-to-point single flow case, GTP performs as well as other UDP-based aggressive transport protocols (e.g. RBUDP, SABUL/UDT), achieving dramatically higher performance than TCP, with low loss rates. Results also show that for multipoint-to-point case, GTP still achieves high throughput with 20 to 100 times lower loss rates than other aggressive rate-based protocols. In addition, simulation results show that unlike TCP and other rate-based protocols, which are unfair to flows with different RTT, GTP responses to flow dynamics and converges to max-min fair rate-allocation quickly.

Target Usage Scenario:

GTP is intended to provide high performance data communication for applications running on high speed networks (e.g. Lambda-grids). Typical usage scenario include:

- fetch data from multiple remote data site concurrently, aggregating a much higher throughput than single data server could provide
- supporting multipoint-to-point and multipoint-to-multipoint data transfer with high throughput, low loss, and max-min fairness.

[1] Ryan X. Wu, and Andrew Chien, "*GTP: Group Transport Protocol for Lambda-Grids*", in Proceedings of the 4th IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGrid), April 2004.

[2] Ryan X. Wu, and Andrew Chien, "*Evaluation of Rate Based Transport Protocols for Lambda-Grids*", to appear in Proceedings of the Thirteenth IEEE International Symposium on High-Performance Distributed Computing (HPDC), Honolulu, Hawaii, June 2004.