ISSN:0975-9646

A Survey of Transport Protocols for Delay Tolerant Satellite Networks

Mitul K. Patel

Head and Assistant Professor

Department of Computer Engineering

Shree Swami Atmanand Saraswati Institute of Technology

Surat

Abstract — The transmission control protocol (TCP) is a connection-oriented, end-to-end reliable protocol working between hosts in packet-switched network, and between interconnected systems of such networks. Though TCP has been greatly successful in the terrestrial Internet and provides a robust reliable service for the transfer of data from one part of the earth to the other using an unreliable network layer, it performs quite poorly in wireless and satellite based network which is one type of delay tolerant network. In a satellite based network, where the RTT is more than that of the terrestrial counterpart, the use of a proper Transport Protocol becomes very important. In the terrestrial internet the main problem faced by the congestion control protocols is the uncertainty of the traffic condition in the network and the fear of overloading the network with excessive traffic. For this whenever the TCP sender doesn't receive any acknowledgement for the data it sent, it considers that the packet was dropped by the router because of congestion in the network and thereby reduce the congestion window. One of the major problems in a satellite-based network is the random packet errors, which are not common in the wired counterpart. TCP protocols react to the lack of arrival of acknowledgements or duplicate ACK as a sign of congestion. Therefore, the congestion window is reduced which leads to unnecessary throughput degradation. Moreover TCP injects a new packet into the network only after it receives an acknowledgement of the previously sent packets. This works fine as long as the RTT is moderate by keeping the network load within tolerable limits and maintaining the reliability of the data transmitted. But when the RTT increases, this mechanism creates the bottleneck in the performance of the protocol. The RTT is constrained by the speed of light and the total amount of data that needs to be sent in one RTT is given by the bandwidth-delay product of the link concerned and is not really achieved by the acknowledgement driven logic of TCP. Moreover there are problems because of bandwidth asymmetry and intermittency of the link.

Keywords— Satellite Network, Delay Tolerant Network, TCP

I. INTRODUCTION

Now a days, many applications required bulk data movement over a high speed network. For such applications, high bandwidth links are required between network nodes. Due to the ongoing convergence of computing, communication and control through the IP transport technology, long delays due to network inefficiency can cause user irritation and even loss of profit. The network performance in general can be boosted with high performance hardware, at the same time; the role of TCP implemented in software cannot be ignored.

Transmission Control Protocol (TCP) was initially design for wired network. The main functionality provided by TCP is congestion control and flow control. In terrestrial wired network, congestion control can be achieved by maintaining two different parameters called congestion window (cwnd) and advertised window (rwnd). The size of congestion window can be regulated by each sender depending upon the ACK received.

A. Satellite Network Architecture

A satellite network can be visualized as combination of two segments: The ground segment and the space segment.

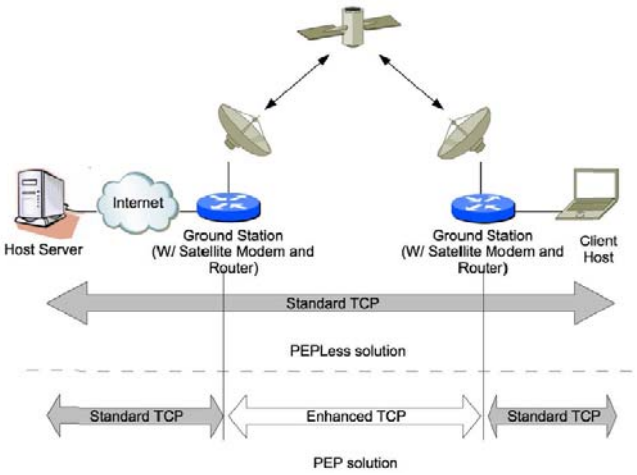

Fig. 1 A Typical Satellite Internet Access Topology [1]

The ground station consists of the gateway stations or hub stations, the user terminals and the network operation centre (NOC). The NOC includes the Network Control Centre (NCC) in charge of configuration management, capacity/bandwidth management, acquisition/synchronization control, performance management, alarm management, security management, billing, and accounting. The space segment includes the satellite equipment and the communication payload.

Satellite systems can be classified on the basis of orbit: GEO, LEO, MEO, HEO or hybrid.

A GEO satellite is located at an altitude of approximately 36,000 km. GEO satellites can have a large footprint (approximately 1/3 on the world surface), and, in theory, just 3 satellites are sufficient to cover the entire world surface. On the other hand, at this altitude high latency presents a serious disadvantage (about 500 ms round trip delay).

LEO (Low Earth Orbit) satellites orbit around the world at an altitude varying between several hundreds of km and a

few thousand km. In this case, the deployment of a constellation of several satellites is needed to achieve contiguous coverage, and efficient handover mechanisms must be implemented to allow service continuity. The propagation delay ranges from several ms up to 80 ms.

The characteristics of MEO (Medium Earth Orbit) satellites fall between GEO and LEO. Both LEO and MEO satellite systems have the advantage of lower propagation delays when compared with GEO system, but they are more complex to manage since both require continuous hand-off, dynamic routing algorithms and tracking mechanisms.

HEO (Highly Elliptical Orbit) satellites are located in elliptical orbits in which the Earth assumes the position of one of the two foci over planes inclined at 63.4° with respect to the equatorial plane. According to the Kepler's second law, satellites in HEO orbits move slowly around the apogee looking "almost" stationary. As a consequence, HEO satellites cover high latitude regions very effectively with a good (high) elevation angle. Finally, hybrid systems use a combination of more than one type of orbit.

II. TRANSPORT LAYER PROTOCOLS FOR DELAY TOLERANT SATELLITE NETWORK

Plenty of new TCP-based protocols are currently available. Some of them introduce "modified" congestion control scheme, while others replace congestion control with a rate control scheme. Of course, all the presented TCP variants address only a sub-set of the satellite constraints (often changing scenario by scenario) above mentioned, nevertheless they can help in identifying a set of strategy to fix inefficiency of the standard TCP over satellite links.

A. High Speed TCP

High-Speed TCP [2, 3] aims to improve the loss recovery time of the standard TCP. Its behavior is equivalent to TCP standard TCP for small congestion windows (below a fixed threshold) while it is more responsive on big windows. It accomplishes this with the use of a modified "Additive Increase Multiple Decease" (AIMD) behaviour during congestion avoidance, dependent to the current window size. In fact, the additional and multiplying factors are proportional to the window size in this way:

$$\begin{cases} cwnd = cwnd + \dfrac{a(cwnd)}{cwnd} & \forall\ ACK\ received \\ cwnd = \big(1 - b(cwnd)\big) \cdot cwnd & if\ 3\ dupACKs \end{cases}$$

With $a(cwnd)$ proportional to *cwnd* and $b(cwnd)$ inversely proportional to *cwnd*. Then, a faster *cwnd* is allowed in CA while after a loss *cwnd* is less than halved.

B. Scalable TCP

As the High-Speed TCP, Scalable TCP modifies the standard TCP only in the management of large cwnd during the congestion avoidance. The concept and the implemented algorithm of this version are so close to the High-speed TCP, that they are defined in the same RFC [2].

C. BIC & CUBIC TCP

BIC (Binary Increase Congestion control) is implemented in the standard Linux kernel since the 2.6.7 version, and it is also used as default TCP for Linux machines. The protocol combines two schemes called "additive increase" and "binary search increase". In the former, congestion control is considered as a searching problem in which the system gives yes/no feedback through packet loss as to whether the current sending rate is larger than the network capacity. The boundary points for such a searching are named "minimum window size" (W_{min}) and "maximum window size" (W_{max}). In particular, W_{max} is cwnd value just before the last fast recovery and W_{min} is the cwnd just after the fast recovery. The algorithm repeatedly computes the midpoint between W_{min} and W_{max}, set the next cwnd size (1 RTT later) to the midpoint, and checks for feedback in the form of packet losses. Based on this feedback, the midpoint is taken as the new W_{max} if a loss is detected and as the new W_{min} otherwise. This process is repeated until the difference between W_{max} and W_{min} falls below a preset threshold, called "minimum increment" (S_{min}). This technique allows bandwidth probing to be more aggressive initially, and becomes less aggressive as the current cwnd gets closer to W_{max}. Then, the increase function is logarithmic. In order to ensure faster convergence, the "binary search increase" is combined with an "additive increase" strategy. When the gap between the midpoint and the current cwnd is larger than a preset "maximum increment" (S_{max})), cwnd is increased by S_{max}. A "Slow Start" strategy is adapted from the "old" W_{max} to $W_{max} + S_{max}$.

CUBIC is an enhanced version of the TCP BIC less aggressive at start up avoiding the additive increase. CUBIC does not perform the binary search while enforces a single algorithm for window adjustment (namely a cubic function) that is:

$$W_{cubic} = C \cdot (t - K)^3 + W_{max}$$

Where C is a scaling factor, t is the elapsed time since the last window reduction, W_{max} is the window size just before the last window reduction and $K = \sqrt[3]{(W_{max} \cdot \beta)/C}$ where β is a multiplicative decrease factor after a packet loss event.

D. TCP Vegas

TCP Vegas [4] proposes a new sender-based congestion control mechanism aiming to prevent congestion events by monitoring the difference between the expected rate and the actual rate. Specifically, TCP Vegas is based on three mechanisms:

1. A new retransmission mechanism. A TCP Vegas sender retransmits more aggressively. In particular, it measures the RTT for every sent segment. Such a RTT measurement is used to set a RTO timer for each segment. Vegas checks whether RTO has expired and then decides for a retransmission in the following situations:

I. when a duplicate ACK is received;

II. when the first or the second non-duplicate ACK after a retransmission is received. Basically, this mechanism aims at reducing the time to detect multiple packet losses.

2. A new Congestion Avoidance mechanism. To predict the congestion, TCP Vegas considers not only the dropped packets, but also the variations of the estimated RTT. To implement this congestion control, it defines the "BaseRTT" as the RTT without congestion (minimum of all measured RTT), and then the expected throughput is calculated as the ratio between the transmission window size and BaseRTT. On the other hand, TCP Vegas calculates the cur-rent sending rate by dividing the amount of outstanding bytes by the RTT. Finally, it compares the current and the expected throughput and adjusts the congestion window accordingly:

$$Diff = Expected - Current$$

In fact, by defining two threshold values, α and β, TCP Vegas implements the following algorithm:

$$\begin{cases} If(Diff < \alpha) & linear\ cwnd\ increase\ in\ next\ RTT \\ If(Diff > \beta) & decrease\ cwnd\ in\ next\ RTT \\ If(\alpha < Diff < \beta) & cwnd\ unchanged\ in\ next\ RTT \end{cases}$$

3. *A modified Slow Start mechanism*. If the standard Slow Start mechanism is not limited by buffer dimensions at the end hosts, the congestion window size will double every RTT (if there are no losses), until an overrun of connection capacity. Therefore, the losses may be on the order of half the current congestion window. TCP Vegas prevents congestion events during the Slow Start allowing an exponential window growth only every other RTT. During this time, the congestion window is fixed. Thus, a valid comparison of the expected and actual rates can be performed.

E. Fast TCP

Fast TCP [5] is a sender-side only modification to the Vegas TCP congestion avoidance algorithm. The congestion control mechanism of FAST TCP can be divided into four components. These four components are functionally independent so that they can be designed separately and upgraded asynchronously. The *data control* component determines which packets to transmit, *window control* determines how many packets to transmit, and *burstiness control* determines when to transmit these packets. These decisions are made based on information provided by the *estimation* component. Window control regulates packet transmission at the RTT timescale, while burstiness control works at a smaller timescale. Under normal network conditions, the "window control" algorithm periodically updates the congestion window W based on the average RTT according to:

$$W = \min\left\{2 \cdot W, (1-\gamma) + \gamma \cdot \left(\frac{baseRTT}{RTT} \cdot W + \alpha\right)\right\}$$

where $\gamma \in (0; 1]$, RTT is the current average round-trip time, baseRTT is the minimum RTT observed so far, and α is a protocol parameter that controls fairness and the number of packets each flow buffered in the network. It is proved in [5] that, in the absence of delay, this algorithm is globally stable and converges exponentially to the unique equilibrium point where every bottleneck link is fully utilized and the rate allocation is proportionally fair.

F. TCP Peach

TCP-Peach is composed of two new algorithms, namely Sudden Start and Rapid Recovery, as well as the two traditional TCP algorithms, Congestion Avoidance and Fast Retransmit [6]. Sudden Start and Rapid Recovery are designed to replace respectively the Slow Start and Fast Recovery algorithms of standard TCP. The new algorithms are based on the novel concept of using dummy segments to probe the availability of network resources without carrying any new information to the sender. Dummy segments are low-priority segments generated by the sender as a copy of the last transmitted data segment. If a router on the connection path is congested, it discards the IP packets carrying dummy segments first. Consequently, the transmission of dummy segments does not cause a throughput decrease of actual data segments. If the routers are not congested, then the dummy segments can reach the receiver. The sender interprets the ACKs related to dummy segments as evidence that there are unused resources in the network, and increases its transmission rate accordingly. By exploiting dummy segments, Sudden Start provides a fast opening of the congestion window at the beginning of the connections, irrespective of the actual RTT, while Rapid Recovery is used to react to losses [6]. The main requirement of TCP Peach is that the network routers implement some form of DiffServ policy, to differentiate between high and low priority traffic.

G. TCP Hybla

TCP Hybla [7][8] has been conceived with the primary aim of counteracting the performance deterioration originated by the long RTTs typical of satellite connections. It consists of a set of procedures which includes an enhancement of the standard congestion control laws for both the Slow Start and the Congestion Avoidance phases, the mandatory adoption of the SACK policy, the adoption of Hoe's channel bandwidth estimation, the use of timestamps and the implementation of packet spacing techniques. The modification of the standard congestion control rules is dictated by the TCP Hybla ideal aim of obtaining for long RTT connections the same instantaneous segment transmission rate of a comparatively fast reference TCP connection (e.g. a wired one). To this scope, TCP Hybla introduces a new parameter: the normalized round trip time, defined as the ratio between the actual RTT and the round trip time of the reference connection denoted by RTT_0:

$$\rho = \frac{RTT}{RTT_0}$$

Then, the standard congestion control laws are replaced by the following, which represent the TCP Hybla congestion control rules when receiving an ACK:

$$W = \begin{cases} W_i + 2^{\rho} - 1 & SS \\ W_i + \frac{\rho^2}{W_i} & CA \end{cases}$$

H. TCP Westwood

TCP Westwood [9] was introduced for the purpose of limiting the consequences of the losses introduced by a wireless channel, which are always erroneously ascribed to

congestion by the TCP protocol. To this end, TCP Westwood introduces a modification of the Fast Recovery algorithm called "Faster Recovery". Instead of halving the *cwnd* after three duplicate ACKs, and fixing the *ssthresh* to this value, TCP Westwood sets the *ssthresh* as a function of the actual available bandwidth. In this way, channel losses do not cause the dramatic slow-down of the transmission rate as in the standard TCP. The bandwidth is estimated by measuring and averaging the rate of returning ACKs. Then, after loss detection, TCP Westwood sets the *ssthresh* and *cwnd* as follows:

$$ssthresh = \hat{b} \cdot RTT_{min}$$

$$cwnd = \begin{cases} ssthresh & if\ cwnd > ssthresh \\ 1 & after\ a\ timeout \end{cases}$$

where, $\hat{b}$ is the estimated bandwidth.

III. Conclusion

In this survey, explanation of transport protocols used in delay tolerant satellite network has been discussed. These protocols perform best in terrestrial wired and wireless network but when these protocols have been used in delay tolerant satellite networks, their performance start degrading. Tradition TCP protocols are based on fixed parameters. Also, they are designed to work for specific application and network type. Almost all transport layer protocols designed for wired or wireless networks works based on propagation delay i.e. RTT. Another parameter that is used by some of the transport layer protocol is bandwidth delay product. Now a days, research is going on to develop new transport protocol which can give high performance in delay tolerant network.

References

[1] Alain Pirovano & Fabien Garcia, "A New Survey on Improving TCP Performances over Geostationary Satellite Link", Network and Communication Technologies, Vol. 2, No. 1, 2013

[2] S. Floyd, *HighSpeed TCP for Large Congestion Windows*, RFC 3649, Experimental, Dec. 2003.

[3] E. Souza, D.A. Agarwal, A HighSpeed TCP Study: Characteristics and Deploy- ment Issues, LBNL Technical Report LBNL-53215.

[4] L. S. Brakmo, L. L. Perterson, *TCP Vegas: End-to-End Congestion Avoidance on a Global Internet*, IEEE Journal on Selected Areas in Communications, vol. 13, n. 8, pp. 1465-1480, 1995.

[5] J. Cheng, D. X. Wei, and H. Steven, *TCP FAST: motivation, architecture, algorithms, performance*, In Proceedings of IEEE Infocom, Mar. 2004.

[6] Ian F. Akyildiz, Giacomo Morabito, and Sergio Palazzo,"TCP-Peach: A New Congestion Control Scheme for Satellite IP Networks", IEEE/ACM TRANSACTIONS ON NETWORKING, VOL. 9, NO. 3, JUNE 2001

[7] C. Caini, R. Firrincieli, *A new transport protocol proposal for Internet via satellite: the TCP Hybla*, In Proceedings of ESA ASMS conference 2003, Jul. 2003.

[8] C. Caini and R. Firrincieli, "TCP Hybla: a TCP Enhancement for Heterogeneous Networks", International Journal of Satellite Communications and Networking, vol. 22, n. 5, pp. 547-566, Sep. 2004.

[9] Shimaa Hagag, Ayman El-Sayed (IEEE Senior Member), "Enhanced TCPWestwood Congestion Avoidance Mechanism (TCP WestwoodNew)", International Journal of Computer Applications (0975 – 8887), Volume 45– No.5, May 2012