ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 8, August 2015

Improve a Performance of System Using AIMD

[1]P.Gayathri, [2] A. Rama

[1,2] Asst. Professor, Dept. of Information Technology, Bharath University, Chennai, Tamil Nadu, India

gayathri.it@bharathuniv.ac.in

ABSTRACT: The goal of our work is to present a protocol that cans maximally utilize the bandwidth of these private links through a novel performance-based system flow control. On high-speed, high-latency, congestion-free networks, a protocol should strive to accomplish two goals: to maximize good put by minimizing synchronous, latency-bound communication and to maximize the data rate according to the receiver's capacity. Latency-bound communication is one of the primary problems of TCP due to the positive acknowledgement congestion window mechanism. When UDP is used in tandem with TCP, UDP packets can be sent asynchronously, allowing the synchronous TCP component to do its job without limiting the overall bandwidth.

I. INTRODUCTION

A certain class of next generation science applications needs to be able to transfer increasingly large amounts of data between remote locations. Toward this goal, several new dedicated networks with bandwidths upward of 10 Gbps have emerged to facilitate bulk data transfers.

Throughput is halved in the presence of detected packet loss and only additively increased during subsequent loss-free transfer. This is the so-called Additive Increase Multiplicative Decrease algorithm (AIMD). In a dedicated link, however, packet loss due to congestion can be avoided. The partitioning of bandwidth, therefore, can be done via some other. More intelligent bandwidth scheduling process, leading to more precise throughput and higher link utilization.The receiver then sends back positive acknowledgements (ACKs) in order to receive the next window. Using timeouts and logic, the sender decides which packets are lost in the window and resends them.

On networks with high latencies, reliance on synchronous communication can severely stunt any attempt for high-bandwidth utilization because the protocol relies on latency-bound communication.

II. LITERATURE SURVEY

Flow Control:

Flow control, term very much concerned to our work, is the process of managing the rate of data transmission between two nodes to prevent a fast sender from outrunning a slow receiver. It provides a mechanism for the receiver to control the transmission speed, so that the receiving node is not overwhelmed with data from transmitting node. Flow control should be distinguished from congestion control is used for controlling the flow of data when congestion has actually occurred.

Flow control is important because it is possible for a sending computer to transmit information at a faster rate than the destination computer can receive and process them. This can happen if the receiving computers have a heavy traffic load in comparison to the sending computer, or if the receiving computer has less processing power than the sending computer.

Types of Flow Control:

There are two types of flow control. They are

a. Network Congestion
b. Data buffer

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 8, August 2015

Network Congestion:

A prevention mechanism that provides control over the quantity of data transmission that enters a device.

Data Buffer:

A prevention control mechanism that provides storage to contain data- bursts from other network devices, compensating for the variation of data transmission speeds.

Existing technique

FAST TCP: : (**F**AST **A**QM **S**calable **T**CP, where **AQM** stands for **A**ctive **Q**ueue **M**anagement, and **TCP** stands for **T**ransmission **C**ontrol **P**rotocol)

FAST TCP (also written "FastTCP") is a TCP congestion control avoidance long-distance, high latency links. The congestion control algorithm in the current TCP, which

has gone through several enhancements since, . It has performed remarkably well and is generally believed to have prevented severe congestion as the Internet scaled up by six orders of magnitude in size, speed, load, and connectivity. It is also well-known, however, that as bandwidth-delay product continues to grow, TCP Reno will eventually become a performance bottleneck itself. The following four difficulties contribute to the poor performance of TCP Reno in networks with large bandwidth-delay products:

1) At the packet level, linear increase by one packet per Round-Trip Time (RTT) is too slow, and multiplicative decrease per loss event is too drastic.

2) At the flow level, maintaining large average congestion windows *requires* an extremely small equilibrium loss probability.

3) At the packet level, oscillation is unavoidable because TCP uses a binary congestion signal (packet loss).

4) At the flow level, the dynamics is unstable, leading to severe oscillations that can only be reduced by the accurate estimation of packet loss probability and a stable design of the flow dynamics.. Its advantage over loss-based approach is small at low speed, but decisive at high speed, as we will argue below[1]. As pointed out in , delay can be a poor or untimely predictor of packet loss, and therefore using a delay-based algorithm to augment the basic AIMD (Additive Increase Multiplicative Decrease) algorithm of TCP Reno is the wrong approach to address the above difficulties at large windows. Instead, a new approach that fully exploits delay as a congestion measure, augmented with loss information, is needed. FAST TCP uses

this approach. Using queuing delay as the congestion measure has two advantages. First, queueing delay can be more accurately estimated than loss probability both because packet losses in networks with large bandwidth-delay product are rare events (probability

on the order ____ or smaller), and because loss samples provide coarser information than queueing delay samples. Indeed, measurements of delay are noisy, just as those of loss probability. Each measurement of packet loss (whether a packet is lost) provides one bit of information for the

filtering of noise, whereas each measurement of queueing delay provides multi-bit information. This makes it easier for an equation-based implementation to stabilize a network into a steady state with a target fairness and high utilization. Second, the dynamics of queueing delay seems to have the right scaling with respect to network capacity. This helps maintain stability

as a network scales up in capacity We exploit these advantages to address the four difficulties of TCP Reno

In this technique they describe an alternative congestion control scheme for TCP, called FAST. FAST TCP has three key differences. First, it is an equation-based algorithm and hence eliminates packet-level oscillations. Second, it uses queueing delay as the primary measure of congestion, which can be more reliably measured by end hosts than loss probability in fast long-distance networks. Third, it has a stable ow dynamics and achieves weighted proportional fairness in equilibrium.

A FAST TCP flow seeks to maintain a constant number of packets in queues throughout the network. The number of packets in queues is estimated by measuring the difference between the observed round trip time (RTT) and the *base RTT*, defined as the round trip time when there is no queueing. The base RTT is estimated as the minimum observed RTT for the connection. If too few packets are queued, the sending rate is increased, while if too many are queued, the rate is decreased.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 8, August 2015

High Speed TCP:

High speed TCP mechanisms deliver all of the familiar characteristics and benefits of TCP, but scale performance to hundreds of megabits per second for individual TCP.

High speed TCP addresses issues in the "congestion avoidance" algorithm by specifying an approach where each individual high-speed TCP connection manages its congestion window as if it were an aggregate of multiple TCP connections. This allows a TCP window to expand more appropriately in a high-speed WAN environment, as well as to reduce its TCP window less disruptively when congestion is encountered[2].

Second adjustment to the TCP algorithms which addresses the management and expansion of each connection's TCP window when in the "slow start" phase. Rather than the old method involving exponential expansion of the TCP window from a direct approach that expands the TCP window more expeditiously in the "slow start" phase. This approach accounts for the round-trip latency in the WAN and prevents TCP connections from stalling at low throughputs relative to overall available bandwidth for extended periods of time[3].

2.3.3 UDT:

UDT is the data transport protocol proposed by this dissertation to support the distributed data intensive applications in wide area high-speed networks. UDT addresses the solution by investigating two orthogonal research problems: 1) the design and implementation of transport protocols with respect to throughput and CPU usage; and, 2) the Internet congestion control algorithm with respect to efficiency, fairness, and stability[4].

UDT is an application level, end-to-end, unicast, reliable, connection-oriented streaming data transport protocol. The UDT protocol is completely at user space above UDP, i.e., it uses UDP to transfer user data and protocol control information. UDT uses packet-based sequencing to check packet loss and guarantee data reliability. It is specially designed for high-speed bulk data transfer by aiming to remove or reduce the overhead of memory copy, loss information processing, acknowledging, etc. UDT provides reliable streaming data transfer service, similar to TCP. The UDT protocol supports a large variety of control algorithms. Moreover, it supports congestion control algorithms to be configured at run time, thus each UDT flow can have its own control algorithm and it can change the algorithm at any time. The built-in (default) UDT congestion control algorithm is proposed to utilize high bandwidth efficiently and fairly.

The UDT algorithm uses a loss-based AIMD mechanism. Bandwidth estimation technique is used to optimize its increase parameter dynamically. A random decrease factor is used to remove the negative effect of loss synchronization. UDT is not used to replace TCP on the Internet where the bottleneck bandwidth is relatively small and there are large amounts of multiplexed short life flows. However, when coexisting with TCP flows, UDT is designed not to occupy more bandwidth than TCP does unless the TCP flows fail to utilize their fair
share due to TCP's efficiency problems in high bandwidth-delay product (BDP) environments. This design goal is due to the fact that TCP will still be used in these high BDP networks, and application that uses UDT may sometimes run on public networks[5].

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 8, August 2015

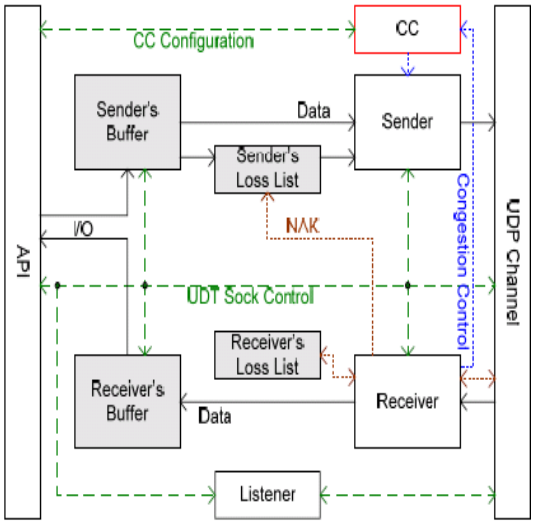

Figure 2.3. 3 UDT architecture

2.3.4 Hurricane:

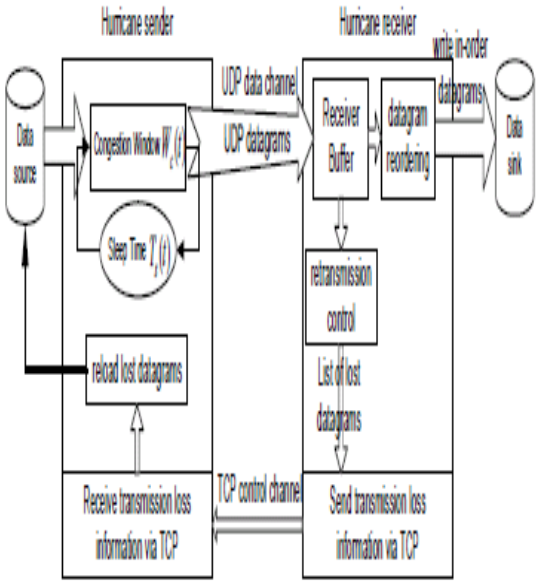

Figure 2.3. 4 Hurricane architecture

Hurricane is based on the basic ideas of existing protocols but provided us a more convenient parameter tuning mechanism. By an explicit manual tuning of the parameters, we were able to achieve goodput rates. Hurricane is developed exclusively for high-speed file transfer on dedicated links. It does not attempt to be TCP friendly unlike some UDP-based protocols (UDT, for example) that are meant to be used over shared networks. Like other protocols in the same category, Hurricane employs a loose flow control mechanism and various (high) levels of persistent source rates. The design goal of Hurricane is to maximize link utilization without any expectation of sharing the channel [6].

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 8, August 2015

It is worth pointing out that in the design of Hurricane, we fix both window size and sleep time to achieve a flat source rate for a transport experiment with a specific bandwidth request. For transport protocols with more sophisticated rate control strategies such as the one we use later for stable streams, these control parameters are dynamically adjusted during one transport session. Hence, we associate these parameters here with time just for generalization purposes. Note that even with fixed window size and sleep time, the measured source rate is not likely to remain exactly the same over time due to the interactions between the kernel and processes[7].

A Hurricane receiver accepts incoming UDP datagrams, which are either written immediately to the local storage device if they arrived in order, or placed temporarily in a buffer for reordering otherwise. Whenever a control event is triggered, a sequential scanning is performed on the receiving buffer to check for a list of missing datagrams. The datagram ID numbers on this list are grouped together and sent over a separate TCP channel to the Hurricane sender. The sender then reloads the missing datagrams into the sender congestion window for retransmission[8].

2.3.5 Tsunami:

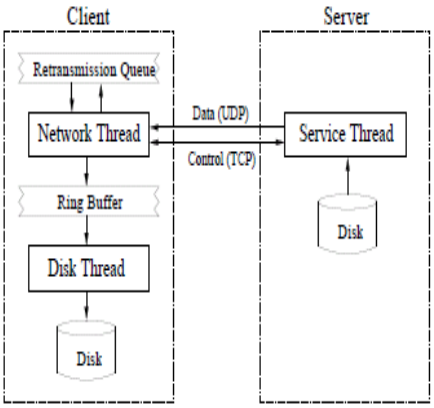

Figure 2.3. 5 Tsunami architecture

Our target networks for using Tsunami are typically not congested, the protocol did need some form of flow control in order to avoid exceeding the capability of the client to handle the incoming traffic. However, we still wanted to avoid collapsing the transmission rate in the presence of low-level packet loss. For this reason, we settled upon an adjustable error threshold. Packet loss above the threshold causes an exponential rise in the inter-packet delay; loss below the threshold causes an exponential decrease in the inter-packet delay until a target rate has been met[9].

During a file transfer, the client has two running threads. The network thread handles all network communication, maintains the retransmission queue, and places blocks that are ready for disk storage into a ring buffer. The disk thread simply moves blocks from the ring buffer to the destination file on disk. The server creates a single thread in response to each client connection that handles all disk and network activity[10].

The client initiates a Tsunami session by connecting to the TCP port of the server. Upon connection, the server sends a small block of random data to the client. The client then *xor*'s this random data with a shared secret, calculates an MD5 checksum, and transmits the result to the server. The server performs the same operation on the random data and verifies that the results are identical. We thus establish client authentication[11].

After exchanging protocol revision codes, the client sends the name of the requested file to the server. If the server indicates that the file is available, the client sends its desired block size, target transfer rate, error threshold, and inter-packet delay scaling factors. The server responds with the length of the file, the agreed-upon block size, the

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 8, August 2015

number of block, and a timestamp. The client then creates a UDP port and transmit the port number. At this point, we are ready to transmit the file.

III. CONCLUSION

The protocol based on the ideas in this paper has shown that transfer protocols designed for high –speed networks should not only rely on good theoretical performance but also be intimately tied to the system hardware on which they run. Thus, high performance protocol should adapt in different environments to ensure maximum performance, and transfer rates should be set approximately to proactively curb packet loss. If this relationship properly understood, optimal transfer rates can be achieved over high-speed, high-latency networks at all times without excessive amounts of user customization and parameter guess work.

REFERENCES

1. Y. Gu and R.L. Grossman, "UDT: UDP-Based Data Transfer for High-Speed Wide Area Networks," Computer Networks, vol. 51, no. 7, pp. 1777-1799, 2007.
2. Udayakumar R., Khanaa V., Saravanan T., "Analysis of polarization mode dispersion in fibers and its mitigation using an optical compensation technique", Indian Journal of Science and Technology, ISSN : 0974-6846, 6(S6) (2013) pp. 4767-4771.
3. N.S.V. Rao, W.R. Wing, S.M. Carter, and Q. Wu, "UltrascienceNet: Network Testbed for Large-Scale Science Applications,"IEEE Comm. Magazine, vol. 43, no. 11, pp. S12-S17, Nov. 2005.
4. Bhuvaneswari B., Hari R., Vasuki R., Suguna, "Antioxidant and antihepatotoxic activities of ethanolic extract of Solanum torvum", Asian Journal of Pharmaceutical and Clinical Research, ISSN : 0974-2441, 5(S3) (2012) pp. 147-150.
5. X. Zheng, M. Veeraraghavan, N.S.V. Rao, Q. Wu, and M. Zhu,"CHEETAH: Circuit-Switched High-Speed End-to-End TransportArchitecture Testbed," IEEE Comm. Magazine, vol. 43, no. 8, pp. 11-17, Aug. 2005.
6. Udayakumar R., Khanaa V., Saravanan T., "Chromatic dispersion compensation in optical fiber communication system and its simulation", Indian Journal of Science and Technology, ISSN : 0974-6846, 6(S6) (2013) pp. 4762-4766.
7. K. Wehrle, F. Pahlke, H. Ritter, D. Muller, and M. Bechler, LinuxNetwork Architecture. Prentice-Hall, Inc., 2004.
8. Sathyanarayana H.P., Premkumar S., Manjula W.S., "Assessment of maximum voluntary bite force in adults with normal occlusion and different types of malocclusions", Journal of Contemporary Dental Practice, ISSN : 1526-3711, 13(4) (2012) pp.534-538.
9. A.C. Heursch and H. Rzehak, "Rapid Reaction Linux: Linux withLow Latency and High Timing Accuracy," Proc. Fifth Ann. LinuxShowcase & Conf. (ALS '01), p. 4, 2001.
10. Udayakumar, R., Khanaa, V., Saravanan, T., "Synthesis and structural characterization of thin films of sno2 prepared by spray pyrolysis technique", Indian Journal of Science and Technology, ISSN : 0974-6846, 6(S6) (2013) pp.4754-4757.
11. A. Hanushevsky, "Peer-to-Peer Computing for Secure HighPerformance Data Cop," http://www.osti.gov/servlets/purl/826702-5UdHlZ/native/, Apr. 2007.
12. T.Nalini ,A.Gayathri,HVS Based Enhanced Medical Image Fusion ,International Journal of Innovative Research in Computer and Communication Engineering,ISSN (Print) : 2320 – 9798 , pp 170-173, Vol. 1, Issue 2, April 2013.
13. S.Thirunavukkarasu, r.K.P.Kaliyamurthie ,EFFICIENT ALLOCATION OF DYNAMICRESOURCES IN A CLOUD,International Journal of Innovative Research in Computer and Communication Engineering, ISSN: 2249-2651, pp 24-29,Volume1 Issue3 Number2–Dec2011.
14. K.G.S. VENKATESAN,Planning in FARS by dynamic multipath Reconfiguration system failure recovery in Wireless Mesh Network,International Journal of Innovative Research in Computer and Communication Engineering,ISSN(Online): 2320-9801,pp 5304-5312,Vol. 2, Issue 8, August 2014.
15. G.Michael, An Empirical Approach – Distributed Mobility Management for Target Tracking in MANETs ,International Journal of Innovative Research in Computer and Communication Engineering , ISSN (Print) : 2320 – 9798 , pp 789-794 , Vol. 1, Issue 4, June 2013.
16. G.AYYAPAN , Malicious Packet Loss during Routing Misbehavior - Identification, International Journal of Innovative Research in Computer and Communication Engineering, ISSN(Online): 2320-9801, pp 4610-4613 ,Vol. 2, Issue 6, June 2014