



TCP Revisited

CONTACT INFORMATION:

phone: +1.301.527.1629

fax: +1.301.527.1690

email: whitepaper@hsc.com

web: www.hsc.com



PROPRIETARY NOTICE

All rights reserved. This publication and its contents are proprietary to Hughes Systique Corporation. No part of this publication may be reproduced in any form or by any means without the written permission of Hughes Systique Corporation, 15245 Shady Grove Road, Suite 330, Rockville, MD 20850.

Copyright © 2006 Hughes Systique Coporation

TABLE OF CONTENTS

<u>SECTION</u>	<u>PAGE</u>
1.0 TCP REVISITED	4
1.1 CONGESTION COLLAPSE.....	4
1.2 HOW TO PREVENT CONGESTION COLLAPSE	4
1.3 TCP BASICS	4
1.4 TCP RENO	4
1.5 CONGESTION DETECTION	4
1.6 SLOW START.....	5
1.7 CONGESTION AVOIDANCE.....	5
1.8 THE TCP VEGAS AND RELATED METHODS.....	5
1.8.1 Background.....	5
1.9 CRITIQUES AND CRITICISMS OF TCP RENO	5
1.10 HOW TCP VEGAS WORKS.....	5
1.11 ANALYSIS OF TCP VEGAS	6
2.0 REFERENCES.....	7

1.0 TCP REVISITED

TCP is transmission Control Protocol, the Layer4 protocol for communication over both wireline as well as wireless links. It is one of the most widespread of protocols in usage today. All key applications defining the web today, http, email transfer, file transfer, etc. use TCP as the backbone transmission protocol.

TCP offers a stream oriented reliable delivery service. Unusually, for a transport layer protocol, it has no explicit support for QoS. Perhaps this is one of the reasons for its success - by eschewing features, it has managed to be simultaneously very efficient and very powerful. Also, it is designed to operate on any standard network layer protocol with as few assumptions as possible.

The key feature in TCP is a very sophisticated scheme of congestion management and rate adaptation. There are multiple variants of TCP, each principally differing in the way they execute these features. For a definitive specification of TCP protocol, please see the standard in rfc0793. In the rest of this article, we shall review the theory of TCP congestion avoidance and the different algorithms available for handling the same.

1.1 Congestion collapse

Congestion collapse takes place when multiple sources start transmitting together at a constant rate. If the number of sources is sufficiently high, this will be greater than the clearing rate of the network. This will, then, lead to overflow of buffers and packet dropping. The dropped packets will, in turn, lead to further retransmissions, which will only increase the load in the network. Eventually, we will reach the stage where no data is going through.

1.2 How to prevent congestion collapse

The job of the transmitter is then, to discover a transmission rate so that the combined transmission rate of all transmitters is less than the network clearing rate. The fact that this is even possible comes from a very deep result in distributed autonomous control, see [Kleinrock95].

1.3 TCP basics

TCP is a sliding window, asynchronous ARQ protocol. It is termed asynchronous because it does not use clocks explicitly for most operations and also, does not need clocks of a very fine granularity. Rather, whenever a packet is received, this is treated as an event, which triggers the next activity. The lack of clocks ensures that processing load is relatively low and TCP has run successfully on 5-6 generations of computers, ranging from the 5Mhz VAX machines of the 1970s to modern day embedded systems and teraflop supercomputers.

1.4 TCP Reno

TCP Reno is the most widespread variant of TCP currently. It is derived from the original version of TCP (TCP Tahoe) which was circulated as part of the FreeBSD protocol stack. The TCP stack in Solaris, Linux and Windows is a variant of TCP Reno.

TCP Reno uses two key techniques for congestion management. One is slow-start and the other is congestion avoidance.

1.5 Congestion detection

Standard TCP was designed for a network where links were relatively reliable, but processing computers were slow and starved for memory. Link speeds were limited (the first Internet was run on 56kb/s dialup lines, which, in the USA would have meant a noise margin of more than 40dB), but the ability of computers to transmit over them was slower still. Consequently, packet dropping was usually due to lack of buffering capability.

Standard TCP thus uses packet dropping as a sign of congestion. Packet dropping can be signalled when the receiver gets a discontinuous sequence, or, in cases of heavy congestion, an entire burst is lost. The difference between the first and the second is the method of feedback. While discontinuous

packets can be signaled to the transmitter using duplicate acknowledgments, a whole burst being lost can only be detected autonomously by the transmitter using a retransmission timeout. One of the biggest challenges in TCP protocol design is to correctly compute the RTO wait period, taking into account queuing delay which can vary substantially and will also dominate link delay.

1.6 Slow Start

Slow start is the mechanism which is used when a TCP session is started. Instead of starting transmission at a fixed rate, the transmission is started at the lowest rate possible and doubled. The rate of doubling is tied to the rate at which acknowledgments come back; thus, the rate is doubled every round trip time. If congestion is detected, the transmission rate is reduced to half of what it is currently and the congestion avoidance process starts. The rate doubling and reduction is nothing but a binary search for the 'right' transmission bandwidth.

1.7 Congestion avoidance

The congestion avoidance algorithm starts where slow start stops. The TCP increases its transmission rate linearly, by adding one additional packet to its window each transmission time. If congestion is detected at any point, the TCP reduces its transmission rate to half the previous value. Thus the TCP seesaws its way to the right transmission rate. Clearly, the scheme is less aggressive than the slow-start phase (note the linear increase against the exponential growth in slow start).

1.8 The TCP Vegas and related methods

The initial design of TCP was for a certain network characterization as discussed above. While it was a very successful design from the point of view of a cooperative network being shared amongst a small group, it was soon clear that there were many issues when it came to scaling it to a worldwide heterogeneous Internet.

1.8.1 Background

A key transition in analysis came when TCP was analyzed, not from the point of view of a single protocol operating in an uncertain environment, but a group of 'greedy' protocols competing with each other for limited resources. [Shenker95] gives a fascinating review in his seminal paper, showing how the 'bandwidth hunting' nature of TCP actually maps to zero-sum competitive games. Subsequently, using game theoretic techniques, equilibrium behaviour and incentives have been developed.

1.9 Critiques and criticisms of TCP Reno

The analysis above yielded the following criticism of TCP Reno.

- By its very nature, a TCP is designed to drive a network into saturation. A TCP will never reduce its transmission rate unless it sees a packet drop. This will only occur when the network is overloaded. Thus a combination of TCPs will together ensure that a network is driven into overload constantly. This makes it difficult for new connections to enter the system, since they are at a natural disadvantage due to slow start.
- TCP Reno attempts to equalize the window size for competing connections. However, a measure of the bandwidth achieved is the bandwidth delay product, not the window size only. Thus, TCP Reno is unfair to connections with larger round-trip times.

1.10 How TCP Vegas works

TCP Vegas uses an alternate method to detect congestion. Instead of depending on packet drops, the

TCP monitors variations in measured round-trip times and achieved throughputs. Depending on these changes, it decides whether to increase the window or decrease the window. The details of the Vegas congestion algorithm are provided in [Brakmo94, Scowcroft95].

Another important innovation in TCP Vegas is the manner in which packet drops are detected. A study by [Balakrishnan98] showed that upto 50% of packet drops in standard TCP are only detected by timeout. In TCP Vegas, whenever a duplicate ack is received, the TCP immediately checks all pending retransmission timers. If a retransmission timer has only a small amount of time to expire (small with reference to the measured round trip time), it is retransmitted immediately.

1.11 Analysis of TCP Vegas

TCP VEGAS created a significant excitement when it was initially described. Analyses such as [Low02] showed that algorithmically, it was superior to the standard TCP Reno. This was confirmed by simulation results in [Mo99].

However, detailed analysis of TCP Vegas belies its claims. While it does indeed perform better than TCP Reno, this is not because of its superior congestion management algorithm, but because of the innovations in timeout management. The problem was pointed out in one of the very first papers, but disregarded by researchers for a long time; TCP Vegas convergence is problematic. Recently [Chloe03] and others have issued enhancements to TCP Vegas (Vegas-A) which claim better convergence properties.

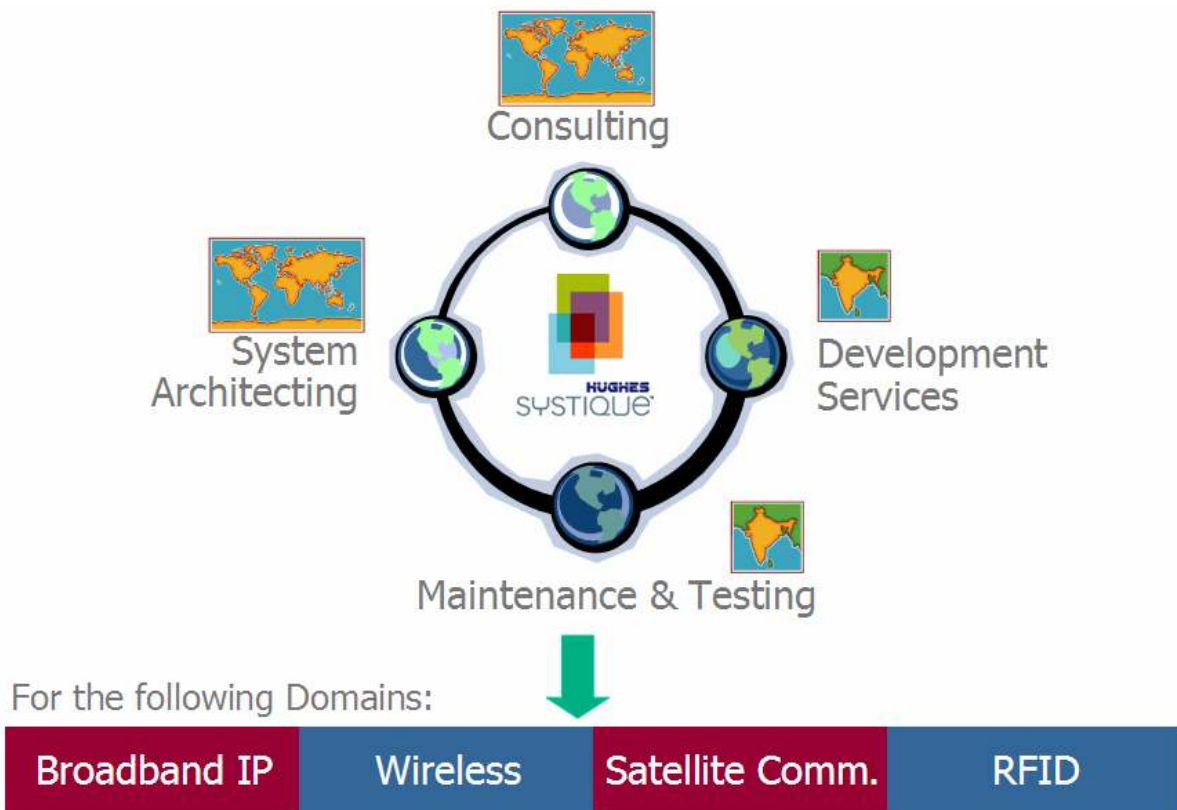
2.0 REFERENCES

- [Shenker95] S. J. Shenker. Making greed work in networks: A game-theoretic analysis of switch service disciplines. IEEE/ACM Transactions on Networking, 3(6):819-- 831, 1995.
- [Kleinrock93] B. Tung, and L. Kleinrock, "Distributed Control Methods," Proceedings of the Second International Conference on High Performance Distributed Computing, 1993.
- [Balakrishnan98] Hari Balakrishnan, Katz, "TCP Behaviour of a busy Internet Server: Analysis and Improvements" Proc. of the IEEE Infocom, 1998
- [Brakmo95] Brakmo, Peterson, "TCP Vegas: End-to-end congestion avoidance on a global internet" IEEE JSAC 1995
- [Mo99] Mo, La, Anantharam, Valrand, "Analysis and comparison of TCP Reno and Vegas", Proceedings of the IEEE Infocom, 1999
- [Low02] Low, Peterson, "Understanding TCP VEGAS: A duality model", Journal of the ACM, 2002
- [Choe03] Choe, Low "Stabilized Vegas", Proceedings of the IEEE Infocom, 2003

APPENDIX A ABOUT HUGHES SYSTIQUE CORPORATION

HUGHES Systique Corporation, part of the HUGHES group of companies, is a leading communications Consulting and Software company. We provide Consulting, Systems Architecture, and Software Engineering services to complement our client's in-house capabilities. Our "Best Shore" model coupled with an experienced management and technical team team is capable of delivering a total solution to our clients, from development to deployment of complex systems, thus reducing time, risk and cost

HSC Solution Space:



CONTACT INFORMATION:

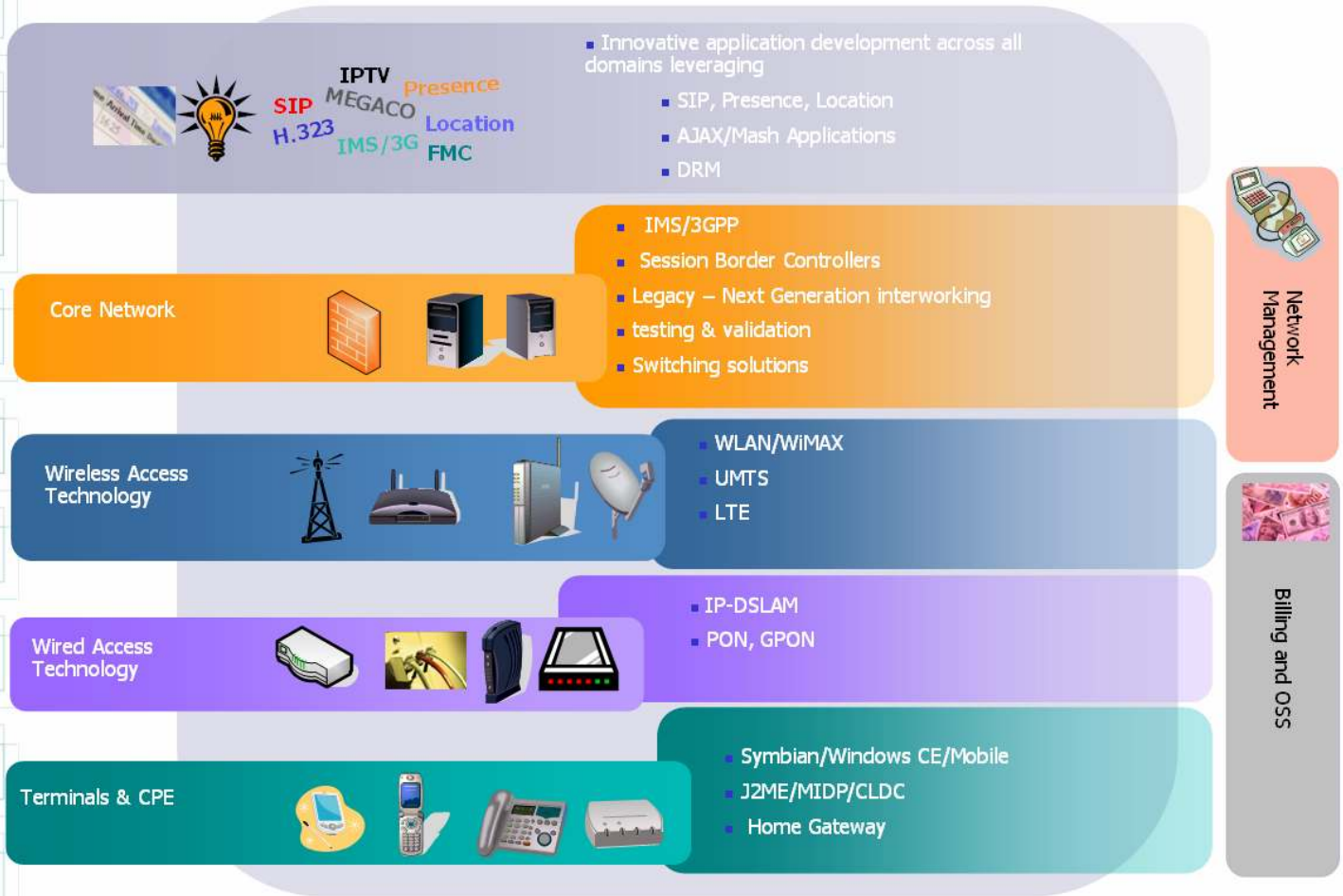
phone: +1.301.527.1629

fax: +1.301.527.1690

email: whitepaper@hsc.com

web: www.hsc.com

HSC Expertise Areas in Brief:



CONTACT INFORMATION:

phone: +1.301.527.1629

fax: +1.301.527.1690

email: whitepaper@hsc.com

web: www.hsc.com