Network Measurement in 2G/3G Cellular Network

A dissertation submitted in partial fulfilment of
The requirements for the degree of
Masters of Technology
in
Computer Application

By,
Amitsingh R Chandele
(2010JCA2204)

under the guidance of
Dr. Aaditeshwar Seth
Department of Computer Science and Engineering

Department of Computer Science
And
Department of Mathematics
Indian Institute of Technology Delhi

May 2012

Certificate

This is to certify that the dissertation titled **"Network Measurement in 2G/3G Cellular Network"** being submitted by **Amitsingh R Chandele,** Entry Number 2010JCA2204 to the Indian Institute of Technology Delhi, for the award of the degree of Master of Technology in Computer Application, Department of Mathematics, Indian Institute of Technology Delhi, is a record of bona fide research work carried out by him under our guidance and supervision at the Department of Computer Science and Engineering, Indian Institute of Technology, Delhi. The work presented in this thesis has not been submitted elsewhere, either in part or full, for the award of any other degree or diploma.

Dr. Aaditeshwar Seth
Department of Computer Science and Engineering
Indian Institute of Technology Delhi
New Delhi 110 016

Date:

Acknowledgments:

First and foremost, I would like to express my sincere gratitude to my thesis advisor, Dr. Aaditeshwar Seth for his wisdom, knowledge, and insight that guided me during my Masters study. It has been my great honour to work with him, and this experience will be a significant presence in my life forever.

Besides my advisor, I would like to thank the rest of my thesis committee and project members: Zahir Korodia, Dr. Vinay Ribeiro, Sipat Turkosi, Anirban Mahanti and Sebastien Ardon for their encouragement and insightful comments.

My sincere thanks also go to Prof. Naveen Garg for constant support and encouragement.

I am grateful to my colleagues: Arun Singh, Harish Iyar and the staff of Computer Science Department at Indian Institute of Technology Delhi.

I would like to thank my parents for their love and support.

Abstract

The rapid growth in the popularity of cellular networks in India has led to aggressive deployment and a rapid expansion of services. According to the Telecom Regulatory Authority of India, India is the second largest and fastest growing mobile market in the world [1]. However there is very little understanding of performance of cellular data connectivity in India.

In this work, we attempt to characterize the cellular data networks available in India through a large scale experiment setup consisting of more than 50 measurement points. We measure TCP throughputs and RTT, network maps through traceroutes, and even link layer radio resource management strategies. Tests will be conducted once every hour over a period of one month at each location. Geographical diversity has been ensured through partnership with NGOs that have rural and urban offices across northern India.

We have conducted several lab experiments and two pilot tests: one urban and one rural to gain early insights into the nature of Airtel GPRS, MTNL HSDPA, and Reliance EVDO networks available in India. Some of the interesting observations are: (a) Airtel and MTNL connections were unavailable at night time in the rural area, (b) The bursty behaviour of Cubic congestion avoidance algorithm results in large number of packet retransmissions and packet losses in GPRS, low BDP connection. (c) Delays caused by Radio resource management policies impacts the end user experience.(d) For Airtel GPRS connection TCP Reno found to be performing better that TCP Cubic in terms of number of retransmission and with little improvement in throughput. (e) We also found that parameters such as TSO, delayed acknowledgement and Initial window size adds onto bursty behaviour of TCP cubic & hence degrades the performance.

Understanding of such network parameters and their impact on performance will help explore strategies for optimizations at end-hosts as well as within the service provider network.

Contents

List of Figures

List of Tables

Chapter 1: Introduction

1.1 Motivation:

In the past decade, cellular communications have experienced an explosive growth due to recent technological advances in cellular networks and cellular phone manufacturing. According to Telecom Regulatory Authority of India, India is the second largest and fastest growing mobile market in the world & number of telephone subscription have reached to 926.53 million registering the growth of 36.22% in year 2010-11. TRAI also reports that the growth rate of subscribers in rural areas during the year (2010-11) was higher at 40.64 % compared to 34.11% in urban areas [1].

However, there is very little understanding of performance of cellular data connection in India. With traditional broadband internet connectivity stuck at 5% [1], cellular data connectivity provides an opportunity to the digital divide. Since the performance of the cellular networks when compared to traditional broadband network is low, it is important to understand the operating characteristics of cellular networks. The performance of the cellular networks is affected by several factors such as implementation and parameterization of transport layer protocols, network buffer sizes, latencies, radio resource management by service providers, their availability etc. Impact of such factors is still unexplained in context of cellular network in India.

Despite the widespread deployment of cellular networks and their importance to emerging applications, they remain relatively unexplored by the academic community. One reason for the lack of large-scale measurement studies on cellular networks is that researchers have limited access to cellular environments. Most academic institutions and research laboratories do not access the Internet over cellular network. We overcame this problem by our well planned measurement methodology in which we will be deploying measurement nodes across multiple locations in Rural India.

1.2 Objective:

In this project we attempt to answer the following questions in context of cellular data networks in India:

- What is the throughput provided by different service providers? Do they match the advertised bandwidth?
- How do properties like RTT, DNS lookup time, network buffer sizes, etc affect network performance?
- Are there diurnal and day of week patterns?
- How does radio resource management by service provider impact the end user experience?
- How does TCP perform over different cellular access technologies?

- How available are cellular data provider in terms of time and at different geographic locations?

Understanding of these factors like throughput, RTT, DNS lookup time, network buffer size, radio resource management etc, in context of cellular network will help us characterize the cellular network & hence improve its performance.

1.3 Layout of thesis:

This thesis comprises of four main parts: literature survey about network measurement, Measurement methodology followed by different experiments performed and finally Results and analysis.

Chapter 2 describes the literature survey which includes brief introduction to measurement methods for link capacity and available bandwidth. It then summarizes in short most popular TCP variants like Tahoe, Reno, New Reno and Cubic. Finally this chapter talks about the related work that has already been done, their methodology and their findings.

Chapter 3 presents the measurement methodology used in the project. It starts with the identification of the parameters to measure, followed by how to measure these parameters. It then goes on to explains the experimental setup used for the measurement and structure of the test suite that we have developed. Finally in this chapter we discuss algorithmic steps for various tests from measurement suite

Chapter 4, in this chapter we describe different tools which were used in project like tcpdump. Dig, iperf etc. Then it presents various experiments we conducted to infer the properties of the network and to understand their impact on performance.

Chapter 5 focuses on TCP dynamics; it describes how various parameters of the TCP such as Initial window size and optimization techniques such as segmentation offloading impacts the performance of TCP.

Chapter 6 here we presents the results obtained from the experiments stated in chapter 4.

Chapter 7, in this chapter the discussion and summary of the work is given. In addition, some of the possible ways to continue this research are also addressed.

Chapter 2: Literature Survey

In this chapter we describe related work about bandwidth estimation tools, TCP variants and some of the large scale measurement studies in other context.

2.1 Available bandwidth and link capacity estimation tools:

In communication network, high available bandwidth is useful, as it allows high volume of data transfer. Obtaining accurate measurement of this metric can be crucial to effective deployment of QoS services in network. Several applications need to know the bandwidth characteristics of network. For example some peer to peer application need estimate of available bandwidth before peers can join the network, overlay networks can configure their routing table based on the available bandwidth of the overlay links etc. This section we describe existing methods for measuring the Available bandwidth and Link capacity.

Available bandwidth is unused capacity along the bottleneck link along the network path. Link capacity is maximum rate at which packets can be transmitted along the path. The existing tools for measuring the Available bandwidth and Link capacity can be categorized as:

1. Variable size probing:

This method finds the capacity for each hop along the path. It measures the latency from source to every hop in the path. It sends probing packets with controlled TTL field so that they expire at a particular hop. This hop then sends an ICMP "TTL exceeded" message back to source. This ICMP packet is then used by source to get the RTT.

```
RTT= 2*(Transmission delay + Propagation delay + Queuing delay)
```

This technique sends multiple probing packets and assumes that at least one of the packets will not experience queuing delay. Using this minimum RTT till current hop it estimates the capacity of this hop. This technique fails if path along which capacity estimation is being done, has layer 2 “store and forward” switches then this technique underestimates the capacity.

Tools using VSP: pathchar[2], Clink[3], pchar[4].

2. Packet pair/ train dispersion technique:

This method finds the end to end capacity of the path. A source sends multiple packet pairs (pair containing same size packet) to receiver. Based on the dispersion observed between the packets it makes an estimate of the capacity of the path. This technique assumes that there is no cross traffic along the path. If the cross traffic is introduced then it causes underestimation or overestimation errors. It tries to mitigate this problem by repeating the experiment several times and then applying some statistical methods.

Tools using Packet pair/ train dispersion: Bprobe[5], Pathrate[6], Sprobe[7]

3. Self loading periodic streams:

Used for measuring end to end available bandwidth of the path. Here the source sends a stream of equal sized packets from source to destination at constant rate R. Based on the rate at which the packets have been received by the destination it makes an estimate of available bandwidth of the path.

Tools using SLoPS: pathload[8], IGI[9], pathChirp[10].

4. Train of packet pair: [11, 12]

This technique is used to estimate the available bandwidth of the path. It sends a train of packet pairs. Here the source sends a train of packet pair with gradually increasing rate. Based on the rate at which this train is received it makes an estimate of end to end available bandwidth. An important difference between this method and Self loading periodic streams is it can also make an estimate of capacity of the tight link.

2.2 Various TCP Congestion Avoidance Algorithms: [13, 14]

TCP is a transport layer protocol which has slow start, congestion avoidance, fast retransmit and fast recovery as four main algorithm built in it.

Slow Start: With every acknowledgement received TCP increases its congestion window by number of packets acknowledged, which effectively causes it to double the congestion window size in each RTT.

Congestion avoidance: The exponential increase in congestion window (cwnd) is bounded by a variable “ssthresh”, slow start threshold. If the current cwnd crosses the ssthresh then TCPenters in its congestion avoidance phase, in which it increases it’s cwnd by 1 each RTT.

Fast retransmit: In TCP when acknowledgement with same sequence number is received for predefined number of times (normally 3) then it retransmits the packet without waiting for timer to go off.

Fast recovery: After a packet loss is detected by TCP, it normally enters into slow start phase but when fast recovery is in place then it reduces its window size by half. When an acknowledgement of new data (called recovery ACK) is received, it returns to congestion avoidance phase.

Variation and improvement in these four basic algorithms according to the need of the network has produced different TCP algorithm. We describe some of these in brief in following sections.

2.2.1 TCP Tahoe:

TCP Tahoe is the simplest variant of TCP. It doesn't have fast recovery. In congestion avoidance phase it treats 3 duplicate acknowledgements as timeout. On timeout it does a fast retransmit, after which it enters in to slow start phase by reducing the congestion window size to 1.

2.2.2 TCP Reno:

When 3 duplicate acknowledgements are received then it reduces its congestion window by half, then performs fast retransmit and enters into fast recovery mode. On timeout it will enter into slow start phase. It performs well when single packet is lost but when burst packet losses occur its performance degrades.

2.2.3 TCP New Reno:

It attempts to counter the bulk packet loss problem that Reno suffers from. It modifies the fast recovery algorithm (where a new data acknowledgement is enough to take TCP out of fast recovery). In TCP New Reno, it is required that all the packets outstanding at the start of the fast recovery period be acknowledged to take TCP out of fast recovery to congestion avoidance. TCP New Reno works by assuming that the packet that immediately follows the partial acknowledgement received at fast recovery is lost, and retransmit the packet.

2.2.4 TCP Cubic:

The protocol modifies the linear growth function of existing TCP standards to be cubic function (contains both concave and convex functions) in order to improve the scalability of the TCP over fast and long distance network. The graph of cubic function is as shown in figure 2.2.4.1, below:

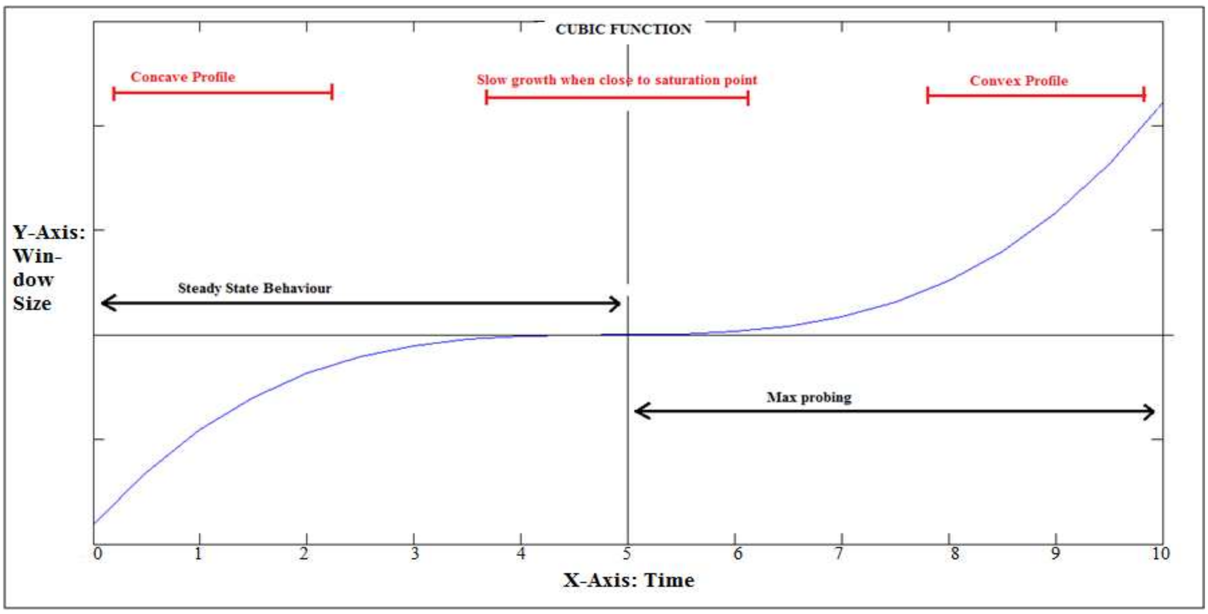

Figure 2.2.4.1: Cubic Function

Features of TCP Cubic:

- Its window growth depends on the real time between two consecutive congestion events. Window growth of CUBIC is cubic function of elapsed time from last congestion event.
- Depending on the value of cwnd (current congestion window size) it works in three different modes.
  - If cwnd is less than window size that standard TCP (AIMD) would reach at time t after the loss event, then cubic is in **TCP Mode.**
  - Otherwise if cwnd is less than Wmax then CUBIC is in **concave region.**
  - If cwnd is larger than Wmax then Cubic is in **convex region.**

TCP Mode: (TCP friendly region)

This mode is just to make TCP CUBIC behave fairly with other standard TCP flows which might be there in the network. (It can be disabled).Here it makes an estimate where would standard TCP increased its window size (on reception of acknowledgement) & grows its own window size to that point.

Concave region:

When received acknowledgement in congestion avoidance phase it receives acknowledgement, if it’s not in TCP mode and cwnd is less than Wmax, then protocol is in concave mode.

Convex region:

The phase when you have reached beyond maximum window size (recorded at last congestion event) then it indicates there is some more bandwidth available in the network. You start slowly & move aggressively to find next maximum.

2.3 Measurement Studies in Other Contexts

2.3.1 Characterizing Residential Broadband Networks [15]

This paper was the first large scale study of major DSL and cable providers in North America and Europe. It characterizes the several properties of broadband network including link capacities packet round-trip times and jitter, packet loss rates, queue lengths, and queue drop policies. To measure the characteristics of broadband links they proposed a probe train test. These probe trains were sent at different rates. High rate probe trains are termed as Floods and low rate probe trains are termed as trickles. Floods measure the networks performance under congestion, and trickles measure it under normal condition. Following are the different types of trains paper proposed:

- Asymmetric large TCP flood: This train saturates downstream link.
- Symmetric large ICMP flood: It saturates links in both uplink and downlink.

- Symmetric small TCP flood: It saturated the network in both downstream and upstream directions but with much smaller packets.
- Symmetric large-ICMP trickle: This train did not saturate the downstream or upstream links.
- Symmetric small-TCP trickle: This packet train did not saturate the downstream or upstream links.

Using these trains they measured the link bandwidth, packet latencies and jitter and packet loss.

Key Findings: Their analysis revealed important ways in which residential networks differ from the conventional wisdom about the Internet

2.3.2 Characterizing delays in Norwegian 3G networks [16].

This paper presented first look at a long term measurements of mobile broadband data connections from three different network operators in Norway. The emphasis of this paper was on delay characteristics. Measurements were performed over duration of 6 months from 90 locations using ping and traceroute as tools for measurement.

Key findings: Some of the important findings of the paper are:

- There are large differences in the delay between the operators.
- The observed difference cannot be explained by different modems alone.
- While there are sometimes large differences between monitors of the same operators, they mainly belong to the same population.
- 3G access network plays an important role in deciding delay characteristics.
- The access network plays decisive factor for delay, but is not responsible for outliers.
- 3G delay exhibit clear diurnal patterns.

They find that the delay characteristics in different 3G networks are mainly network dependant rather that monitor dependant, and that each operator has its own “signature” in delay characteristic.

2.3.3 HSDPA Performance in Live Networks [17].

This paper compares the performance of WCDMA and HSDPA network from the aspect of end user experience. The measurements are done with HTTP and VoIP applications. The paper presents performance with TCP and UDP in WCDMA and HSDPA live operational networks. It compares measured delay and jitter values using one-way measurements which are important especially in case of wireless network where uplink and downlink are asymmetric. It shows examples how link level re-transmissions create bursts in packet delivery. Both WCDMA and HSDPA technologies were evaluated within the same operator in Finland. TCP goodput measurements were performed with the MOSET software developed at VTT Technical Research Centre of Finland [18], which uses HTTP GET messages to start a file download from a web server to a local laptop. UDP goodput measurements were performed with the QoSMeT tool [19, 20].

The findings of the paper are:

- As expected, HSDPA offers a clearly higher throughput and lower delay compared to WCDMA.
- The measured maximum goodput values correlate well with advertised performance values in HSDPA and WCDMA.
- We also found that link-level retransmissions in WCDMA create delay spikes of order of hundreds milliseconds.

Chapter 3: Measurement Methodology

In order to characterize the network's performance we need to understand the important parameters of the network, for example error rate, latency, throughput etc. Once we know important parameters, we need to understand how to measure them. This chapter attempts to answer these two questions. We then present a brief overview of the experimental setup and then give algorithm for various tests implemented to capture these properties and analyze them.

3.1 Measurement tests:

The measurement suite that conducts several tests at transport, network, and link layer helps us understand important properties of the network & how do they impact its performance. Following are the **proposed** set of tests which are expected to gather sufficient number of network parameters in order to characterize the network.

Application layer:

Tests at application layer, mentioned in Table 3.1.1, captures the parameters such as number of TCP connections created, how does the scheduling policies implemented by ISPs impacts the long lived TCP flows, detecting the presence of middle boxes in the network which may potentially impact the performance.

Table 3.1.1: Measurement tests at application layer.

Sr. No	Test Name	Description & Purpose
1	Application layer performance test *	Download pages from well known sites like Youtube, Facebook along with automated login procedure. To understand the application layer parameters like Number of TCP connections created impact of scheduling policies on long live TCP flows, etc. Understanding of such parameters will help us to improve the end user experience.
2	In network antivirus detection	Download well known EICAR file http://www.eicar.org/anti_virus_test_file.htm
3	Existence of http and dns aware middle boxes *	Send invalid http and dns packets on port 80/53 of the server. The server also responds with invalid http/dns packets. Middle boxes aware of these protocols will drop such packets. Detecting the presence of middle boxes in network will help us understand how they impact the performance.
4	Virus attacks over the network	Run tcpdump when running some of the tests. Look for spurious udp packets/tcp SYNs in the trace.

Transport Layer:

Tests at transport layer, mentioned in table 3.1.2, captures the properties such as TCP throughput in uplink and downlink direction, number of packet losses and retransmissions, long and short duration flow breaks, whether bunching effect is present or not etc.

Table 3.1.2: Measurement tests at transport layer.

No	**Name**	**Description**
1	Curl Download	To understand the network parameters in downlink direction perform a download of large file and collect the traces at both server and client end.
2	Iperf test	To understand the network parameters in uplink direction perform an iperf test, in which iperf client runs at measuring point(MP) and iperf server runs at Server.

Network layer:

Tests at network layer, mentioned in Table 3.1.3, captures the properties such as latency to gateway and other landmark nodes, DNS lookup times of local DNS servers to see if there is caching policy to improve the performance, traceroute to landmark nodes to understand the network map, finding network buffer size at gateway, presence of NAT, and support for IPv6.

Table 3.1.3: Measurement tests at network layer.

No	**Name**	**Description**
1	Ping to GGSN	To measure latency experienced by network because of the last hop.
2	Ping to landmark nodes	To measure latency variations among different landmark nodes which are geographically diverse and to check if they show diurnal patterns. To measure contribution of last hop (from test 1) on overall latencies observed (from test 2).
3	Lookup times at default DNS server(s)	To measure lookup times at default DNS servers. If found too high (specially in case of rural area) then check whether it will help if we use public DNS from Google, Dnsadvantage, OpenDNS, etc.
4	Traceroute to landmark nodes	To understand if the network infrastructure in case of Rural and Urban area different or not.
5	BFind (Buffer size at GGSN)	To estimate buffer size for the netbook at GGSN. Run ping to local DNS server for 1 minute. Now download a large file to the netbook with the ping still running. Do this for the second minute. Median RTT for 2^{nd} minute minus median RTT for 1^{st} minute is buffer size at GGSN.
6	Buffer size at netbook	Send several large UDP packets to sever at infinite data rate. The first packet dropped indicates the local buffer size.
7	Nmap[22]	It's a well known tool used to discover hosts and services on

		a computer network, thus creating a "map" of the network.
8	IPv6 adoption[23]	http://test-ipv6.com/
9	Existence of NAT and its type *	To check for the presence of NAT and understand the its type.

Physical layer:

Tests at physical layer capture the parameters such as signal strength, availability of the network, impact of state transitions in physical devices etc.

Table 3.1.4: Measurement tests at physical layer.

No	Name	Description
1	Signal strength and associated base station	To track correlation between signal strength/handoffs and results of all other tests.
2	Network availability test	Ping GGSN/Google once every 5 seconds using each of the three dongles on the netbook.
3	3G device state transitions	Algorithms picked up from a paper Characterizing Radio Resource Allocation for 3G Networks [24]

Note: '*' marked tests have not been implemented.

3.2 Experimental Setup:

The measurement suite will be deployed on netbook's which will be shipped to different locations across the country. The netbook will measure characteristics of GPRS, HSPDA, and EVDO access technologies provided by Airtel, MTNL, and Reliance respectively.

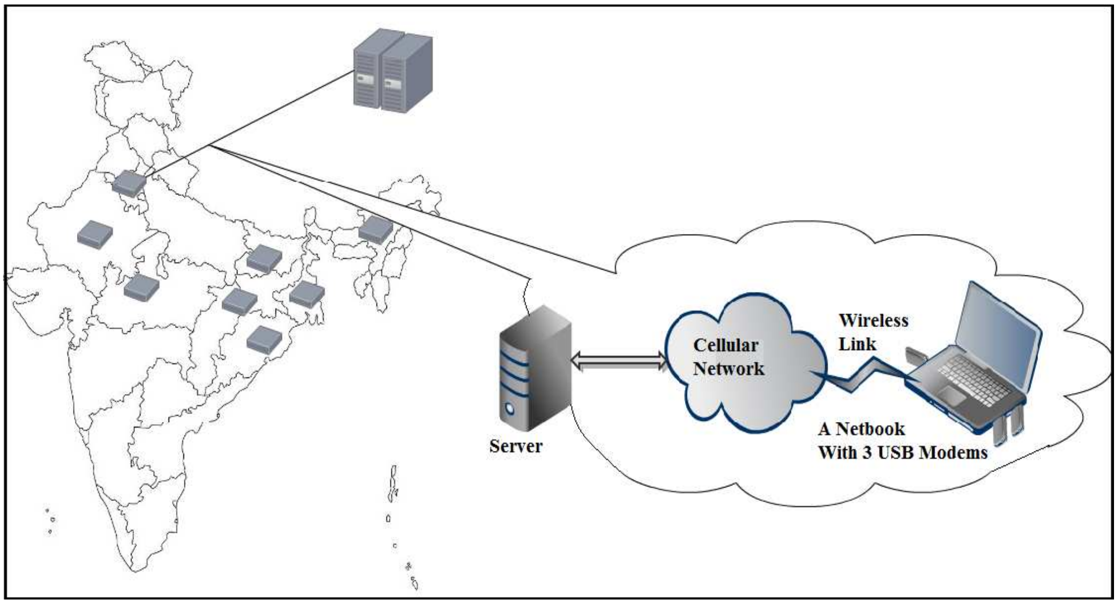

Figure 3.2.1: Overview of experimental setup.

Work Flow:

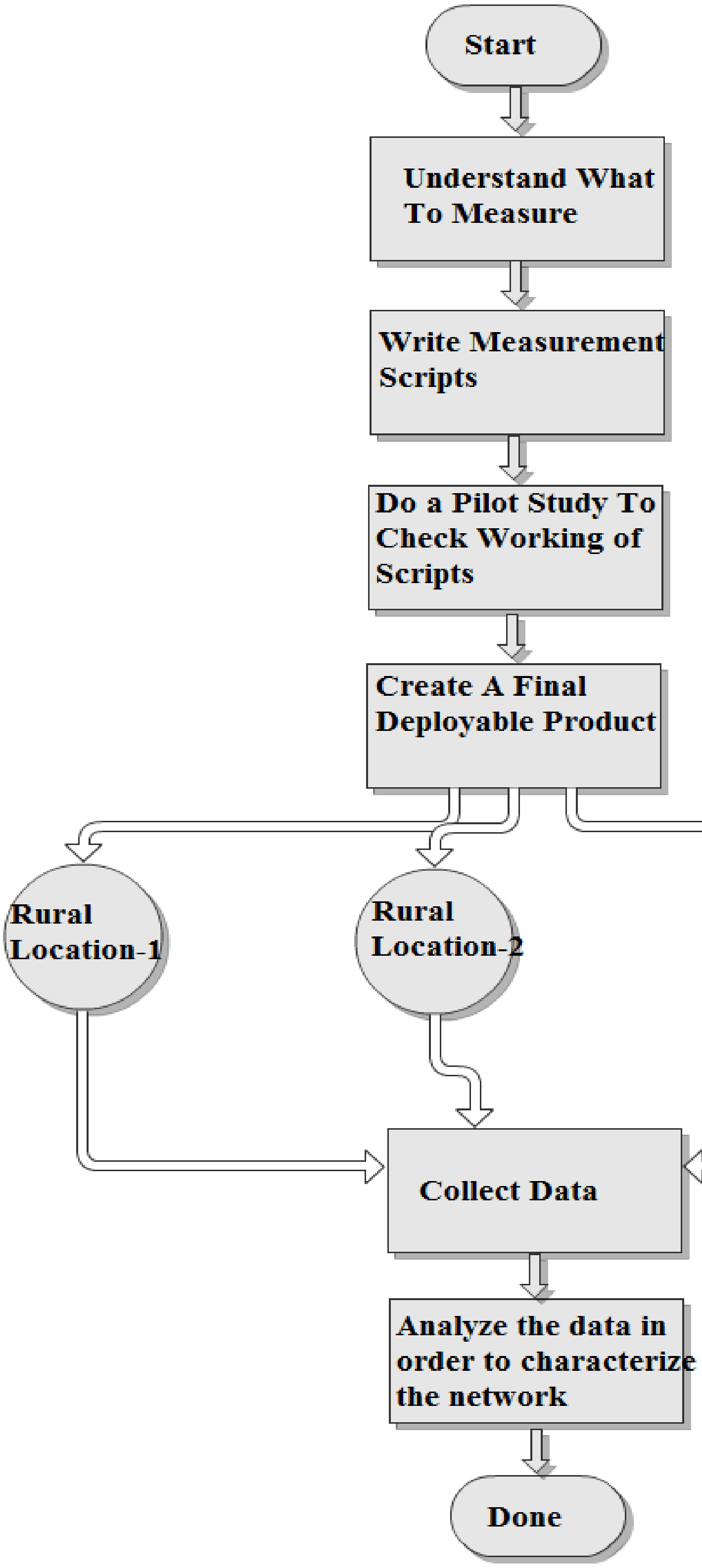

Figure 3.2.2 Flow of work.

3.3 Development of test suite and analysis scripts:

After getting the idea of how to measure the network parameters, the next step was to develop these tests. Mentioned below are the algorithms/steps for some of the tests we which have implemented in our test suite.

a) CURL Test: (Downlink test)
   Download a file from server with tcpdumps running at both ends.
   Steps:
   1. Start tcpdump at server.
   2. Start tcpdump at client.
   3. Start downloading of file.
   4. Kill tcpdump at server.
   5. Kill tcpdump at client.
   6. Collect tcpdump trace files at client.
   7. Done.

b) Iperf Test: (Uplink test)

   Steps:
   1. Start iperf server on server.
   2. Start tcpdump on server.
   3. Start tcodump on client.
   4. Start iperf client on MP.
   5. Kill iperf server & tcpdump on server.
   6. Kill tcpdump at client.
   7. Collect tcpdump trace files at client.
   8. Done.

c) Ping to gateway:
   Steps:
   1. Find the gateway.
      a. Use route command. or
      b. First IP seen when pinging to any well known site(like www.google.com)
   2. Run ping to gateway for predefined amount (conf file) of time and log the output.

d) Ping to landmark nodes:

   1. Make a list of landmark nodes and store them in a file (landmarkNodes.list)
      The landmark nodes used for the tests are:
      www.google.co.in, www.uwaterloo.ca, www.timesofindia.co.in, web.mit.edu, etc.

2. For each entry in landmarkNodes.list file
    a. Run a ping for predefined amount of time (specified from conf file).
    b. Log the output of the test.
3. Done.

e) Find DNS & its lookup time:
   Steps:
   1. Find the list of DNS servers (from /etc/resolv.conf file).
      In Linux, modem manager populates this file with DNS server information when DHCP is enabled.
   2. Using a tool named *Dig* [25] find the lookup time of these servers.

f) Traceroute to landmark nodes:
   Steps:
   1. For each entry in landmarkNodes.list file
      a. Run a *traceroute*[26] command.
         Traceroute command finds all the IP layer devices along the path from source to destination.
      b. Log the output of the test.

g) UDP uplink test:
   Steps:
   1. Run an UDP server at server.
   2. Start tcpdump collecting UDP at server.
   3. Start tcpdump collecting UDP at client.
   4. Run UDP client at MP providing parameters such as number of packets to send, the rate at which they should be sent & size of UDP packets.
   5. Kill UDP server and tcpdump running at server.
   6. Collect the dump files at client.
   7. Done.

h) UDP downlink test:
   Steps:
   1. Run a UDP server at client
   2. Start tcpdump collecting UDP at server.
   3. Start tcpdump collecting UDP at client.
   4. Run UDP client at Server providing parameters such as number of packets to send, the rate at which they should be sent & size of UDP packets.
   5. Kill UDP server and tcpdump running at server & client.
   6. Collect the dump files at client.
   7. Done.

i) BFind (Measure the buffer size):

   Steps:

   1. Run ping to local DNS server for 1 minute.
   2. For second minute, download a large file to the Netbook with the ping still running in background.
   3. Kill the download at the end of second minute.
   4. Median RTT for 2^{nd} minute minus median RTT for 1^{st} minute is buffer size at GGSN

j) NAT existence:

   Steps:

   1. Note local IP address as seen by the netbook.
   2. Send a UDP packet from client to server and note the “from” address at the server.
   3. If the two addresses are different a NAT is present.
   4. Send multiple such packets over 10s of seconds.
   5. If the from address alternates then NAT does not use a single IP for a single internal IP.

k) Analysis scripts:

   The tests usually run for a very long duration like few day or few weeks. So there is possibility that some of the traces collected will be incomplete, for reasons like unavailability of network. These traces must be removed from the data for further analysis. A script “collectUsefulData.sh” has been written which does this job and makes a list of correct traces.

   Steps:

   1. Latency analysis:
      For Gateway and Landmark Nodes over all the correct traces:
      i. Collect Stats: Find the Average, Min, Max RTT, standard deviation of RTT for all the correct traces.
      ii. Plot RTT graphs.
   2. Throughput, retransmissions analysis
      For all the correct traces:
      i. Scan all the dump files using tool named *tcptrace* [27].
      ii. Extract the throughput, unique bytes sent, number of retransmissions.
      iii. Find the average throughput & its standard deviation.
   3. Buffer size analysis:
      i. Implement the BFind algorithm stated in (i)
      ii. Find the average Buffer size and its standard deviation.
      iii. Plot buffer size for different time of day to check for its consistency.
   4. Traceroute data analysis:
      i. Compare the traceroutes for landmark nodes in urban and rural areas.

Chapter 4: Tools & Experiments

In this chapter we discuss various experiments performed and tools used for these experiments. The result of these experiments & their analysis is presented in next chapter.

4.1 Tools used for experiment:

1. Curl: [28]

   Curl is a client program which is used for transferring files between client and server. It supports various protocols such as HTTP, HTTPS, FTP, GOPHER, DICT, TELNET, LDAP etc. This tool was used to download a file from server to client generating traffic in downlink direction.

2. Iperf:[29]

   Iperf is a tool which can generate both UDP and TCP traffic for testing bandwidth, latency and packet loss. It has a two main components iperf client and iperf server. Iperf client generates a traffic which is consumed by iperf server. The tool also reports the bandwidth between the network paths. In our experiment iperf server was running on server and iperf client was running on measurement point, generating traffic from client to server i.e. in uplink direction.

   Reason for using two different tools for uplink and downlink direction is, for some service provider like Airtel it is not possible to set up a direct connection from outside world hence the connection must be initiated from inside. The reason being that hosts are allocated private IP address.

3. Dig: [25]

   This command is used to query DNS name servers. It performs DNS lookup and displays answers that are returned from the name server to which the query was sent along with query time.

4. Tcpdump:[30]

   Tcpdump is a common packet analyzer. It captures the packet & their details on specified interface. It supports very powerful filtering mechanism. This tool is used to collect the TCP & UDP packet traces at both client and server side.

5. Tcptrace:[27]

   This tool takes as input the files produced by several popular packet-capture programs, including tcpdump, snoop, etherpeek, HP Net Metrix, and WinDump.
   Tcptrace can produce several different types of output containing information on each connection seen, such as elapsed time, bytes and segments sent and recieved, retransmissions, round trip times, window advertisements, throughput, and more.

4.2 Experiment Specifications:

This section describes the technical details about the experimental setup.

Measurement Point: Ausus Netbook Eee PC .
Servers Used:

1. Linode, Japan: A Virtual machine.
2. Glenstrom, IIT Delhi: A Virtual machine.
3. Vayu, IIT Delhi: A Dedicated machine.

Operating System: Ubuntu 11.04, Kernel version: 2.6.38-8-generic.

Service providers & USB Modems:

Table 4.2.1: Service providers, their access technology and device information

Connection	Access technology	USB Modem
Airtel	GPRS/ EDGE	Huawei E1731
Reliance	EV-DO	ZTE AC2737
MTNL	HSPA	Huwai E173

Default TCP congestion avoidance algorithm: TCP Cubic.

Location:

1. Urban: IIT Delhi.
2. Rural: Dholpur, Rajasthan.

4.3 Data sets:

In this section we present the summary of how different data sets were obtained from the execution of set of tests mentioned in chapter 3.

4.3.1 Data Set From Rural Area:

As mentioned in the previous section we have performed pilot study at rural location in Dholpur, Rajasthan. Table 4.4.1 lists the set of tests performed and their status with respect to different service providers. All the tests in Dholpur were performed with Linode server.

- Dholpur Data Set:

  Reliance network is not present in this rural location hence none of the tests worked for Reliance EVDO connection.

Table 4.3.1.1 Dholpur Data Set

Sr. No	Test	Is test performed?		
		Airtel GPRS	Reliance EVDO	MTNL HSDPA
1	Curl Test	✓	×	✓
2	Iperf Test	✓	×	✓
3	Ping to gateway	✓	×	✓
4	Ping to landmark nodes	✓	×	✓
5.	Traceroute to landmark nodes	✓	×	✓
6.	DNS Lookup latency test	✓	×	✓
7.	Test to understand impact of TSO	×	×	×
8.	Test to understand impact of initial window	×	×	×
9.	Test to understand long flow breaks.	×	×	×
10.	Network buffer size at gateway	✓	×	✓
11.	Availability	✓	×	✓
12.	Signal strength test	✓	×	✓
13	Existence of middle boxes	×	×	×
14.	TCP Reno & TCP Cubic comparison	×	×	×
15	Test to understand bursty behaviour of Cubic	×	×	×

4.3.2 Data Sets From Urban Area:

Tests such as ping to gateway, landmark nodes; traceroute to landmark nodes; DNS lookup latency test etc are all independent of server. All these tests were also performed in urban area. We categorize data obtained from rest of the tests into two sets based on the server used.

- IIT Delhi Server Data Set:

Table 4.3.2.1 IIT Delhi data set.

Sr. No	Test	Is test performed?		
		Airtel GPRS	Reliance EVDO	MTNL HSDPA
1.	Curl test	✓	✓	✓
2.	Iperf test	✓	✓	✓
3.	Test to understand impact of TSO	×	×	×
4.	Test to understand impact of initial window	×	×	×
5.	Test to understand long flow breaks.	×	×	×
6.	TCP Reno & TCP Cubic comparison	×	×	×
7.	Test to understand bursty behaviour of Cubic	×	×	×

- Linode Server Data Set:

Table 4.3.2.2 Linode data set.

Sr. No	Test	Is test performed? Airtel GPRS	Reliance EVDO	MTNL HSDPA	Airtel HSDPA
1.	Curl test	✓	✓	✓	✓
2.	Iperf test	✓	✓	✓	✓
3.	Test to understand impact of TSO	✓	×	×	✓
4.	Test to understand impact of initial window	✓	×	×	×
5.	Test to understand long flow breaks.	✓	×	×	✓
6.	TCP Reno & TCP Cubic comparison	✓	×	×	×
7.	Test to understand bursty behaviour of Cubic	✓	×	×	✓
8.	Long duration curl download	✓	×	×	✓

4.4 Experiments performed:

Apart from the execution of measurement suite, we have performed some other experiments. These experiments were conducted to understand the network in greater depth. This section describes these experiments in more details.

4.4.1 Long duration curl downloads to measure downlink characteristics:

This experiment was performed to understand the impact of time of day on TCP with Airtel GPRS connection. Here we connected a netbook with Airtel GPRS connection and downloaded a file of 5MB using curl each hour for two days. The same test was repeated with Airtel HSDPA connection.
Location: IIT Delhi.
Server: Linode, Japan.
This test also brought out some other interesting points like long duration flow breaks resulting from exponential increase in RTO, some traces showed acknowledgement bunching.

4.4.2 A Test to understand the bursty behaviour of TCP Cubic.

To understand the impact on TCP flow of transition between three phases of TCP Cubic namely Concave phase, TCP friendly phase and Convex phase, an experiment was performed where we collected internal parameters of TCP connection namely slow start threshold and current congestion window. To log the congestion window we used a tool *tcp_probe* developed by *Hemminger*.

4.4.3 A Test to compare the performance of TCP cubic and TCP new Reno

Bursty nature of Cubic in Cellular network (especially GPRS) where BDP is very less, is causing packet losses and hence retransmissions which affects throughput. So it was necessary to compare the performance of Cubic with standard TCP (TCP Reno).

Experiment:

1. Make TCP Cubic as default congestion avoidance algorithm on server. This can be easily done by setting of parameter net.ipv4.tcp_congestion_control.
2. Connect netbook with GPRS connection and download a file from server an collect tcpdumps at both ends.
3. Change the default congestion algorithm on server from Cubic to TCP Reno.
4. Download a file from server. Collect tcpdumps at both ends.
5. Repeat steps 1 to 4 for different sizes of Initial congestion window.

Chapter 5: TCP Dynamics

Some of our early attempts in understanding the network characteristics brought in front few interesting observations such as server sending large segments of data irrespective of the value of MTU, some of the time-sequence number plot representing TCP flow showed breaks in the middle, server always start a TCP flow by sending three large segments etc. Our further investigating into these observations lead us to parameters such as TSO: an optimization technique used by servers, Initial window size being set to large value, not following the standard stated in RFC3390. In this chapter we describe how these TCP specific parameters impact the performance.

5.1 Experiments

In this section we describe various experiments performed to understand the impact of the parameters like TCP segmentation offloading, impact of initial window size and few experiments performed to understand the reason for long duration flow breaks.

5.1.1 Test to understand the Impact of Initial window size on TCP Cubic

There has been a lot of debate going on whether congestion window size should be 3 or larger. A recent study from[1] shows that IW should be at least 10 as the average bandwidth in today's network is around 1.2 Mbps. Also Ubuntu kernels after 2.6.39 have by default[2] set their initial congestion window size to 10. Same can be seen from the trace files we have collected. What essentially happens when you increase the initial window size is it makes the traffic bursty.

Controlling Initial Window Size in Ubuntu:

Step1: run "ip route" and note the line containing default in its output.

Eg:

amitsingh@li378-87:~$ ip route

default via 106.187.35.1 dev eth0 metric 100

106.187.35.0/24 dev eth0 proto kernel scope link src 106.187.35.87

Step 2: add "initcwnd " parameter to it.

Eg:

ip route change default via 106.187.35.1 dev eth0 metric 100 initcwnd 20

1 http://monolight.cc/2010/12/increasing-tcp-initial-congestion-window/

2 http://kernelnewbies.org/Linux_2_6_39#head-d11935223b203d28a660417627514973de4e218

Experiment:

To understand the impact of Initial window size on TCP throughput we performed a test where:

For Initial Window Size of 4, 10 and 30

1. Download a file from server
2. Collect server and client side traces
3. Note the parameters like throughput and number of retransmissions.

The results of this test are presented in next chapter.

Further Work:

This test also has been performed only for the Airtel GPRS connection. It should also be done for other service providers.

5.1.2 Long duration test to check the impact of TCP segmentation offloading (TSO)

TSO is a technique for increasing the outbound throughput of high-bandwidth connection by reducing CPU overhead. Segmentation is often done by TCP protocol in host computer, offloading the task of segmentation to NIC is called TCP segmentation offload. The following figures explain the working of TSO.

When TSO is disabled: transport layer will do the job of segmenting the application layer data.

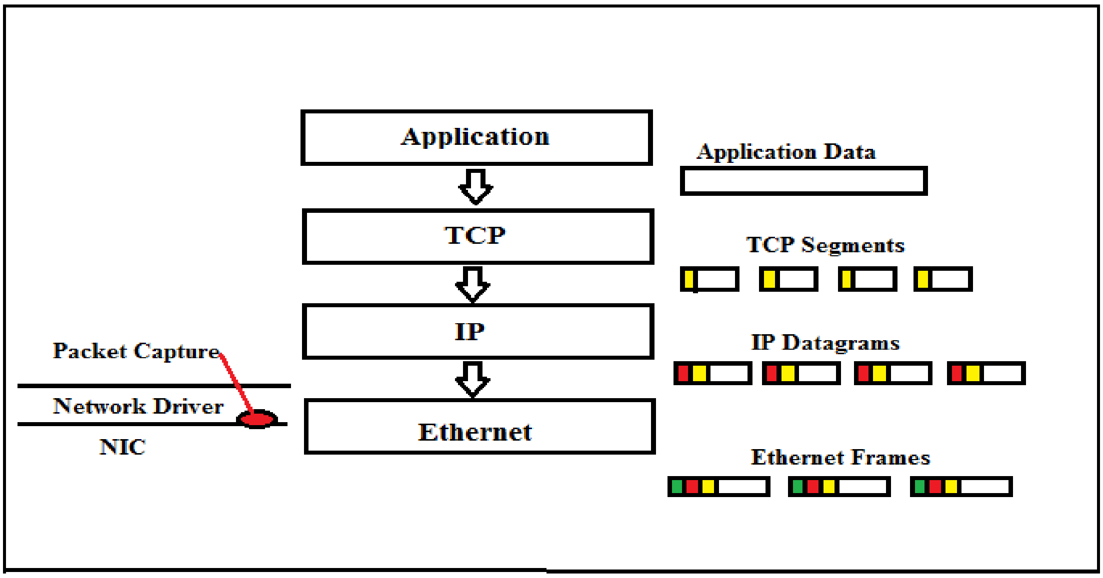

Figure 5.1.2.1: Segmentation when TSO is disabled.

When TSO is enabled: transport layer will not perform the segmentation and will offload this task to NIC.

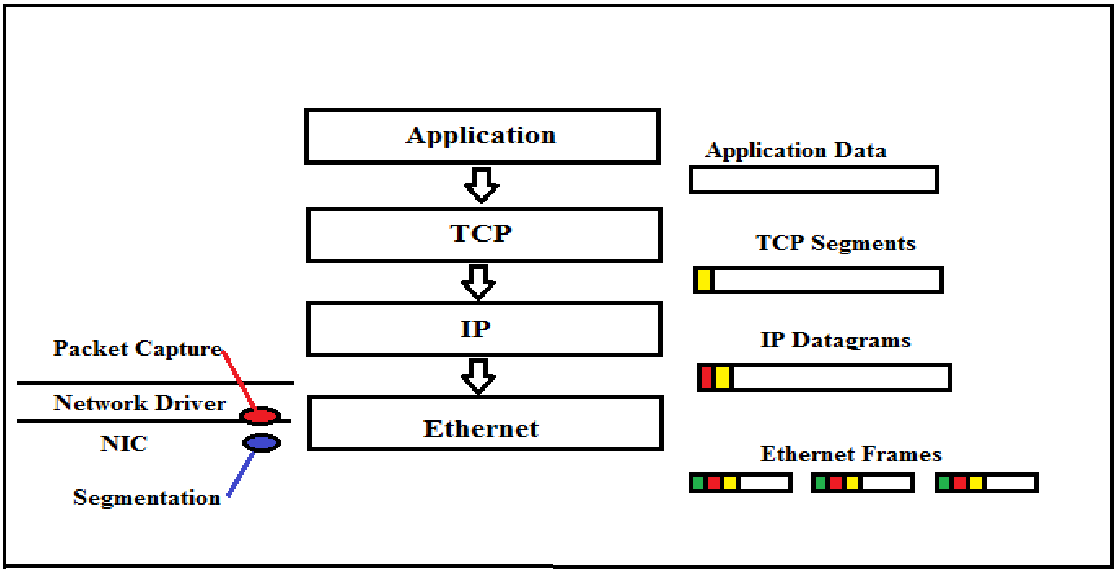

Figure 5.1.2.2: Segmentation when TSO is enabled.

Controlling the TSO in Linux:

1. TSO can be disabled or enabled in Linux with tool *ethtool*. By default it's enabled.
2. A parameter named tcp_tso_win_divisor controls what percent of congestion window can be consumed by a single TCP segmentation offload (TSO) frame.

The setting of this parameter is a trade-off between burstiness and building larger TSO frames.

Test:

1. Test to check impact of TSO on TCP throughput:

To check whether TSO impacts the TCP performance or not we performed an experiment in which

- Downloaded a file with TSO enabled for different values of tcp_tso_win_divisor. Note the TCP throughput and number of retransmissions.
- Downloaded a file with TSO disabled. Note the TCP throughput and number of retransmissions.

2. Test to check impact of TSO on server CPU load:

To check how disabling TSO affects the CPU load at server another experiment was performed as:

Generate heavy load at server for which we started parallel download from 10 computers each with two TCP connections downloading same file.

- Download a file with TSO enabled & note the CPU utilization at server.
- Downloaded a file with TSO disabled & note the CPU utilization at server.

Results obtained from this experiment are explained in the next section.

Further Work:

Currently this test has been performed only for Airtel GPRS connection. To gain more understanding about how presence of TSO impacts throughput and number of retransmissions it needs to be repeated with other access technologies like EVDO and HSDPA.

5.1.3 Parallel UDP with curl download to understand long duration flow breaks.

Long duration test described in previous chapter also brought in front another interesting observation that TCP flows suffer from a very long duration flow breaks with exponential increase in retransmission timeout period. The duration of these breaks ranged from ~5 seconds to ~260 seconds. Frequency of such long flow breaks is more at peak time but they also have been noticed at other time of day. The following sequence number diagram shows the presence of such breaks.

Client side view:

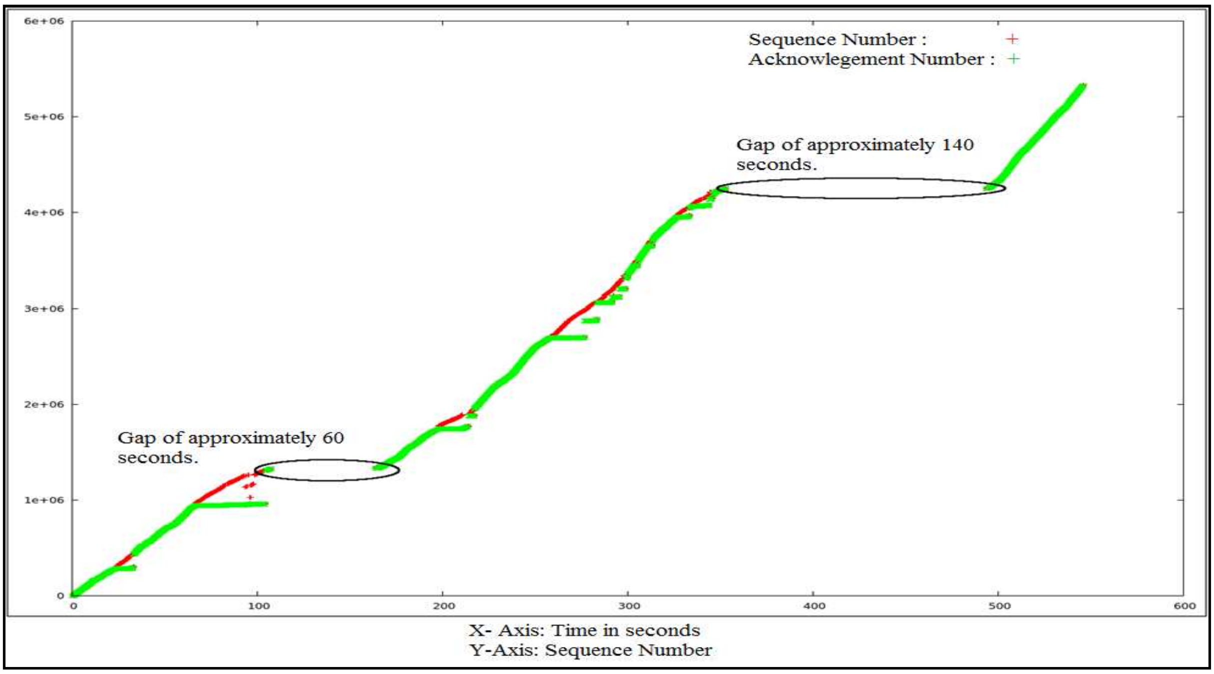

Figure 5.1.3.1: Client side view showing long duration flow breaks.

Server side view:

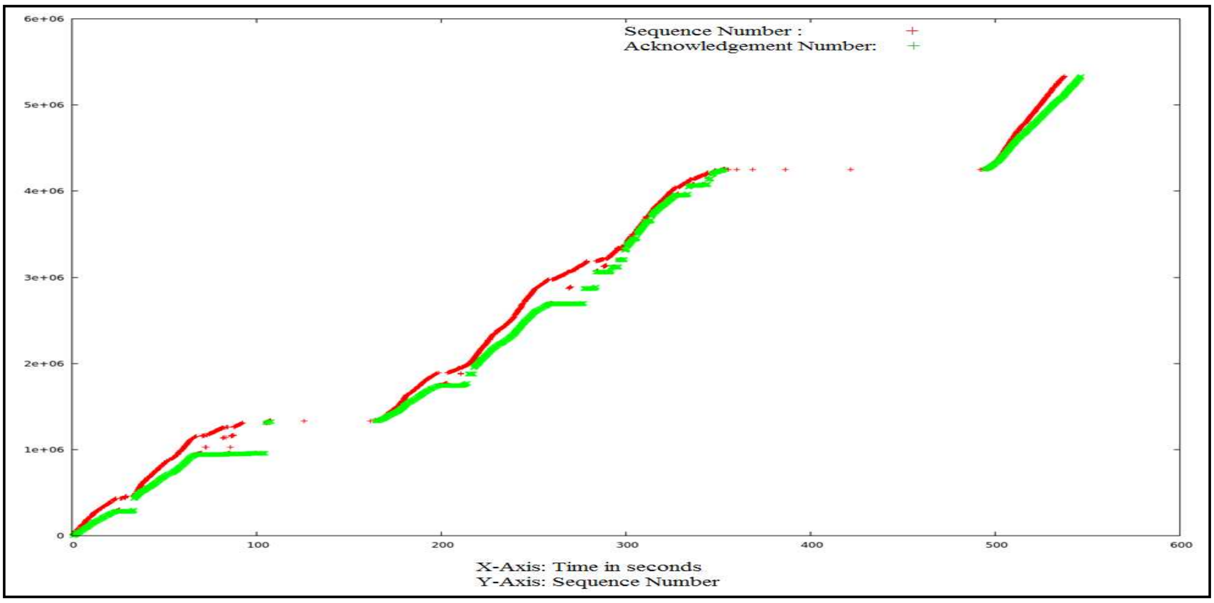

Figure 5.1.3.2: Server side view showing long duration flow breaks.

Closer look at server side trace (graph) shows exponential increase in RTO as

2 sec➔ 4.5 sec➔7 sec➔ 17 sec➔ 35 sec ➔ 71 sec

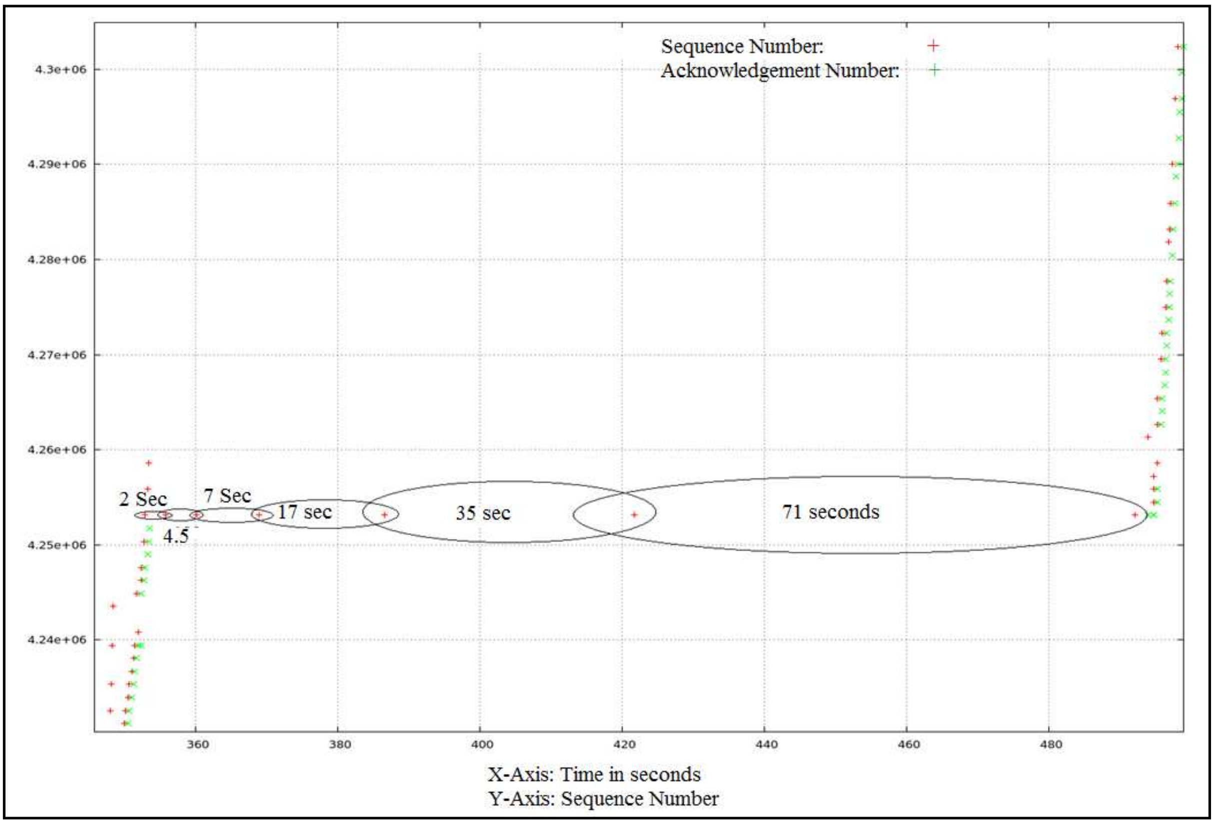

Figure 5.1.3.3: Server side view showing exponential increase in timeout.

Few tests were conducted to understand why & when do these long flow breaks occur in network:

Test1:

To check if it was signal strength at client that went down and hence caused these long duration flow breaks, download a file and also collect the signal strength information at client.

Test 2:

To check if this problem is because of some wireless property (PHY/MAC layer issue) or it is due to something in the TCP protocol, we started a low bit rate UDP flow running in parallel. If UDP packets keep getting through when TCP is stuck then it will suggest that problem is not because of lower level protocol.

Further Work:

To confirm if all the flows in the network suffer from this problem we can perform another test where we start a long duration download from two different measuring points in the same network, possibly residing within same cell & downloading file in parallel from same server.

5.2 Results

5.2.1 Impact of Initial window size:

A recent study from [32], claim that initial window size for TCP connection should be atleast 10, because average bandwidth in today's network is around 1.2Mbps. Also Ubuntu kernels after 2.6.39 come with default initial window size to 10. We suspect that increasing the IW size is not a good idea when working with cellular network as it makes the traffic bursty & hence increases the chances packet loss with large number of retransmissions.

Validation:

For validation we performed experiment as described in section 4.4.4, with Airtel GPRS: high latency low bandwidth network. This test was performed with different file sizes of 80kB, 400 kB and5 MB.

Findings:

Results from experiment, table 5.2.1, shows that if your initial window size is large then it results in large number of packets being retransmitted.

Table 5.2.1 Impact of Initial Window Size

File Size	IW = 3		IW = 10	
	Throughput (in KBps)	Number of Retransmissions	Throughput (in KBps)	Number of Retransmissions
80KB	5.3 +/- 1.3	3.6 +/- 3.3	13.3 +/- 4.4	2.4 +/- 4
400KB	20.6+/-2.5	6.2+/- 5.2	18.06+/-4.8	10+/- 9.0
5MB	17.4 +/- 2.0	282.5 +/-74.8	15.0+/-7.5	354+/-108

Increasing the initial window size would give you better performance only if the network is able to handle the bursty behaviour. If the router has a bias against the bursty traffic then it is better to start with lower IW size. There is a trade-off between large initial window size and bursty behaviour; if you increase IW then it also increases bursty behaviour, which potentially may result in large number of retransmissions.

5.2.2 Impact of TSO:

TSO is an optimization technique used by server where the task of segmenting the application layer data is offloaded to NIC. In the process of this optimization server ends up by sending a large burst of packets into the network which then impacts the performance.

Validation:

To find how severely does this optimization technique impacts the bursty behaviour we conducted some experiments as explained in chapter 4, section 4.3.2.

Findings:

1) The results from the experiment showed that even though there was reduction in average throughput after disabling TSO, it was only 6% compare to throughput observed when TSO was enabled. But the reduction in number of retransmission after disabling TSO was 20.3% compared to when TSO was enabled. The reason behind the increase of number of retransmissions when TSO was enabled is, TSO increases the bursty behaviour of TCP algorithm.

2) Second test measured the impact of TSO on CPU utilization. It was found that disabling TSO increases the CPU utilization. Following graphs shows the impact on CPU utilization:

CPU Utilization when TSO was enabled: only once CPU utilization was more that 4%.

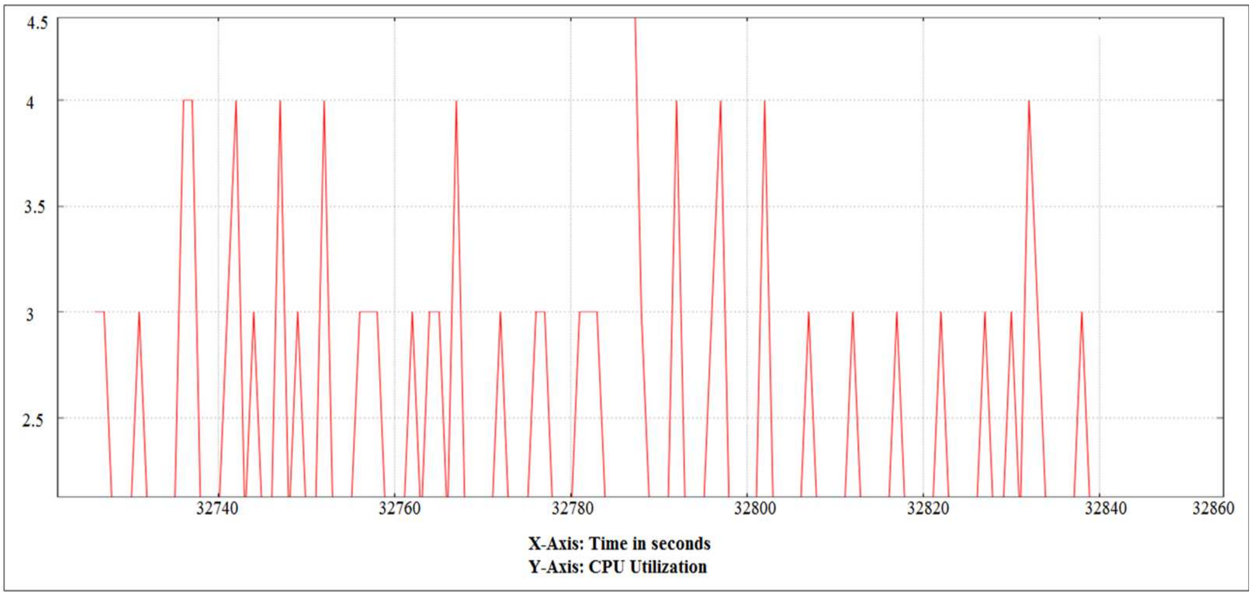

Figure 5.2.2.1: CPU utilization when TSO was enabled.

CPU Utilization when TSO was disabled: more than 7 times CPU utilization was greater than 4.5 %

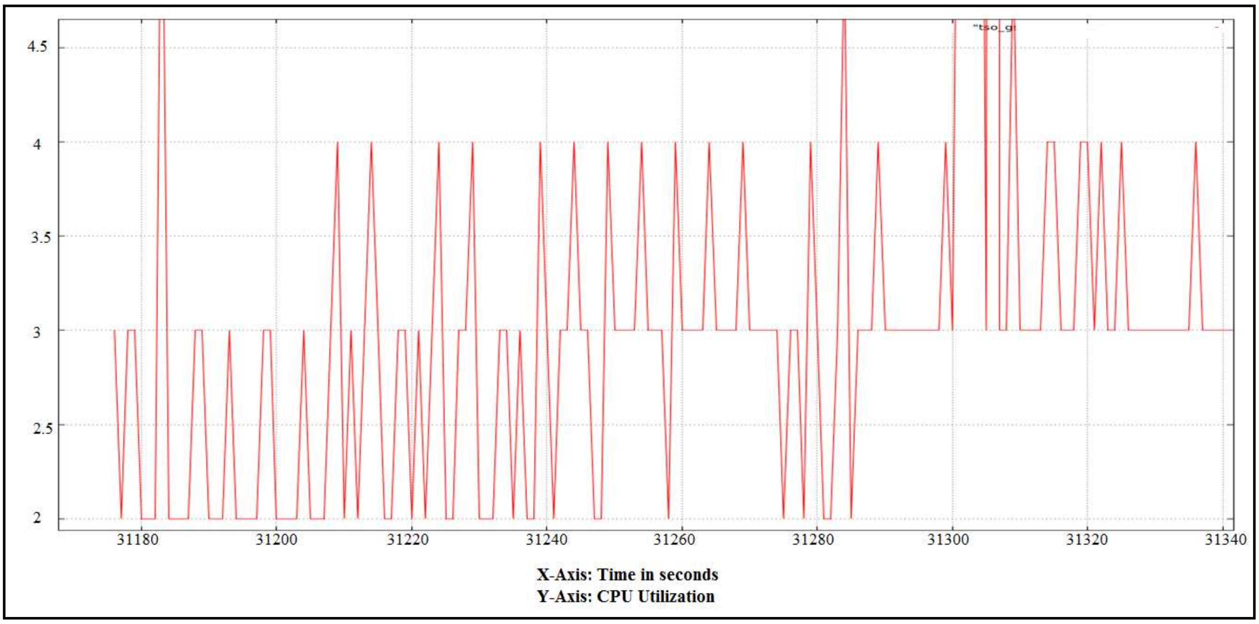

Figure 5.2.2.2: CPU utilization when TSO was disabled.

Explanation:

Following graphs explain why TSO increases bursty behaviour of TCP:

When TSO was disabled server sends out the data in response to every acknowledgement it receives.

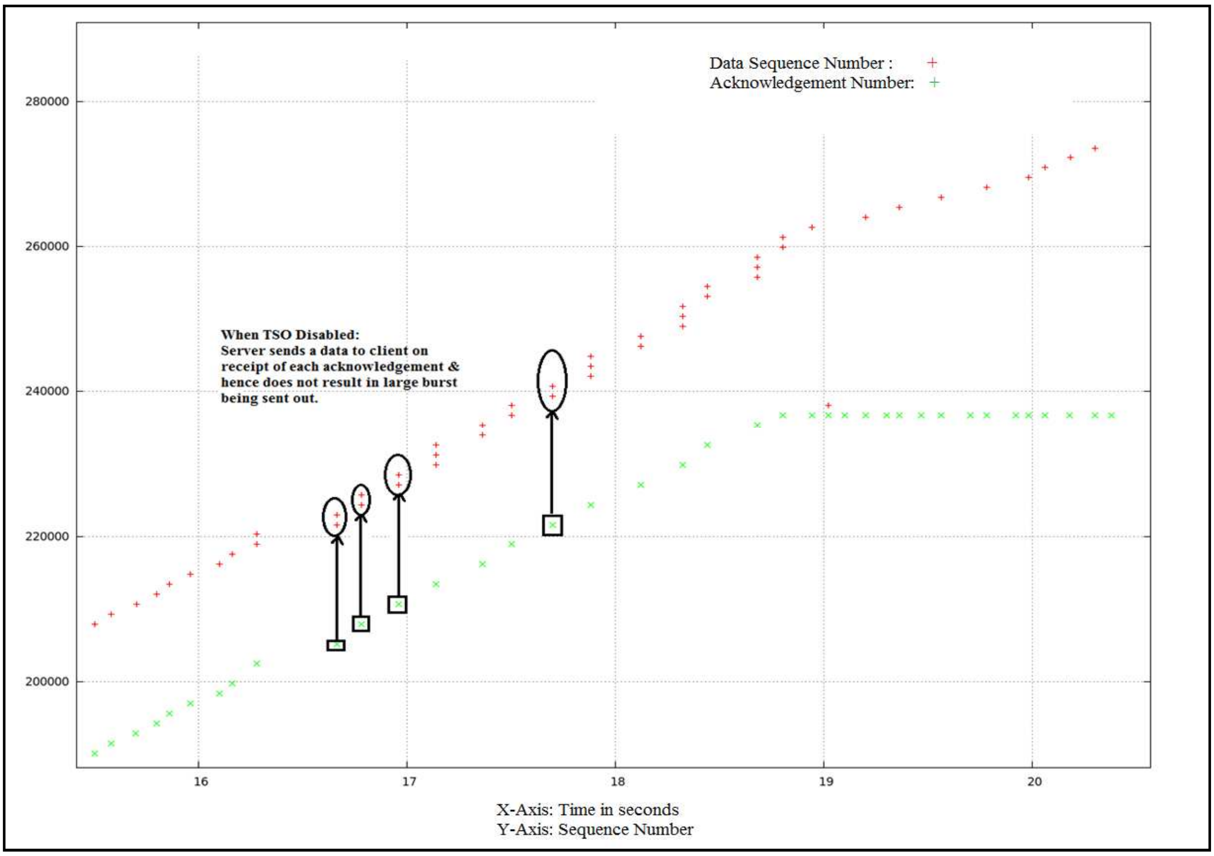

Figure 5.2.2.3: TCP flow when TSO was disabled.

When TSO is enabled server does not send data out in response to every ack it receives. It accumulate the data and then sends a large segment out which then results in burst of packets being sent out. This behaviour is as shown in following figure.

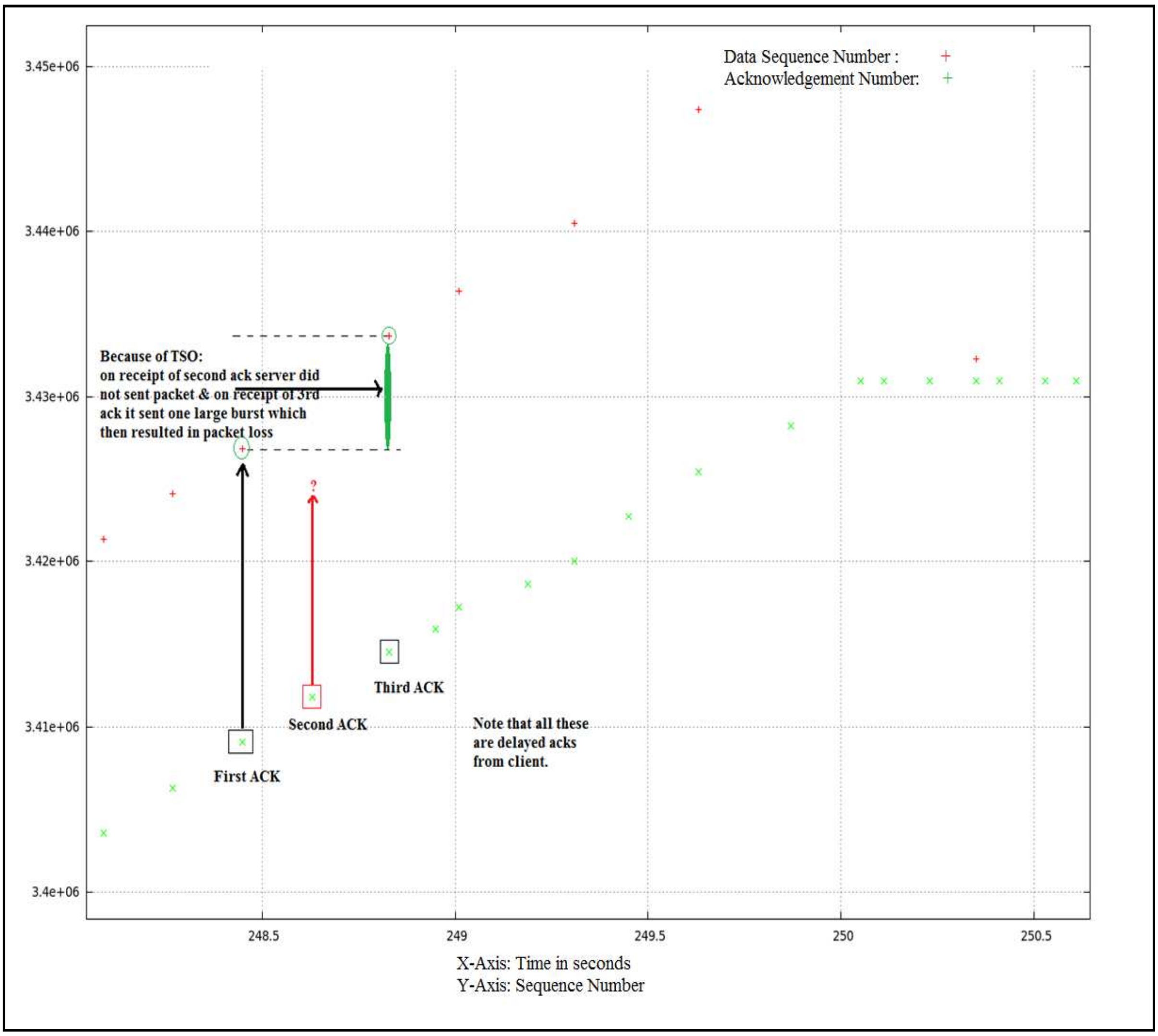

Figure 5.2.2.4: TCP flow when TSO was enabled.

Impact:

When using TSO there is a trade-off between the CPU utilization and bursty behaviour caused. The advantage of segmentation offloading comes in how much data was DMA'ed across the bus and how much the CPU was unburdened by segmentation task. But at the same time larger the data offloaded more is the burst size produced. This combination of TSO and delayed acknowledgements is responsible for generating large bursts which cannot be handled by the network with smaller buffer sizes & hence it results in large number of retransmissions.

Chapter 6: Results and analysis

Apart from the TCP specific tests and their results discussed in the previous chapter there are other results obtained from various other tests, some of which are discussed in chapter 4. This chapter summarizes the results obtained from these tests and their analysis.

6.1 Availability of connections in Rural and Urban area:

From end users perspective it is important to know how available are the services provided by different service providers.

Validation:
During our pilot test in rural area we noticed a strange diurnal pattern: both Airtel and MTNL connections got disconnected in the evening and reconnected in the morning. We noticed this for several days, Airtel: 24 days and MTNL: 6 days. We suspect that the base stations in the rural regions were operating on alternate sources of energy (diesel/solar), and were conserving energy by turning off or significantly reducing power of data interfaces at night.

In urban area the connections did not show such strange diurnal patterns in term of availability but the stability of MTNL connection was very less compare to that of Airtel and Reliance network.

Stability of connection = {Successful runs of the test without any disconnection} / {Total number of times test was conducted}

In urban area stability of Airtel EDGE connection was 91.13%, for Airtel HSDPA it was 99.19%, for Reliance EV-DO it was 86.84% and for MTNL HSDPA it was 62.6%. A large number of disconnections were experienced with MTNL connection.

Impact:
Unavailability of network services at night time in rural area severally degrades the user experience. Similarly frequent disconnections observed with MTNL connection in urban area also impacts real time application such as VOIP.

6.2 Data rate in cellular network:

From end users perspective, it is also important to know, whether the actual throughput achieved over cellular network matches the data rates advertised by the service providers.

Validation:
A week long curl test was conducted in urban and rural area at different time of day. Table 6.2.1 shows the comparison of average data rate observed in the network versus the advertised data rates.

Table 6.2.1: Comparison of achievable throughput and advertised

Connection	Achievable Throughput (in Kilo bits per second)						Advertised data rate (in Kilo bits per second)	
	Downlink			Uplink			Downlink	Uplink
	Average	Min	Max	Average	Min	Max		
Airtel Edge	64.9 +/- 42.8	12	200	68.6 +/- 20.6	22.6	100.2	20 to 30[1]	20 to 30
Reliance EVDO	1001.3 +/- 536.8	403.2	2206.4	611.5 +/- 131.5	80.8	786.72	upto 3100[2]	Upto 1800
MTNL HSDPA	757.6 +/- 274.4	331.2	1536.8	284.4 +/- 133.44	6.64	375.36	Upto 3600	

Surprisingly, in case of Airtel network the actual throughput is very large than advertised data rate. The average downlink throughput achieved was only 32.3% of advertised downlink data rates in case of Reliance EVDO connection and 21.04% in case of MTNL HSDPA connection. Even maximum downlink throughput seen was only 42.69% of advertised downlink data rate for MTNL HSDPA connection and 71 % for Reliance EVDO network.

Impact:
These observations show that achievable throughput are very less than advertised data rates in both uplink and downlink direction for MTNL HSDPA and Reliance EDVO network, which ultimately impacts the end user experience & degrades the quality of service.

6.3 Diurnal patterns in Throughput.

It is also important to understand how the time of day impacts the TCP throughput and to see whether diurnal patterns are present or not.

Validation:
To capture the long term diurnal stability of link bandwidth, weeklong test in urban area was conducted. Test running every hour captured the diurnal pattern in downlink throughput for all three connections. Uplink bandwidths are less than downlink bandwidth in Reliance and MTNL connection with exception of Airtel connection. Figure 6.3.1, 6.3.2 and 6.3.3 shows the diurnal variation in bandwidth for these three connections in downlink direction.

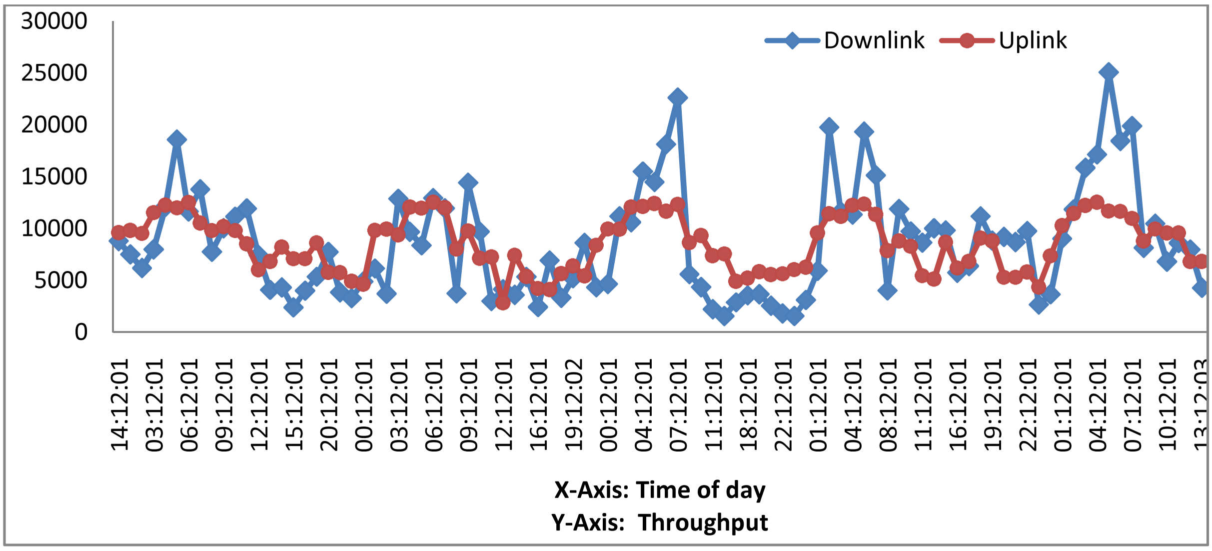

Figure 6.3.1: Throughput variation in Airtel GPRS connection

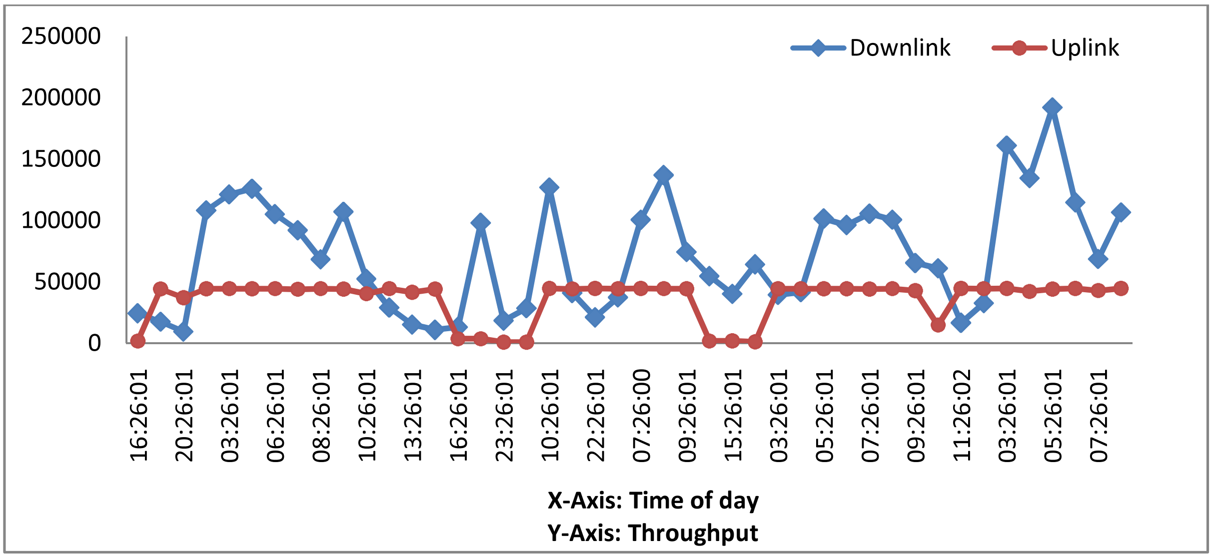

Figure 6.3.2: Throughput variation in Reliance EVDO connection

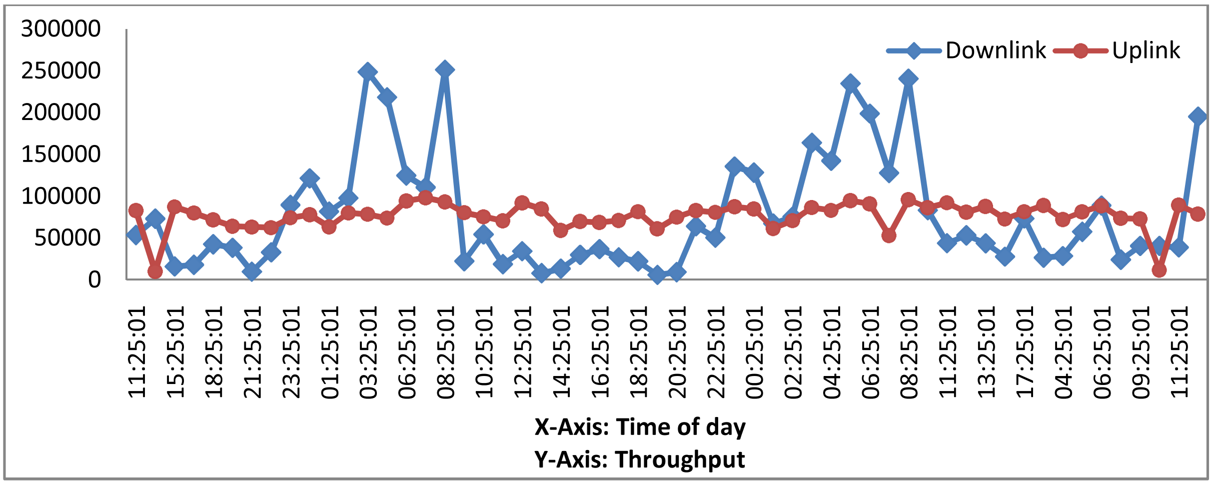

Figure 6.3.3: Throughput variation in MTNL HSDPA connection

The downlink throughput achieved in early morning is very high compare to that achieved during rest of the day. In case of Airtel connection, reduction in downlink throughput achieved during the day compared to that achieved in early morning (1.00 AM to 7.00 AM) is 71.57%, for Reliance reduction was 35.43% and for MTNL reduction was 33.56%. These findings about variation of bandwidth in downlink direction confirm the popular idea that competing traffic affects the bandwidth of cellular host.

Impact and Suggestion:
Presence of these diurnal patterns can be exploited as:

From user's perspective: Delay tolerant application, such as sending email, can be used to send data at night time when high throughput is achieved.

From service provider's perspective: They can provide incentive to their customers for using internet at night time when throughput is very high.

6.4 Last hop is bottleneck on network path.

The ping test described in section 3.4 (c) and (d), reveals that RTT to Gateway is major contributor of the delay experience over network path to landmark nodes. Table 6.4.1 shows the average of RTTs observed over different landmark nodes over three different connections.

Urban Data:

Table 6.4.1: Latency to landmark nodes Urban area.

Node ------------ Connection	RTT to Gateway (msec)	RTT to bcc.co.uk (msec)	RTT to uwaterloo.ca (msec)	RTT to facebook.com (msec)	RTT to nicta.com.au (msec)
Airtel	301.8 +/- 133.2	484.21+/- 96.6	586.7 +/- 93.49	544.4+/- 116.65	440.3 +/- 91.7
MTNL	476.52 +/- 78.4	435.5+/- 142.3	525.2 +/- 84.6	526.0 +/- 167.0	478.4 +/- 59.4
Reliance	111.1 +/- 30.0	258.7 +/- 36.1	352.4 +/- 30.6	430.6 +/- 26.9	444.2 +/- 30.2

Rural Data:

Table 6.4.2: Latency to landmark nodes Rural area.

Node ------------ Connection	RTT to Gateway (msec)	RTT to bcc.co.uk (msec)	RTT to uwaterloo.ca (msec)	RTT to facebook.com (msec)	RTT to nicta.com.au (msec)
Airtel	317.6 +/- 120.2	533.76 +/- 265.5	692.1 +/- 237.2	705.3 +/- 257.7	515.01 +/- 304.5
MTNL	168.0838 +/- 27.04692	496.8 +/- 203.5	643.9 +/- 111.8	482.4 +/- 65.5	426.3 +/- 115.6

For instance in RTT to bbc.co.uk, RTT to gateway contributes 51% of delay in Airtel GPRS connection, 90.7 % delay in MTNL HSDPA and 31.5% in Reliance EDVO for all of the landmark nodes in Airtel and MTNL connection. We also find that contribution of last hop delay in case of MTNL HSDPA connection for RTT's of all the landmark nodes is more than 90%.

Impact: Having large RTT degrades the user experience and it also affects real time application (voice applications). Hence Voice over IP is not a good idea if user is connected using cellular network.

6.5 Buffer size at gateway.

A common rule of thumb [31] suggests that router queue length should be equal to RTT of an average flow through the link. Larger queues lead to high queuing delays in the network and smaller will lead to packet loss if traffic is bursty.

Validation:
We measure queue length in milliseconds based on the BFind approach, as explained in section 3.4 (i). In urban area, 111 samples were collected over duration of week in case of Airtel GPRS connection and 61 for Reliance EVDO connection. In rural area we managed to collect 28 samples of this test for Airtel GPRS connection over period of 10 days.

In urban area the average queue size for Airtel GPRS connection was 2980 +/- 1906 milliseconds and in rural area it was on average 4229.8 +/- 2741 msec. For Reliance connection, in urban area, the queue size was on average 949.11 +/- 923.18 milliseconds.
It can be seen that average queue size at gateway in Reliance connection is very less than average queue size for Airtel connection observed in both Rural and Urban area.

Impact:
Presence of such large queue size in Airtel GPRS network causes long queuing delay at gateway and hence increasing overall latency, which then impacts real time application and hence also degrades quality of service.

6.6 Packet losses

Packet loss analysis of execution of measurement suite in urban area revealed the following characteristics about packet losses.

Table 6.6.1 Packet loss analysis

Sr. No	Connection	Average number of packet losses %	What % of Packet losses are because of flow break	Average time spent in flow breaks (in Seconds)	What % of packet losses are because of TSO + Delayed acknowledgement.
1	Airtel GPRS	1.6% to 3.8%	~ 0.5%	23.9	~ 44.0%
2	Reliance EVDO	0.57% to 1.3%	~0.92%	23.44	~ 68.0 %
3	Airtel HSDPA	0% to 1.6%	~0.1%	3.3	~ 20.7%
4	MTNL HSDPA	0% to 1.2%	~0.1%	17.37	~ 8.0 %

We can observe that Airtel GPRS connection has large number of packet losses compare to other connections, minimum number of packets losses here are approximately 1.6% of total number of packets transmitted from server to client, which is close to maximum number of packet losses seen in for any of the other connections.

The duration of time spent and number of packet losses because of long duration flow breaks observed are more in case of Airtel GPRS and Reliance EVDO connections when compare to HSDPA connections, approximately 0.5% , 0.9% of total packet losses were because of long duration flow breaks.

It can also be seen that Airtel GPRS and Reliance EVDO connections suffer more because of long bursty behaviour caused by TSO and delayed acknowledgement. For Airtel 44% & for Reliance 68% of packet losses were because of long burst sent from server which we know is because of combination of delayed acknowledgement and TSO.

Further Work:
A better algorithm is needed to distinguish between the packet loss because of TSO and packet loss because of delayed acknowledgement. Current algorithm blames every packet loss resulted from large segment sent from server to TSO+Delayed acknowledgement.

6.7 Comparison of TCP Cubic and TCP Reno

TCP Reno is not bursty in its nature as TCP Cubic hence it is expected that TCP Reno will perform better compare to TCP Cubic over GPRS network. To validate the claim we conducted few comparison tests.

Results:

1. Throughput comparison:

The CDF graph 5.8.1 shows the throughput obtained for Cubic and Reno.

The average throughput for Cubic is: **13997.6** +/- 4322.6 KBps.

The average throughput for Reno is: **14312.8** +/- 6022.1 KBps.

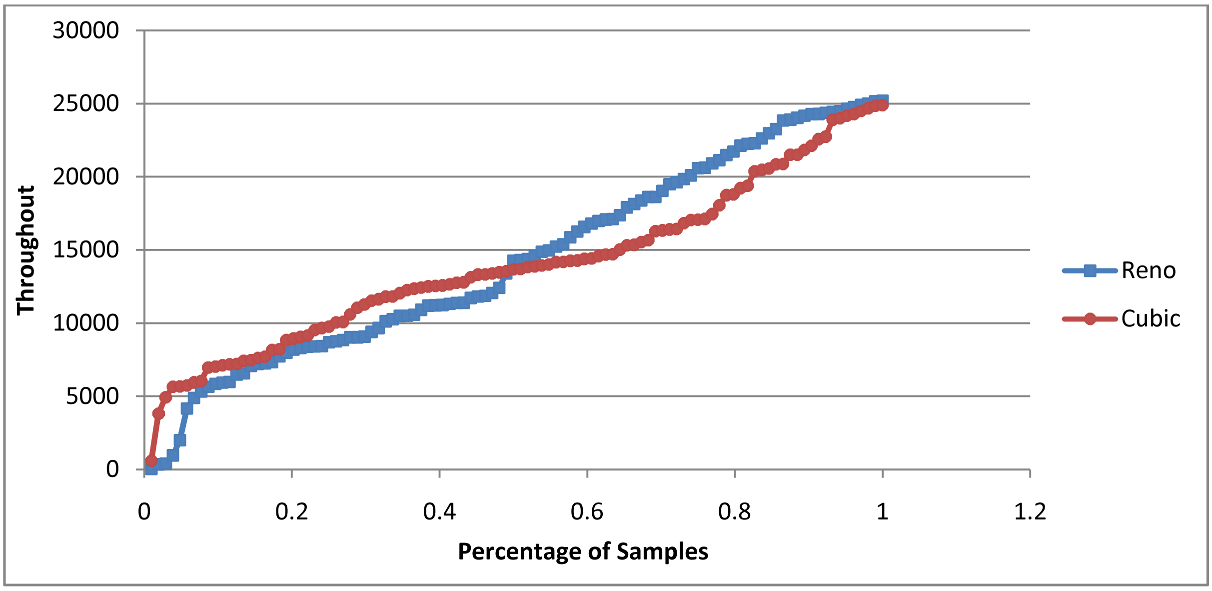

Figure 6.7.1: Throughput comparison of TCP Cubic and TCP Reno.

2. Number of retransmissions:

The CDF graph 5.8.2 shows the numbers of retransmissions during file download.

Average Number of retransmission for Cubic: **223.3** +/- 75.3

Average Number of retransmission for Reno: **90.1 +/-** 48.0

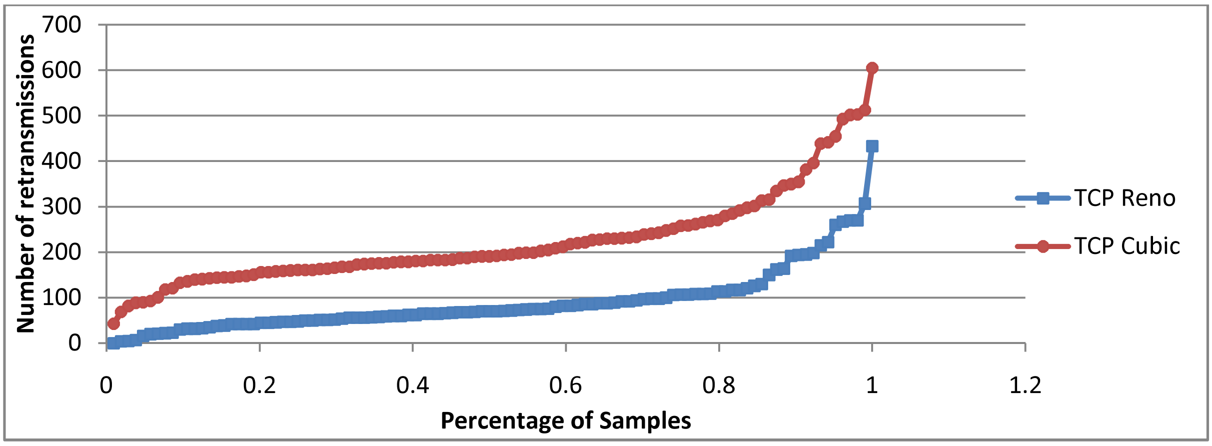

Figure 6.7.2: Number of retransmission comparison of TCP Cubic and TCP Reno.

Findings:

Long duration tests showed that average throughput achieved over TCP Reno is better than average throughput over TCP Cubic. Also the number of retransmissions observed with TCP Reno is 60% less compared to number of retransmissions with TCP Cubic.

Suggestion:

Based on these findings we would suggest that one should use TCP Reno as congestion control algorithm when working with networks like GPRS having high latency and low bandwidth.

6.8 Radio Resource Management Test:

Base stations normally put devices not on active call in idle state to conserve battery power of end device and to utilize radio resources better. The time of inactivity after which a device is put in idle state and the time it takes to return to active state can impact end user experience. In our experiments, we found idle timeouts and activation times for Reliance and MTNL as shown in Table5.9.1.

Table 6.8.1: Timeout and Transition time with Radio Resource Management.

Connection VS Observation	Reliance	MTNL
Idle timeout	10 Sec	50 Sec
State transition delay(Idle to Active)	1.6 Sec	2 Sec

Impact: Based on these values it is easily conceivable that a user browsing the web could potentially experience a delay of two seconds every time he/she visits a new page.

6.9 Some application layer tests:

6.9.1 In network antivirus detection:

This test is carried out to detect if the service providers network has inbuilt antivirus software. For **validation** we downloaded the files with virus signature from well known EICR site over Airtel GPRS, Reliance EVDO and MTNL HSDPA connection. It was **found** that none of the service provider has inbuilt network antivirus protection implemented.

6.9.2 IPv6 Adoption test:

June 06, 2012, the Global transition from Internet Protocol version 4 (IPv4) to Internet Protocol version 6 (IPv6) began and the day was celebrated as the 'World IPv6 Launch Day'. To check if the service providers are ready for this transition a test provided by Google at http://www.worldipv6launch.org/ was carried out.

Findings:

It was observed that none of the service providers currently supports IPv6 protocol. Only the DNS server of MTNL service provider supports ipv6.

Impact:

If service provider does not support IPv6 then "IPv6 only" sites will not be accessible.

Chapter 7: Summary and future work

7.1 Summary and Contribution:

Following are the tasks achieved and findings we have made in this project.

- Design rural Internet measurement suite.

  A robust and rich set of tests have been identified, which will help us infer several network characteristics during final deployment and data collection phase. The design also has provision to automatic results collection and auto upgrades.

- Understand availability and stability of connection provided by different service providers in urban and rural area.

  Execution of test suite at rural and urban location revealed that MTNL and Airtel services were unavailable at night time in rural area. It is suspected that base stations at rural location were conserving energy by turning off or significantly reducing power of data interfaces at night.

  MTNL connection showed very less stability because of large number of disconnection. Such frequent disconnections severely impact the real time application and also degrade the QoS from end users perspective.

- Comparing the data rates advertised by different service providers with achievable data rates.

  It was observed that with MTNL (HSDPA) and Reliance (EV-DO) service provider's actual throughput achieved was only 32.3% and 21.04% of advertised data rates respectively. But with Airtel GPRS connection achievable throughput was very large compare to advertised data rates.

- Understanding the impact of time of day on TCP throughput.

  Presence of diurnal pattern in achievable throughput was detected for all three service providers. Throughput achieved during 1.00 AM to 7.00 AM is very high compared to other time of day. These diurnal patterns can be exploited as,

    - From users perspective: Delay tolerant application, such as sending email, can be used to send data at night time when high throughput is achieved.
    - From service provider's perspective: They can provide incentive to their customers for using internet at night time when throughput is very high.

- Confirmation that the last hop is the bottleneck on the network path.

  Ping test confirmed well know fact that last hop is the bottleneck in network connection.

- Comparison and Impact of buffer sizes at gateway for different service providers over different access technology.

  Queue size at gateway was very large for Airtel GPRS connection compared to other service providers. Presence of such large queue size in Airtel GPRS network causes long queuing delay at gateway and hence increasing overall latency, which then impacts real time application and hence also degrades quality of service.

- TCP Cubic dynamics and impact of TSO and IW on TCP flow.

  TCP cubic is known to be bursty in its nature. It was observed that most of the packet losses occur when TCP cubic is in its convex phase where it is probing for more bandwidth.

  The presence of optimization techniques such as TCP segmentation offloading and larger initial window size adds on to this bursty nature of TCP which then results in packet losses and increased number of retransmissions. This gives the non optimal performance of TCP with high latency and low bandwidth networks like GPRS.

  When we disable TSO and use standard initial window size (4) as specified in RFC 3390, with little impact on TCP throughput we see drastic improvement in number of retransmissions.

- Comparison of TCP Cubic and TCP Reno congestion avoidance algorithms.

  Comparison test of TCP Cubic and TCP Reno over high latency and low bandwidth network showed that. TCP Reno gives better average throughput while reducing the number of retransmissions drastically.
  So we suggest that TCP Reno should be used as congestion control algorithm over networks with high latency and low bandwidth, like Airtel GPRS.

- IPv6 deployment:

  With June 6th 2012 being marked as "IPv6 Launch Day", major ISPs are now permanently turned on IPv6. But our test showed that none of the service providers(Airtel, MTNL and Reliance) supports infrastructure for communication over IPv6 protocol. Hence with these service providers we cannot access IPv6 only sites.

7.2 Future Work:

There are still lot of scope and work to do in this project, like:

- Finding reason for unexplained observations such as long duration flow breaks because of exponential increase in RTO.

- Creating a final deployable product which includes the entire proposed test.

- Detecting the presence of middle box and its type in network and to understand how do then impact.

- Detecting the traffic shaping done by service providers and understanding how it impacts the performance.

References

[1] Telecom Regularity Authority of India, Annual report 2011-2012. *http://www.trai.gov.in*

[2] V. Jacobson, "Pathchar: A Tool to Infer Characteristics of Internet Paths," *ftp://ftp.ee.lbl.gov/pathchar/*, Apr. 1997.

[3] A.B. Downey, "Using Pathchar to Estimate Internet Link Characteristics", in Proceedings of ACM SIGCOMM, Sept.1999, pp. 222–223

[4] B. A. Mah, "pchar: a Tool for Measuring Internet Path Characteristics", *http://www.employees.org/~bmah/Software/pchar/*, February 1999

[5] R. Carter, M. Crovella, "Measuring Bottleneck Link Speed in Packet-Switched Networks", Performance Evaluation, Vol. 27-8, pp. 297-318, Oct. 1996

[6] C. Dovrolis, P. Ramanathan, and D. Moore, "Packet dispersion techniques and capacity estimation", submitted to IEEE/ACM Transactions of Networking.

[7] S. Saroiu, P. Gummadi, and S. Gribble, Sprobe: A Fast Technique for Measuring Bottleneck Bandwidth in Uncooperative Environments, *http://sprobe.cs.washinton.edu*, 2002

[8] Manish Jain, Constantinos Dovrolis, "Pathload: A measurement tool for end-to-end available bandwidth".

[9] N. Hu and P. Steenkiste, "Initial Gap Increasing (IGI) and Packet Transmission Rate (PTR)", *http://www.cs.cmu.edu/~hnn/igi*, March 10, 2003.

[10] V. Ribeiro, R. Riedi, R. Baraniuk, J. Navratil, and L. Cottrell, "pathChirp: Efficient Available Bandwidth Estimation for Network Paths," in Proceedings of Passive and Active Measurements(PAM) workshop, Apr. 2003.

[11] B. Melander, M. Bjorkman, and P. Gunningberg, "A New End-to-End Probing and Analysis Method for Estimating Bandwidth Bottlenecks", in IEEE Global Internet Symposium, 2000.

[12] B. Melander, M. Bjorkman, and P. Gunningberg, "Regression-Based Available Bandwidth Measurements," in International Symposium on Performance Evaluation of Computer and Telecommunications Systems, 2002.

[13] Fall, K. and Floyd, S. 1996. Simulation-based Comparisons of Tahoe, Reno and SACK TCP. ACM SIGCOMM Computer Communication Review, 26(3):5-21.

[14] Allman, M., Paxson, V. and Blanton, E. 2009. TCP Congestion Control. RFC 5681.

[15] Marcel Dischinger, Andreas Haeberlen, Krishna P. Gummadi, Stefan Saroiu, "Characterizing Residential Broadband Networks", IMC’07, October 24-26, 2007, San Diego, California, USA.

[16] Ahmed Elmokashfi, Amud Kvalbein, Jie Xiang and Kristan R. Evensen, "Characterizing delays in Norwegian 3G networks", *http://simula.no/publications/Simula.simula.1116/simula_pdf_file*.

[17] Marko Jurvansuu, Jarmo Prokkola, Mikko Hanski and Pekka Perälä, "HSDPA Performance in Live Networks", In the ICC 2007 proceedings.

[18] M. Palola, M. Jurvansuu, and J. Korva, Breaking down the mobile service response time, in Proc. IEEE ICON 2004, Vol. 1, Nov. 2004, pp.31 – 34.

[19] J. Prokkola, M. Hanski, M. Jurvansuu and M. Immonen, Measuring WCDMA and HSDPA Delay Characteristics with QoSMeT, in proc. of ICC’07, Glasgow, UK, 24-28 June 2007.

[20] M. Hanski, A Tool for Monitoring End-to-end Quality of Service in IP Networks, Master's thesis, University of Oulu, Department of Electrical and Information Engineering, Oulu, Finland. 2005

[21] Andreas Hanemann, Athanassios Liakopoulos, Maurizio Molina, D. Martin Swany, "A Study on Network Performance Metrics and their Composition", *http://marco.uminho.pt/~dias/MIECOM/GR/Projs/P4/TNC_Metric_Comp_FWork-full-v4.pdf*

[22] http://en.wikipedia.org/wiki/Nmap

[23] Christian Kreibich, Nicholas Weaver, Boris Nechaev, Vern Paxson, "Netalyzr: Illuminating The Edge Network", IMC’10, November 1–3, 2010, Melbourne, Australia.

[24] Feng Qian et. al, “Characterizing Radio Resource Allocation for 3G Networks”, IMC 2010.

[25] http://linux.about.com/od/commands/l/blcmdl1_dig.htm

[26] http://en.wikipedia.org/wiki/Traceroute

[27] http://www.tcptrace.org/

[28] http://linux.about.com/od/commands/l/blcmdl1_curl.htm

[29] http://en.wikipedia.org/wiki/Iperf

[30] http://linux.about.com/library/cmd/blcmdl8_tcpdump.htm

[31] C. Villamizar and C. Song., “High performance TCP in ANSNET”. ACM Computer Communication Review, 24(5):45–60, 1994

[32] Nandita Dukkipati Tiziana Reice Yuchung Cheng Jerry Chu Natalia Sutin, Amit Agarwal Tom Herbert Arvind Jain, "An Argument for Increasing TCP's Initial Congestion Window"