

**Assessment of the Multicast Dissemination
Protocol for Constrained Satellite Channels**
by

Sandeep Muddasani and Stephen Horan

NMSU-ECE-03-002

Assessment of the Multicast Dissemination Protocol for Constrained Satellite Channels

Sandeep Muddasani and Stephen Horan
Manuel Lujan Space Tele-Engineering Program
New Mexico State University
Las Cruces, NM

Prepared for

National Aeronautics and Space Administration
Goddard Space Flight Center
Greenbelt, MD

under Grant NAG5-9323

April 30, 2003



Klipsch School of Electrical and Computer Engineering
New Mexico State University
Box 30001, MSC 3-O
Las Cruces, NM 88003-8001

Contents

List of Figures	ii
List of Tables	iii
INTRODUCTION	1
Simulator Environment	2
Purpose	5
Scope	5
Topic Development	5
Intended Audience	6
METHODS, ASSUMPTIONS, AND PROCEDURES	7
Investigation Procedure	7
Configuring the simulator	9
Configuring PPP	10
Test Files	12
Configuration of MDP	14
System of Measurement	17
Investigation Statistical Analysis	17
RESULTS AND DISCUSSION	20
Data Throughput Measurements	20
Example Analysis of Results	24
CONCLUSIONS	25
REFERENCES	26
APPENDIXES	27
LIST OF SYMBOLS, ABBREVIATIONS, AND ACRONYMS	44

List of Figures		
No.	Title	Page
1	SGLS hardware configuration.	3
2	Arrangement of VI modules in SGSL.	4
3	The error generation in SGSL.	4
4	The four layers of a TCP/IP Protocol suite.	7
5	Different levels of MDP and FTP	8
6	The IP addresses of the experimental setup	9
7	File transfer time versus BER plot for different files using MDP .	20
8	File transfer time versus BER plot for different files using FTP.	21
9	MDP throughput as a percentage of available channel capacity	23
10	FTP throughput as a percentage of available channel capacity	23

List of Tables		
No.	Title	Page
1	Randomized File Sequence List	13
2	The data files and their corresponding BER	14
3	PPP Optimal Frame Size (bytes)	17
4	The average values of file transfer times at different BER's using MDP	21
5	The average values of file transfer times at different BER's using MDP at optimal frame size.	22
6	The average values of file transfer times at different BER's using FTP at optimal frame size.	23
7	Experiment results for FTP testing	26
8	Experiment results for MDP testing	27
9	Example computations for Honestly Significant Difference	29

Introduction

The use of Internet-type protocols for space communications is no longer considered a “new” topic of investigation. Research on the subject has been conducted at NASA, DoD, NMSU and contractors. Most of the research examined the performance of TCP based file transfer protocols in space communications. An area of interest for many future missions is to identify file transfer protocols that will provide reliable data delivery over a space communication channel. TCP provides reliable transfer but it is known to have limitations in communication environments with long propagation delays and high error rates. Also, TCP will not function at all without a two-way communication link to provide acknowledgment responses.

An alternative to TCP is UDP, which does not use any of the session setup or acknowledgement techniques of TCP. UDP is a connection less protocol which means that UDP simply adds some stream multiplexing and checksum information to the user's data, sends the packet and forgets about the packet. Because of the UDP basic structure, additional software is needed to realize an end-to-end connection approaching reliable transport.

One such file transfer protocol designed on top of UDP is Multicast Dissemination Protocol (MDP)[1]. MDP is designed to provide reliable multicast data and file delivery services on top of the generic UDP/IP multicast transport. Reliability implies that all packets in a session are guaranteed to be delivered to the receivers in optimum period of time. MDP is an efficient negative acknowledgement (NACK) based reliable multicast protocol that is especially suitable for efficient bulk data transfer. Efficient implies that the MDP has the property of optimum utilization of the space channel and also has less number of retransmissions in high BER environment. The core MDP framework makes no design assumptions about network structure, hierarchy, or reciprocal routing paths. The techniques and building blocks utilized in MDP are directly applicable to "flat" multicast groups but could be applied to a given level of a hierarchical (e.g. tree-based) multicast distribution system if so desired. In our study we are not testing the multicast

features of MDP but we are testing its efficiency and reliability in space channel environment.

The space channel poses a number of challenges to providing reliable, end-to-end data communication with a user specified level of service. Losses due to transmission errors, large propagation delays, constrained bandwidth, asymmetric link rates, and intermittent connectivity all conspire to severely limit the performance of file transfer protocols.

Simulator Environment

We have used the NMSU Space-to-Ground Link Simulator (SGLS)[2] to provide simulation capabilities to test the protocols. The satellite environment being simulated has one terminal at a ground station and the other terminal at the satellite. The SGLS channel simulator as mentioned in current paper performs the error generation, intermittent connectivity and link delay operations to test the protocol performance.

Basically, the SGLS has the following characteristics

1. Simultaneous bi-directional data flow which means it provides both forward and return channels
2. Allow user-selectable error rates varying from 0 through 0.001, as it would be in space environment
3. Simulates asymmetric links by providing data rates from 2400 bps through 115200 bps in both forward and return directions.
4. Provide for a simulated delay up to 5 seconds on each link.
5. User selectable link cut off time in both directions to simulate the intermittent connectivity

The SGLS utilizes the LabVIEW [5] programming language to control data throughput through the simulator, mix the base band data stream with the user-selected error vector, and provide for the user-selectable link delay value. The hardware configuration is illustrated in Figure 1. The LabVIEW software is run as an application on each of the SGLS computers. Typically, the LabVIEW modules are the only applications software running on the computers. This configuration was developed to model point-to-point satellite links in its current configuration.

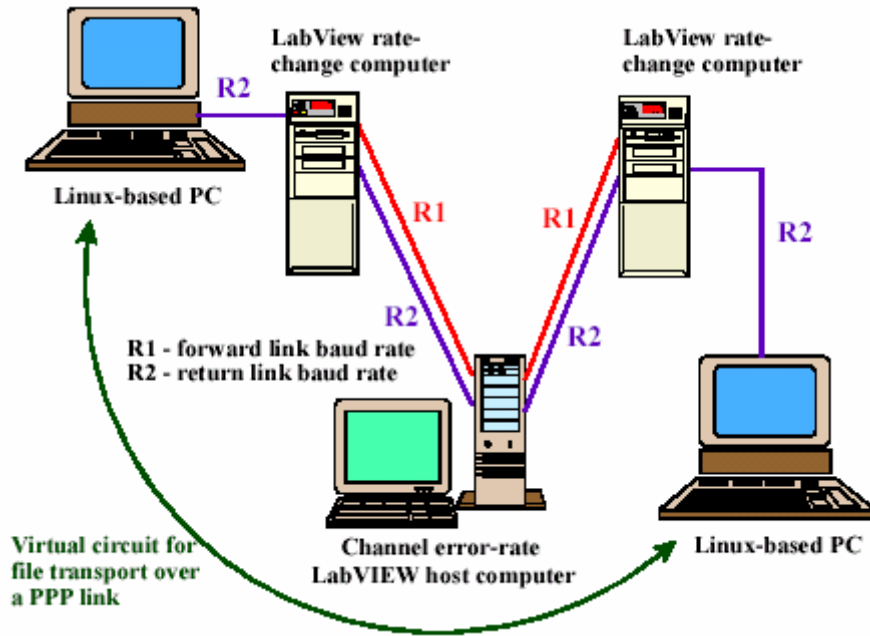


Figure 1: SGLS Hardware Configuration

The three PCs in the SGLS are Dell 4100 computers with 128 Mbytes of memory running Windows 98. The first Linux-based PC is a Dell 600 computer. This is our logical ground station computer. The second Linux-based PC is a Gateway 266 MHz. This is our logical satellite computer. The logical ground station computer is running SuSe Linux version 6.1 and the logical satellite computer is running Red Hat Linux version 7.2. The SGLS is connected to the Linux computers using serial cables connected to the COM serial ports on each computer. The data connections are configured without hardware or software handshaking to allow for a simulation that would be similar to interfacing with a satellite radio system. The link speed (R2 as shown in the Figure 1) between the SGLS and the two Linux computers is fixed at 115200 bps. The tests run with the symmetric links will have both forward (R1 in Figure 1) and return (R2 as shown in the Figure 1) links set to 115200 bps. The tests run with asymmetric links had the forward link set to 2400 bps and the return link set to 115200 bps.

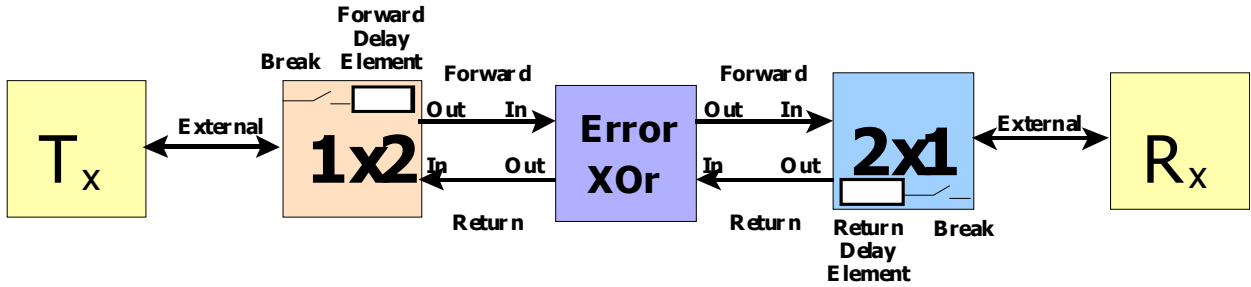


Figure 2: Arrangement of VI modules in SGLS.

Figure 2 explains the error generation, rate splitting and intermittent connectivity operations of SGLS. In Figure 2, the Tx and Rx are logical satellite and logical ground station PC's respectively. The term *External* means the data link between Tx, SGLS and Rx. The term *Forward* implies the path from Tx to Rx inside the SGLS and the term *Return* implies the path from Rx to Tx inside the SGLS. The module 1 X 2 is the interface between the SGLS and Tx. Module 1 X 2 also introduces the rate splitting, link cutoff for intermittent connectivity and propagation delay element. The module 2 X 1 provides the interface between SGLS and Rx and also it introduces propagation delay element and optional link cut-off properties. In both 1 X 2 and 2 X 1 modules the link cut-off time and propagation delay time are user selectable. The *Error XOR* block in Figure 2 does the error generation function.

Input Data Stream: ... 0 0 0 1 1 0 0 1 ...

Error Vector: ... 0 0 0 0 0 1 0 0 ...

Output Data Stream: ... 0 0 0 1 1 1 0 1 ...

bit error location ↑

Figure 3. The error generation in SGLS.

As shown in the Figure 3 the output data stream is produced by XORing the input data stream and the error vector. The error vectors consist of all 0's except for 1's at locations where bit errors are to occur. The number of bit errors to be generated is user selectable.

Purpose

In this report we compared the performances of UDP/IP based file transfer protocol called MDP and TCP/IP based file transfer protocol FTP using Space-to-Ground Link Simulator (SGLS).

The goal of this study is to answer the basic questions, namely

- (1) Is there an overall advantage of UDP/IP based file transfer protocol Multicast Dissemination Protocol (MDP) over TCP/IP based file transfer protocol (FTP).
- (2) How does BER affect the protocol performance
- (3) How do MDP and FTP behave under intermittent connectivity

Scope

This document covers experiments made to test the following features :

- Behavior of UDP/IP based protocols in high BER environment.
- Showing the performances of UDP/IP based protocol MDP and TCP/IP based protocol FTP statistically and graphically different.

The experiments only cover unicast mode and not multicast mode. The experiments are run at only a single data rate of 115200 bps. Multiple channel error rates are covered.

Topic Development

The remaining chapters in this report are developed along the following lines:

- The Methods and Procedures chapter will discuss the methods for running the tests and under what conditions the tests are run.
- The Results chapter will discuss results of the conducted tests
- The Conclusions chapter
- The Appendix chapter includes the raw data of the tests at different BER's.
- The reference chapter contains the list of references.

At the end there is an acronym chapter.

Intended Audience

The intended audience for this study are the institutions that are doing research on internet protocol performance in space.

Methods, Assumptions, and Procedures

Investigation Procedure

The performance of MDP and FTP are tested by calculating the average time for the transfer of data files. TCP/IP protocol suite is the combination of different protocols at various layers. TCP/IP is normally considered to be a four-layer system as shown in Figure 4.

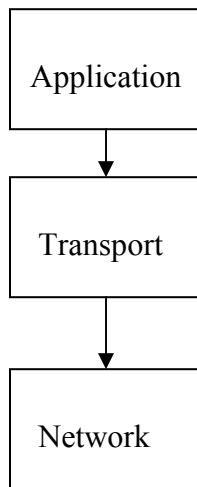


Figure 4: The layers of a TCP/IP Protocol suite.

In Figure 4 each layer has a different responsibility

1) The network layer handles the movement of packets around the network. Routing of packets takes place here. IP (Internet Protocol), ICMP (Internet Control Message Protocol), and IGMP (Internet Group Management Protocol) provide the network layer.

2) The transport layer provides a flow of data between two hosts, for the application layer above. In the TCP/IP protocol suite there are two vastly different transport protocols namely

- TCP (Transmission Control Protocol).

- UDP (User Datagram Protocol).

TCP provides a reliable flow of data between two hosts. It is concerned with things such as dividing the data passed to it from the application in to appropriately sized chunks for the network layer below, acknowledging received packets, setting timeouts to make certain the other end acknowledges packets that are sent and so on. Because this reliable flow of data is provided by the transport layer, the application layer can ignore all these details.

UDP, on the other hand, provides a much simpler service to the application layer. It just sends packets of data called datagrams from one host to the other, but there is no guarantee that the datagrams reach the other end. Any desired reliability must be added by the application layer.

3) The application layer handles the details of the particular application. The common TCP/IP applications are Telnet, FTP, SMTP (Simple Mail Transfer Protocol), etc.

Corresponding to the layers described in Figure 4 the protocols used in this report are divided as shown in the Figure 5.

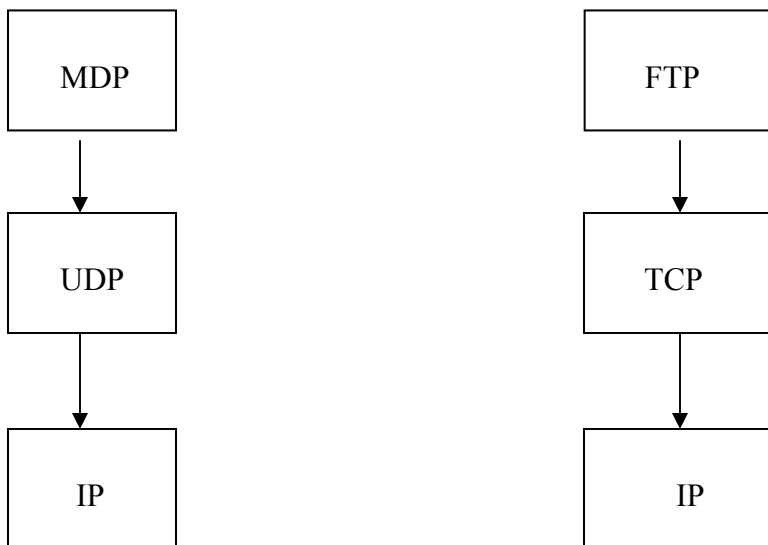


Figure 5: Different levels of MDP and FTP

Here the application layer consists of two different file transfer protocols namely MDP and FTP. MDP is based on UDP/IP whereas FTP is based on TCP/IP.

Configuring the Simulator

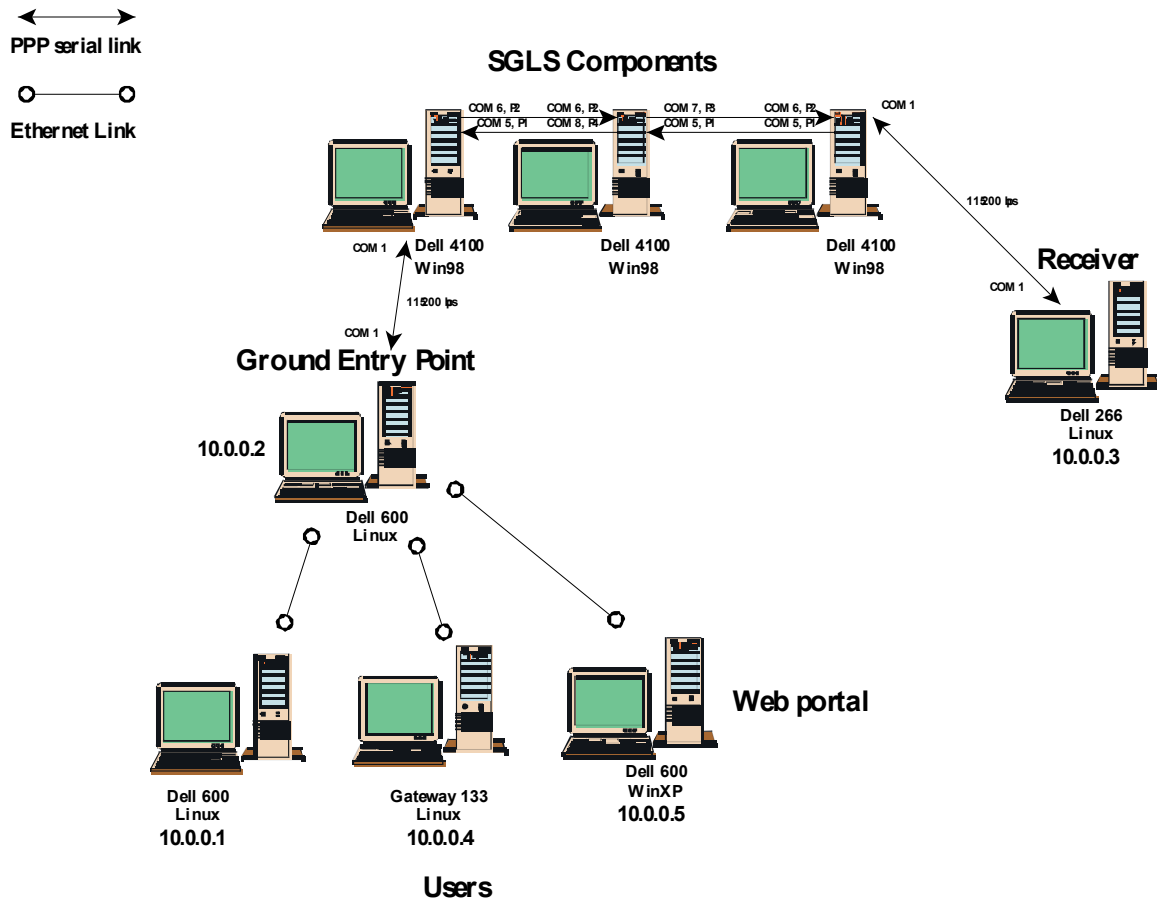


Figure 6: The IP addresses of the experimental setup

Configuring the experimental setup for our experiments

Figure 6 shows the overall experimental setup. In the Figure 6 the right most computer is the logical satellite computer and its IP address is 10.0.0.3 and the PC to the left of SGLS is the logical ground station with an IP address of 10.0.0.2. The serial port (COM 1) of

logical satellite computer is connected to the serial port (COM 1) of the PC at the right end of SGLS and on the other side the serial port (COM 1) of logical ground station computer is connected to the serial port (COM 1) of PC to the left end of the SGLS using a cable. The cabling inside the SGLS is as shown in the Figure 6. The next few lines explain the experimental setup configuration for our tests.

- The transfer of data is done from the logical satellite computer (with IP address 10.0.0.3) and collect the data at the logical ground station (10.0.0.2).
- The BER's are introduced by the middle computer of SGLS.
- For our experiments we used symmetrical data rates which means the forward and return data rates is same (115200 bps).
- The intermittent connectivity for the experiments is provided by either of the end computers of SGLS by breaking the link. Using the LABVIEW interface we can specify the time to next link cut-off and the duration of the link cut-off.

Configuring PPP

The communication [3] between transmitter and receiver will be through a serial port. The communication port used on both transmitter and receiver is “ /dev/ttyS0 ” (COM1 in DOS/Windows). In Linux PPP is implemented by the PPP daemon “ pppd ”. Its configuration is done through files in the “ /etc/ppp ” directory. In the next sections we illustrate how to configure the PPP daemon for both the transmitter and receiver.

On the receiver side

An options file /etc/ppp/options.ttyS0 is created which contains PPP options specific to connections through “ /dev/ttyS0 ”. The entries in /etc/ppp/ttyS0 file are

local

```
noauth
10.0.0.2:10.0.0.3
115200
mru 1500
mtu 1500
```

The meaning of each line is as follows :

- *local* indicates that modem lines are not used
- *noauth* indicates that it won't need any authorization for communication with transmitter.
- *10.0.0.2:10.0.0.3* indicates that *10.0.0.2* is the remote IP address and *10.0.0.3* is the local IP address.
- *mru* indicates the maximum receiving unit. By changing this parameter we adjust the PPP frame size. By default this is 1500.
- *mtu* indicates the maximum transmitting unit. By changing this parameter we adjust the PPP frame size. By default this is 1500.

On the transmitter side

On the transmitter side the entries in `/etc/ppp/ttyS0` file are similar to that of receiver side except the remote and local IP addresses will be swapped. The configuration file contains the following entries :

```
local
noauth
10.0.0.3:10.0.0.2
115200
mru500
```

mtu 1500

Installing pppd

The PPP daemon can be installed in two ways

- *getty*-like installation
- Installing manually

In *getty*-like installation we can have the *pppd* (PPP daemon) start when we boot the system. A typical way of achieving this is to edit the `/etc/inittab` file. This file contains information for initializing the system. We add the following to this file:

```
# Start pppd for the serial laplink.  
pd:2345:respawn:/usr/sbin/pppd /dev/ttyS0 nodetach
```

This is interpreted as follows: for run levels 2, 3, 4 and 5 start `/usr/sbin/pppd /dev/ttyS0 nodetach` and if it dies (at the end of a connection) respawn (start a new one). The `"nodetach"` option makes that *pppd* stays connected to the terminal that started it, rather than forking and exiting. This option is necessary because the `"init"` process would respawn a new one immediately otherwise.

We can start the PPP connection manually by typing

```
/usr/sbin/pppd /dev/ttyS1 nodetach
```

at the command line.

We can kill *pppd* by typing

```
killall pppd
```

at the command line.

Test Files

The test files used for testing the performance are arranged randomly instead of arranging them from smallest to largest. The files arranged randomly to ensure that the results are

not order dependent We have used 10K, 100K, 1M and 10M bytes of files and the order of files are as shown in the table.

Table 1: Randomized File Sequence List			
Sequence Number	File Identifier	Sequence Number	File Identifier
1	100KB01	33	10KB10
2	1MB02	34	10MB02
3	100KB15	35	10KB14
4	10MB12	36	10KB04
5	100KB14	37	10KB09
6	10MB06	38	100KB13
7	100KB10	39	10KB02
8	10MB10	40	10KB07
9	10MB16	41	10MB15
10	100KB12	42	1MB13
11	1MB04	43	1MB12
12	10KB15	44	10MB07
13	10MB13	45	100KB06
14	1MB08	46	100KB09
15	1MB07	47	10KB05
16	10KB08	48	100KB12
17	10KB11	49	10MB05
18	10KB11	50	1MB15
19	10MB09	51	1MB01
20	1MB14	52	10KB06
21	10KB13	53	1MB05
22	100KB05	54	100KB04
23	100KB08	55	10MB03

24	100KB07	56	1MB06
25	1MB09	57	1MB10
26	100KB02	58	10KB01
27	100KB16	59	1MB03
28	1MB16	60	10KB16
29	10KB03	61	100KB03
30	10KB12	62	10MB04
31	10MB11	63	10MB14
32	10MB08	64	10MB01

The error files which control the channel error characteristics are given in the Table 2.

Table 2 :The data files and their corresponding BER		
File	BER	Number of errors in file
infinite.dat	0	0
A1050d.dat	10^{-6}	10
A950c.dat	10^{-5}	100
A825k.dat	10^{-4}	1000

Configuration of MDP

Starting mdp is done via a UNIX command line application with no GUI. The command line of MDP used on logical satellite and logical ground station PC's is as follows

On logical satellite computer

```
mdp -A 10.0.0.2/portnumber -F 10.0.0.3 -r 88000 -d6 -l /home/report -L  
/home/report -i 300 /home/testfiles
```

On logical ground station computer

```
mdp -A 10.0.0.3/portnumber -F 10.0.0.2 -L /home/report -d6 -l /home/debug -D  
/home/data -a -u
```

The command line parameter and option descriptions of the above command line is as follows

- | | |
|-----------------------|--|
| -a | Configure client to permanently store (archive) files instead of just using the "-D" archive Directory as a temporary cache for received files. |
| -u | Configure client to unicast protocol messages back to the server. This may be useful for some asymmetric network topologies. |
| -A <address/port> | Destination IP address/port of transfer session |
| -F<interface Address> | Specify the IP address of the network interface to use for multicast traffic |
| -r <txRate> | Maximum transmission rate in bits per second. |
| -d <debug Level> | Set debug level. Determines level of debugging detail displayed or logged. Debug levels currently include: <ul style="list-style-type: none">• 0 : Misc errors only.• 2 : Major events (e.g. file tx/rx start/stop) plus client statistic reports.• 4 : Detailed file reception progress |

- 6 : General protocol operation.
- 8 : More detailed protocol operation.
- 10 : High level NACK construction/handling
- 12 : Detailed NACK construction/handling

- l <debugLogFile>* Specify debug message log file.
- L <log File>* Log transmit/receive completion events to specified log file.
- i <object Interval>* Time delay between transmission of individual files in seconds
- <tx directory/file names ...>* Files/Directories to transmit (These should be last on the command-line)

The procedure for file transfer using MDP is

1. Configuring MDP as shown in the above section.
2. The frame size is set at the application level according to Table 3 and the PPP frame size is set to default(1500 bytes).
3. The randomized list of files given in the Table 1 are transferred separately at 0, 10^{-6} , 10^{-5} and 10^{-4} BER's.
4. The time interval of 3000 seconds is set between each file transfer so that only one file is transmitted at a time.
5. The transfer time for each file is noted from the log file (/home/report).

The procedure for file transfer using FTP

1. Configuring the FTP.
2. The frame size of PPP is set according to Table 3

BER	PPP Optimal Frame Size (bytes)
0	1500
10^{-6}	1300-1500
10^{-5}	650
10^{-4}	128

Table 3: PPP optimal frame size at different BER's.

3. The randomized list of files given in the Table 1 are transferred separately at 0, 10^{-6} , 10^{-5} and 10^{-4} BER's.
4. The time for each file transfer are noted.

System Of Measurement

The system of measurement for testing the performance of MDP and FTP is time. The average time of transfer of 10K, 100K, 1M and 10M at different BER's are calculated and tabulated both for FTP and MDP.

Investigation Statistical Analysis

Based on the analysis done by [4] we chose the number of times each file to be sent as 16, which is sufficiently large to statistically detect the significant mean difference of 1 second at a 95% confidence level. Most satellite transfers can be thought of a single-attempt trial. The network would have no memory of previous results or chance to optimize based on previous data streams. We consider 16 replicate observations will be representative of these single shot attempts at data transfers.

When comparing the file transfer time, the initial measurement is the average transmission time, μ . The average transmission time μ for each file, namely 10K, 100K, 1M and 10M at $0, 10^{-6}, 10^{-5}$ and 10^{-4} BER's is calculated using the formula

$$\mu = \frac{1}{N} \sum_{i=1}^N t_i \quad (1)$$

Where t_i is the transfer time for a single run of a file through SGLS in a set of N measurements to be transmitted.

The standard deviation in the time, s is computed using the following equation:

$$s = \frac{1}{N-1} \sum_{i=1}^N (t_i - \mu)^2 \quad (2)$$

Our experiment goal is to determine if there is any statistically meaningful difference between MDP and FTP under same conditions. For this we need the average time and standard deviation in the average time of both MDP and FTP for the statistical comparison and the desired confidence level for the comparison is chosen as 95%.

The experimental analysis then becomes a testing of the hypothesis that the mean transfer time, μ_A , of Configuration A (MDP) is statistically same (or statistically different) as the mean transfer time, μ_B , of configuration B (FTP) at a confidence level of 95%.

The hypothesis testing problem can be written as [6]:

- H_0 : The mean of the first configuration (MDP), μ_A is same as the mean of the second configuration (FTP), μ_B
- H_1 : The mean of the first configuration (MDP), μ_A is different from the mean of the second configuration (FTP), μ_B

To test the difference in the means of two configurations, we use the basic confidence level method of discriminating between two means.

If we conduct N experiments each on the two systems such that there is a one-to-one correspondence between the i th test on a system A and the i th test on system B, then the observations are called *paired*. For example, if x_i and y_i represent the performance on the i th workload, the observations would be called *paired*. If there is no correspondence between the two samples, the observations are called *unpaired*.

The analysis of paired observations is straight forward. The two samples are treated as one sample of N pairs. For each pair, the difference in performance can be computed. A confidence can be constructed for difference. If the confidence interval includes zero, the systems are not significantly different.

For smaller samples, confidence interval is constructed only if the observations come from a normally distributed population. For such samples, the $100(1-\alpha)\%$ confidence interval is given by

$$(\bar{x} - t_{[1-\alpha/2; n-1]}s / \sqrt{n}, \bar{x} + t_{[1-\alpha/2; n-1]}s / \sqrt{n}) \quad (3)$$

Here, $t_{[1-\alpha/2; n-1]}$ is the $(1-\alpha/2)$ quantile of a t -variate with $n-1$ degrees of freedom.

Here we are looking at the equation for the difference in the means assuming since a small number ($N < 30$) of measurements is available, the critical value for the t -distribution is used instead of the critical value for the normal distribution, z . The critical values are given in tables in the Appendix. In this test procedure, if the 95% confidence level includes 0, then the hypothesis H_0 is accepted and if the 95% confidence level does not include zero the hypothesis H_1 is accepted. For a 95% confidence level, $\alpha = 0.05$ and $\alpha/2 = 0.025$ and the number of degrees of freedom, v , is $v = N - 1$ where N is the number of data points in each configuration.

An alternative statistic for telling if the difference in the means is statistically significant is Tukey's Honestly Significant Difference. This method can also be used when the differences between more than two populations are being made. In this method, the hypothesis H_0 is rejected if

$$\left| \bar{x}_i - \bar{x}_j \right| \geq q_{\alpha, a, a(n-1)} \sqrt{MS_e/n} \quad (4)$$

In the above equation, the quantity n is the number of measurements made in each experiment set (assumed to be the same), α determines the confidence level, and a is the number of different experiment sets being compared. The critical value of the studentized statistic is given in the Appendix for various range values, a , and the number of degrees of freedom. The quantity MS_e is computed using

$$MS_e = \frac{T - A}{N - a} \quad (5)$$

Where T and A are summations over the number of points ($1 < j < n$) and the number of populations being compared, m ($1 < i < m$):

$$T = \sum_i \sum_j t_{ij}^2 \quad (6)$$

$$A = \frac{\sum_i \left(\sum_j t_{ij} \right)^2}{n} \quad (7)$$

Both of these analysis techniques can be used to see if the results are consistent.

Results and Discussion

Data throughput measurements

The data throughput measurements can be done in terms of file transfer time or the effective channel utilization. In the former method the file transfer time is the absolute transfer time of the file and in the later method channel utilization is the percent of the channel utilized by the protocol.

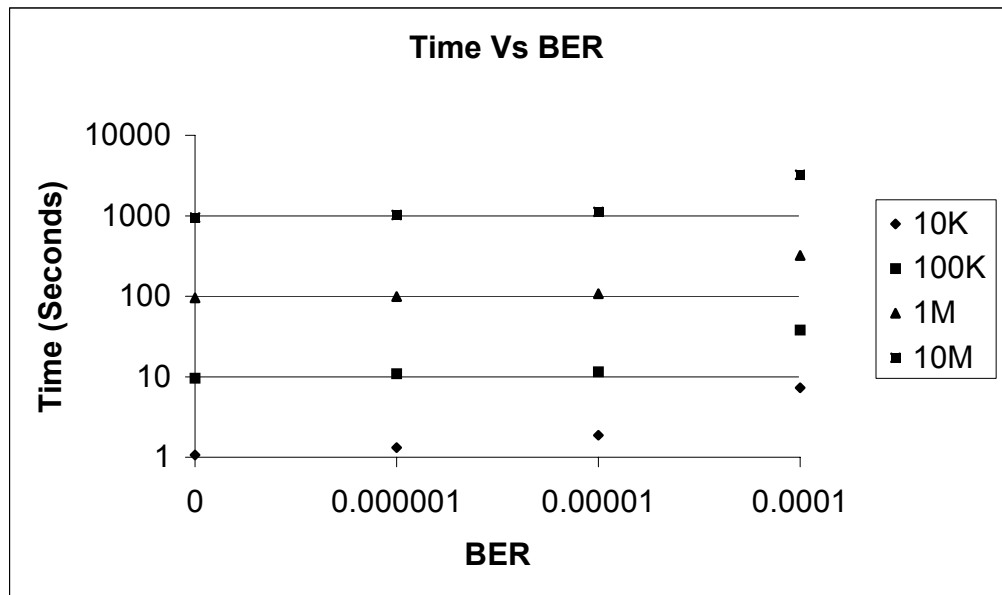


Figure 7: File transfer time versus BER plot for different files using MDP with frame size of 1500 bits.

In Figure 7 the file transfer time of different files are plotted against different BER's using MDP. From this plot we can see that as BER increases the file transfer time increases enormously. The data for the plot in Figure 7 is shown in Table 4.

File Size In Bytes	μ at BER:0	μ at BER:0.000001	μ at BER:0.00001	μ at BER:0.0001
10K	1.0625	1.3125	1.875	7.3125
100K	9.6	10.875	11.5625	38
1M	95.3125	99	107.438	321.56
10M	953.375	1026.2	1118.813	3220.5

Table 4: The average values of file transfer times at different BER's using MDP .

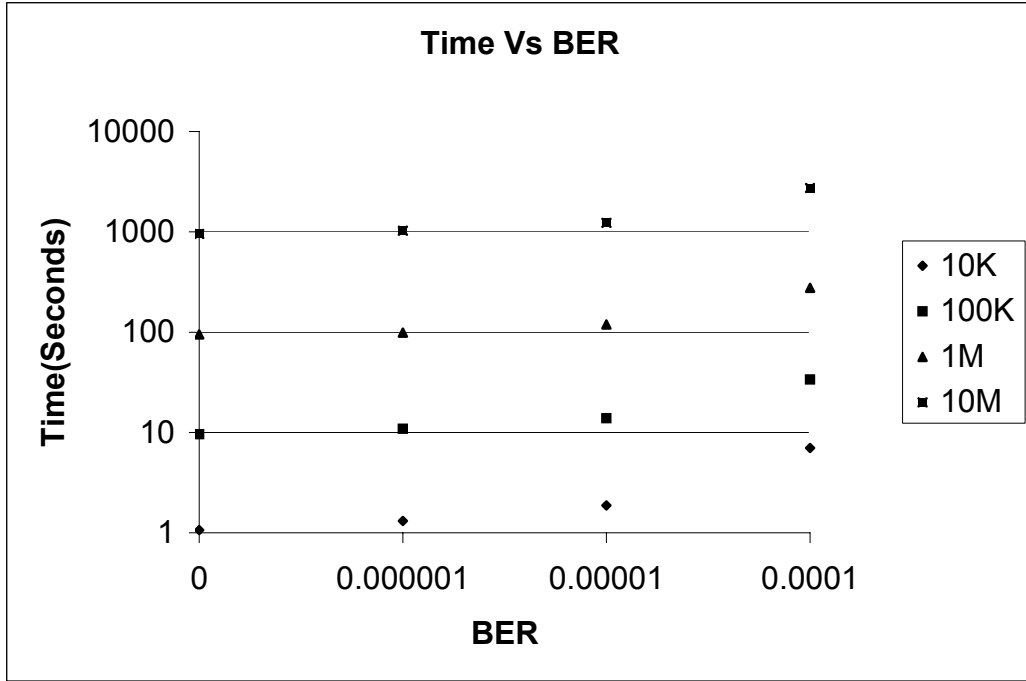


Figure 8: File transfer time versus BER plot for different files using MDP according to the frame sizes given in table 3.

File Size In Bytes	μ at BER:0	μ at BER:0.000001	μ at BER:0.00001	μ at BER:0.0001
10K	1.0625	1.3125	1.875	7
100K	9.6	10.875	13.875	33.625
1M	95.3125	99	119.375	276.875
10M	953.375	1026.2	1227.5625	2736.938

Table 5: The average values of file transfer times at different BER's using MDP at optimal frame size.

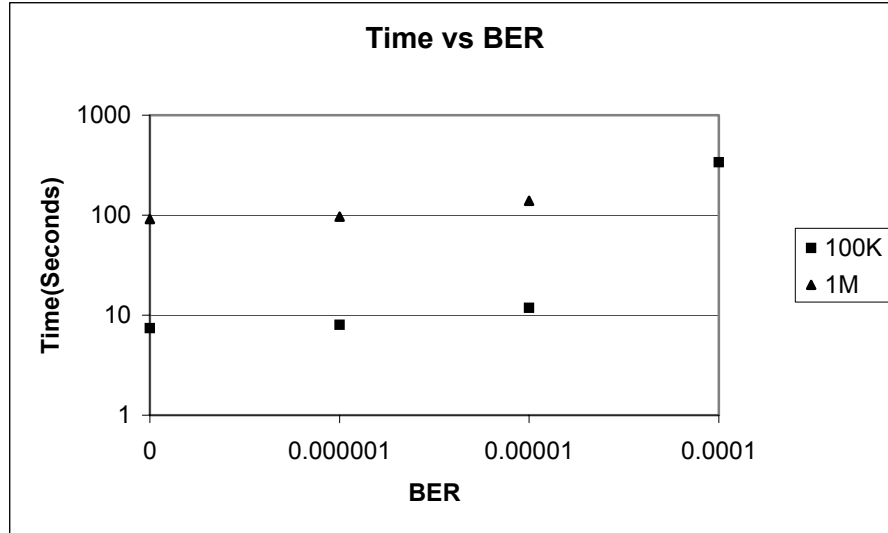


Figure 9: File transfer time versus BER plot for different files using FTP according to frame sizes given in table 3.

File Size In Bytes	μ at BER:0	μ at BER:0.000001	μ at BER:0.00001	μ at BER:0.0001
100K	7.40125	8.0125	11.825	339.0625
1M	91.525	96.9	139.3125	

Table 6: The average values of file transfer times at different BER's using FTP at optimal frame size.

From figure 8 and figure 9 when we compare the file transfer times of FTP and MDP we can see that as the BER increases the increase in file transfer time using FTP is significantly more than the increase in file transfer time using MDP. We also observed that at high BER, FTP stops or takes very long time in transmitting large files (like 1M and 10M). MDP takes much less time than FTP in transmitting large files at high BER.

The experimental throughput can also be measured in terms of the channel data rate. The percentage of channel capacity utilized, T_N , is computed using the following equation

$$T_N = \left[\left\{ \frac{F}{\mu} \right\} / R_b \right] * 100 \quad (10)$$

Where F is the file size, R_b is the channel data rate and μ is the average time to complete the file transfer. The value $\left\{ \frac{F}{\mu} \right\} / R_b$ is called normalized throughput and is unit less and includes all of the protocol overhead time as well as actual time transmitting the actual data files. The channel data rate in our case is 88000 bps.

Figure 10 illustrates the channel utilization capacity of MDP at 1500 bit frame size.

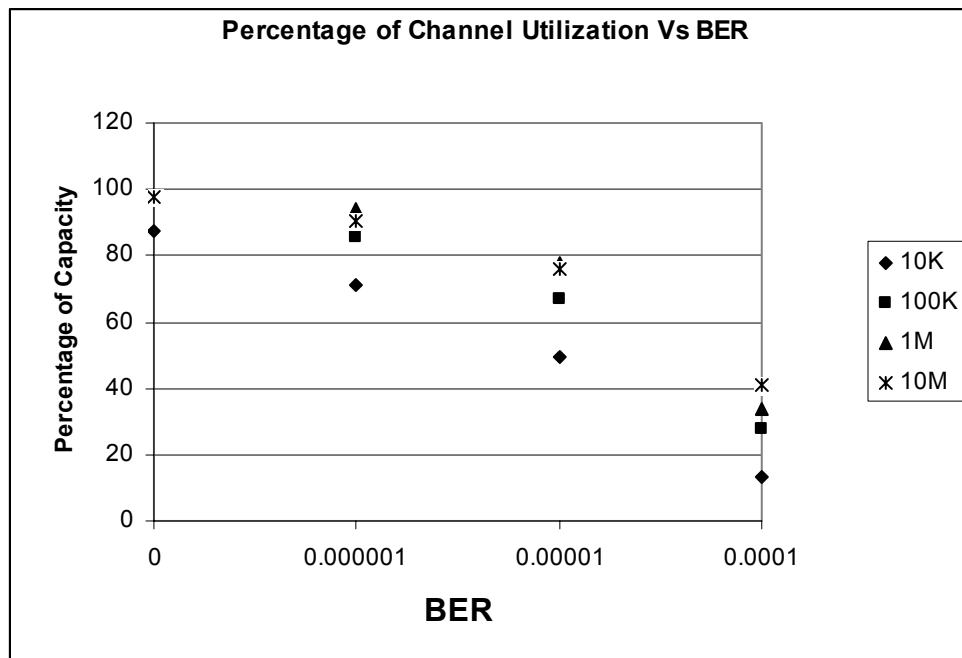


Figure 10: MDP throughput as a percentage of available channel capacity

Advantages of MDP over FTP

- 1) If we look at the Figures 7 and 8, initially the file transfer time of different files at zero BER using FTP is less than that of MDP at zero BER. But as the BER increases the file transfer time using FTP increases significantly than MDP.

- 2) FTP stops transferring large files (1M and 10M) at higher bit error rates whereas MDP can transfer files of any size at higher BER's.

Example Analysis Of Results

The analysis can be done using the experiment data given in Tables 5 and 6. Table 5 contains experiment results for ftp protocol while Table 6 contains experiment results for the Multicast Dissemination Protocol (MDP). By using Equations (1) and (2), the average values and the standard deviations for the measurements are given as the last row for each experiment. At this point we can start investigating a number of questions.

For example, is there an effect due to channel error rate for each protocol? That is, let us compare the 100K file transmission time for 0 channel errors and a channel error rate of 0.0001. The hypothesis H_0 states there is no difference. For the ftp results, we used

sixteen similar workloads on two systems. The observations are $\{(7.3, 233), (7.57, 304), (7.59, 197), (7.56, 675), (7.33, 251), (7.34, 310), (7.32, 272), (7.3, 209), (7.31, 199), (7.32, 697), (7.3, 285), (7.33, 426), (7.33, 203), (7.63, 248), (7.6, 598), (7.29, 318)\}$

The performance differences constitute a sample of sixteen observations, $\{225.7, -296.43, -189.41, -667.44, -243.67, -302.66, -264.68, -201.7, -191.69, -689.68, -277.7, -418.67, -195.67, -240.37, -590.4, -310.71\}$

For this sample:

Sample mean = -331.661

Sample variance = 28525.05899

Sample standard deviation = 168.893632

The confidence interval for mean using equation 3 = $-331.661 \mp t \sqrt{28525.05899/16} = -$

$331.661 \mp t (42.22)$

The 0.95-quantile of a t-variate with 15 degrees of freedom is 1.753

95% confidence interval = $-331.661 \mp (1.753)(42.22) = (-405.6726, -257.6493)$

Table 7: Experiment results for ftp testing

	BER:0		BER : 0.000001		BER:0.00001		BER:0.0001	
	File Size		File Size		File Size		File Size	
	1M	100K	1M	100K	1M	100K	100K	10K
	91	7.3	95.9	7.93	142	11	233	17.2
	91.5	7.57	97.3	8.38	132	13.4	304	5.76
	91.6	7.59	95.4	7.58	137	11.6	197	11
	91.5	7.56	95.3	7.74	135	10.4	675	14
	91.5	7.33	96.8	7.97	135	12.9	251	6.31
	91.2	7.34	94.8	7.34	139	11.3	310	15.8
	91.1	7.32	99.1	8.44	144	11.9	272	7.55
	91.8	7.3	96.1	7.81	144	10.9	209	13.4
	91.7	7.31	96.6	7.52	138	12.9	199	10.6
	91.6	7.32	95.4	8.24	134	10.9	697	11.8
	91.8	7.3	97.9	8.12	139	12.9	285	16.4
	91.8	7.33	96.3	7.74	151	10.9	426	8.71
	91.5	7.33	96.9	8.06	143	13.2	203	13
	91.7	7.63	96.7	9.25	131	11.6	248	9.04
	92	7.6	100	8.44	139	12.7	598	16
	91.1	7.29	99.9	7.64	146	10.7	318	9.37
μ	91.525	7.40125	96.9	8.0125	139.3125	11.825	339.0625	11.62125
s	0.290975	0.13286	1.596246	0.470199	5.424866	1.017513	168.9266	3.670818
$(\sum t)^2$	2144467.36	14023.2964	2403740.16	16435.24	4968441	35796.64	29430625	34573.68
$\sum t^2$	134030.48	876.7208	150271.98	1030.5188	310969	2252.82	2267457	2362.9788

Table 8: Experiment results for MDP testing

	BER:0		BER : 0.000001		BER:0.00001		BER:0.0001	
	File Size		File Size		File Size		File Size	
	1M	100K	1M	100K	1M	100K	100K	10K
	96	9	98	10	120	15	50	15
	95	9	100	11	119	14	38	7
	95	10	101	11	119	13	38	8
	96	10	99	11	119	14	42	6
	95	10	98	10	119	15	44	5
	95	10	99	11	118	15	31	4
	95	10	99	11	120	14	30	13
	96	9	99	12	117	13	26	14
	95	10	100	11	123	13	29	10
	95	10	98	11	119	12	29	9
	95	9	100	11	120	17	36	5
	95	10	99	10	120	15	31	4
	96	10	98	10	118	14	33	3
	95	9	99	11	119	13	29	3
	95	9	99	11	120	13	27	3
	96	9	98	12	120	12	25	3
μ	95.3125	9.5625	99	10.875	119.375	13.875	33.625	7
s	0.463512	0.512348	0.894427	0.599479	1.31021	1.31021	7.10750	4.1150
$(\sum t)^2$	2325625	23409	2509056	30276	3648100	49284	289444	12544
$\sum t^2$	145355	1467	156828	1898	228032	3106	18848	1038

Since the region does not include the origin, there is a significant difference between the two cases.

Next we apply the Honestly Significant Difference between the two protocols(FTP and MDP) to the case of zero channel errors and a channel BER of 10^{-5} . The results of the required computations from Equations 5,6, and 7 are summarized in Table 7. If we next apply Equation 4, we can see that in the case of zero channel errors, the difference in the means, 2.1625, exceeds the designated value 1.14, hence the hypothesis H_0 is rejected and there is a significant difference between the two experiments. For the case of 10^{-5} channel BER also the difference in means 2.05 exceeds the designated value 0.84, so we conclude that there is a significant difference between two experiments.

BER	T	A	MS_e	q	$ \mu_A - \mu_B $
0	2262.721	2339.52	2.55	2.88	2.1625
0.00001	5358.82	5317.54	1.376	2.88	2.05
0.0001	2286305	1857504.313	14293.356	2.88	305.4375

Table 9: Example computations for Honestly Significant Difference

Finally, let us look at how both of the techniques compare when comparing the performance of 100K file when the channel error rate is 0.0001. The null hypothesis, H_0 , is that there is no difference in FTP and MDP protocols. First we compute the result using the confidence interval method.

Here, the observations are $\{(50, 233), (38, 304), (38, 197), (42, 675), (44, 251), (31, 310), (30, 272), (26, 209), (29, 199), (29, 697), (36, 285), (31, 426), (33, 203), (29, 248), (27, 598), (25, 318)\}$

The performance differences constitute a sample of sixteen observations, $\{-183, -266, -159, -633, -207, -279, -242, -183, -170, -668, -249, -395, -170, -219, -571, -293\}$

For this sample:

Sample mean = -305.438

Sample variance = 28812.39583

Sample standard deviation = 169.742

The confidence interval for mean = $-303.438 \mp t \sqrt{\frac{28812.39583}{16}} = -305.438$
 $\mp t(42.43)$

The 0.95-quantile of a t-variate with 15 degrees of freedom is 1.753

95% confidence interval = $-303.438 \mp (1.753)(42.43) = (-377.817, -229.058)$

Since this interval does not include the origin, the two protocols are seen to have significant differences. The Honestly Significant Difference method has its computations summarized in Table 7 as well. Using these numbers, the criterion for the difference in the means to exceed in order to reject the null hypothesis, H_0 is 86.07. Since the difference in the means is 305.4375, the null hypothesis, H_0 is rejected using this test as well for this experiment.

Discussion

In the first case for the experiment results of FTP we can see a statistically meaningful difference in the means of the clean channel and a corrupted channel for a given file, hence we can say that the introduction of BER in the channel effects the performance of FTP. In the second case both the clean channel and corrupted channel showed statistically meaningful difference in the means of FTP and MDP.

In the final case with 95% confidence interval we can say that the mean of file transfer time of 100K file using FTP is significantly different from the mean of file transfer time of 100K file using MDP at a channel BER of 0.0001.

Conclusion

The results show that at zero BER MDP utilizes around 97.6% of the available channel capacity for a 10MB file and at 0.0001 BER for the same file size MDP uses 41.2% of the channel capacity, which is good in such a high BER.

If we compare the performance of a 100KB file at a BER of 0.0001 using MDP and FTP statistically, we can see that the 95% confidence interval (-377.817, -229.058) does not include zero. Which means the two systems are significantly different. Hence we can say with 95% confidence that MDP and FTP file transfer times are significantly different and if we see the graphs in Figure 8 and Figure 9 we can say that performance of MDP is better than FTP at high BER's like 0.0001. Also FTP does not work if there is any link cut-off as it is a connection oriented protocol whereas MDP is unaffected by link cut-off as it is a connection less protocol. MDP shows better performance with optimal frame size than with the default frame size.

References

- [1] Multicast Dissemination Protocol version 2 (MDPv2) by [Joe Macker_\(NRL\)](#) and [Brian Adamson \(Newlink\)](#)
URL< <http://manimac.itd.nrl.navy.mil/MDP/>>
- [2] S. Horan and R. H. Wang, “Design of a Channel Error Simulator Using Virtual Instrument Techniques for the Initial Testing of TCP/IP and SCPS Protocols,” NMSU-ECE-99-002, April 1999.
- [3] William J. Atsma, “Serial Laplink HOWTO.”
URL<<http://www.ibiblio.org/pub/Linux/docs/HOWTO/Serial-Laplink-HOWTO>> 2001-07-20.
- [4] S.Horan and R. H. Wang, “The Behavior of TCP and Its Extensions in Space,” NMSU-ECE-01-008
- [5] W. R. Stevens, TCP/IP Illustrated, Volume 1: The Protocols. Addison Wesley, 1994.
- [6] S. Horan, “Protocol Testing Procedures,” NMSU-ECE-02-002

Appendices

Appendix 1:File transfer times for different file sizes at zero BER using MDP

Sequence Number	File Size	Time in Seconds	Sequence Number	File Size	Time in Seconds
1	100K1	9	33	10K10	1
2	1M2	96	34	10M2	954
3	100K15	9	35	10K14	1
4	10M12	953	36	10K4	1
5	100K14	10	37	10K9	1
6	10M6	952	38	100K13	10
7	100K10	10	39	10K2	1
8	10M10	954	40	10K7	1
9	10M16	953	41	10M15	954
10	100K11	10	42	1M13	95
11	1M4	95	43	1M12	95
12	10K15	1	44	10M7	953
13	10M13	953	45	100K6	10
14	1M8	95	46	100K9	9
15	1M7	96	47	10K5	1
16	10K8	1	48	100K12	10
17	10K11	1	49	10M5	953
18	1M11	95	50	1M15	95
19	10M9	954	51	1M1	95
20	1M14	95	52	10K6	1
21	10K13	1	53	1M5	96
22	100K5	10	54	100K4	9

23	100K8	9	55	10M3	954
24	100K7	10	56	1M6	95
25	1M9	95	57	1M10	95
26	100K2	10	58	10K1	1
27	100K16	10	59	1M3	96
28	1M16	96	60	10K16	1
29	10K3	1	61	100K3	9
30	10K12	1	62	10M4	954
31	10M11	953	63	10M14	953
32	10M8	953	64	10M1	954

Appendix 2:File transfer times for different file sizes at 10^{-6} BER using MDP

Sequence Number	File Size	Time in Seconds	Sequence Number	File Size	Time in Seconds
1	100K1	10	33	10K10	1
2	1M2	98	34	10M2	1032
3	100K15	11	35	10K14	2
4	10M12	1029	36	10K4	1
5	100K14	11	37	10K9	1
6	10M6	1026	38	100K13	11
7	100K10	11	39	10K2	1
8	10M10	1031	40	10K7	1
9	10M16	1027	41	10M15	1026
10	100K11	10	42	1M13	100
11	1M4	100	43	1M12	98
12	10K15	1	44	10M7	1026
13	10M13	1031	45	100K6	10
14	1M8	101	46	100K9	10
15	1M7	99	47	10K5	1
16	10K8	1	48	100K12	11
17	10K11	1	49	10M5	1025
18	1M11	98	50	1M15	100
19	10M9	1024	51	1M1	99
20	1M14	99	52	10K6	0
21	10K13	1	53	1M5	98
22	100K5	11	54	100K4	11
23	100K8	11	55	10M3	1017
24	100K7	12	56	1M6	99

25	1M9	99	57	1M10	99
26	100K2	11	58	10K1	4
27	100K16	11	59	1M3	98
28	1M16	99	60	10K16	1
29	10K3	1	61	100K3	12
30	10K12	2	62	10M4	1011
31	10M11	1030	63	10M14	1023
32	10M8	1035	64	10M1	1018

Appendix 3: File transfer times for different file sizes at 10^{-5} BER using MDP

Sequence Number	File Size	Time in Seconds	Sequence Number	File Size	Time in Seconds
1	100K1	13	33	10K10	2
2	1M2	108	34	10M2	1110
3	100K15	12	35	10K14	2
4	10M12	1230	36	10K4	2
5	100K14	13	37	10K9	1
6	10M6	1112	38	100K13	11
7	100K10	12	39	10K2	2
8	10M10	1115	40	10K7	3
9	10M16	1109	41	10M15	1109
10	100K11	12	42	1M13	107
11	1M4	107	43	1M12	108
12	10K15	2	44	10M7	1114
13	10M13	1110	45	100K6	12
14	1M8	107	46	100K9	11
15	1M7	108	47	10K5	2
16	10K8	1	48	100K12	11
17	10K11	3	49	10M5	1111
18	1M11	108	50	1M15	108
19	10M9	1111	51	1M1	108
20	1M14	107	52	10K6	2
21	10K13	2	53	1M5	107
22	100K5	11	54	100K4	11
23	100K8	11	55	10M3	1112
24	100K7	12	56	1M6	107

25	1M9	107	57	1M10	108
26	100K2	11	58	10K1	1
27	100K16	11	59	1M3	107
28	1M16	107	60	10K16	1
29	10K3	2	61	100K3	11
30	10K12	2	62	10M4	1109
31	10M11	1111	63	10M14	1112
32	10M8	1114	64	10M1	1112

Appendix 4: File transfer times for different file sizes at 10^{-5} BER using MDP at 650 bit frame size

Sequence Number	File Size	Time in Seconds	Sequence Number	File Size	Time in Seconds
1	100K1	15	33	10K10	2
2	1M2	120	34	10M2	1235
3	100K15	14	35	10K14	2
4	10M12	1223	36	10K4	2
5	100K14	13	37	10K9	2
6	10M6	1219	38	100K13	17
7	100K10	14	39	10K2	2
8	10M10	1221	40	10K7	1
9	10M16	1235	41	10M15	1248
10	100K11	15	42	1M13	123
11	1M4	119	43	1M12	119
12	10K15	2	44	10M7	1221
13	10M13	1230	45	100K6	15
14	1M8	119	46	100K9	14
15	1M7	119	47	10K5	1
16	10K8	1	48	100K12	13
17	10K11	6	49	10M5	1232
18	1M11	119	50	1M15	120
19	10M9	1224	51	1M1	120
20	1M14	118	52	10K6	1
21	10K13	1	53	1M5	118
22	100K5	15	54	100K4	13
23	100K8	14	55	10M3	1232

24	100K7	13	56	1M6	119
25	1M9	120	57	1M10	120
26	100K2	13	58	10K1	2
27	100K16	12	59	1M3	120
28	1M16	117	60	10K16	1
29	10K3	2	61	100K3	12
30	10K12	2	62	10M4	1219
31	10M11	1228	63	10M14	1218
32	10M8	1232	64	10M1	1224

Appendix 5: File transfer times for different file sizes at 10^{-4} BER using MDP

Sequence Number	File Size	Time in Seconds	Sequence Number	File Size	Time in Seconds
1	100K1	51	33	10K10	4
2	1M2	318	34	10M2	3222
3	100K15	36	35	10K14	6
4	10M12	3198	36	10K4	6
5	100K14	40	37	10K9	9
6	10M6	3119	38	100K13	37
7	100K10	40	39	10K2	7
8	10M10	3228	40	10K7	9
9	10M16	3233	41	10M15	3244
10	100K11	38	42	1M13	320
11	1M4	326	43	1M12	317
12	10K15	8	44	10M7	3239
13	10M13	3235	45	100K6	22
14	1M8	320	46	100K9	43
15	1M7	328	47	10K5	12
16	10K8	5	48	100K12	37
17	10K11	8	49	10M5	3224
18	1M11	319	50	1M15	316
19	10M9	3191	51	1M1	322
20	1M14	322	52	10K6	8
21	10K13	5	53	1M5	318
22	100K5	35	54	100K4	35
23	100K8	39	55	10M3	3220
24	100K7	36	56	1M6	330

25	1M9	317	57	1M10	320
26	100K2	39	58	10K1	7
27	100K16	38	59	1M3	331
28	1M16	321	60	10K16	10
29	10K3	7	61	100K3	42
30	10K12	6	62	10M4	3217
31	10M11	3251	63	10M14	3252
32	10M8	3215	64	10M1	3240

Appendix 6:File transfer times for different file sizes at 10^{-4} BER using MDP at 128 bit frame size

Sequence Number	File Size	Time in Seconds	Sequence Number	File Size	Time in Seconds
1	100K1	50	33	10K10	13
2	1M2	323	34	10M2	2719
3	100K15	38	35	10K14	14
4	10M12	2747	36	10K4	10
5	100K14	38	37	10K9	9
6	10M6	2727	38	100K13	36
7	100K10	42	39	10K2	5
8	10M10	2740	40	10K7	4
9	10M16	2745	41	10M15	2742
10	100K11	44	42	1M13	279
11	1M4	312	43	1M12	284
12	10K15	15	44	10M7	2704
13	10M13	2751	45	100K6	31
14	1M8	285	46	100K9	33
15	1M7	271	47	10K5	3
16	10K8	7	48	100K12	29
17	10K11	8	49	10M5	2736
18	1M11	264	50	1M15	293
19	10M9	2726	51	1M1	263
20	1M14	269	52	10K6	3
21	10K13	6	53	1M5	261
22	100K5	31	54	100K4	27
23	100K8	30	55	10M3	2758
24	100K7	26	56	1M6	276

25	1M9	266	57	1M10	268
26	100K2	29	58	10K1	3
27	100K16	29	59	1M3	260
28	1M16	256	60	10K16	3
29	10K3	5	61	100K3	25
30	10K12	4	62	10M4	2733
31	10M11	2763	63	10M14	2749
32	10M8	2744	64	10M1	2707

Appendix 7: File transfer times for different file sizes at zero BER using FTP

Sequence Number	File Size	Time in Seconds
1	100K1	7.3
2	1M2	91
3	100K15	7.57
4	100K14	7.59
5	100K10	7.56
6	100K11	7.33
7	1M4	91.5
8	1M8	91.6
9	1M7	91.5
10	1M11	91.5
11	1M14	91.2
12	100K5	7.34
13	100K8	7.32
14	100K7	7.3
15	1M9	91.1
16	100K2	7.31
17	100K16	7.32
18	1M16	91.8
19	100K13	7.3
20	1M13	91.7
21	1M12	91.6
22	100K6	7.33
23	100K9	7.33
24	100K12	7.63

25	1M15	91.8
26	1M1	91.8
27	1M5	91.5
28	100K4	7.6
29	1M6	91.7
30	1M10	92
31	1M3	91.1
32	100K3	7.29

Appendix 8: File transfer times for different file sizes at 10^{-6} BER using FTP

Sequence Number	File Size	Time in Seconds
1	100K1	7.93
2	1M2	95.9
3	100K15	8.38
4	100K14	7.58
5	100K10	7.74
6	100K11	7.97
7	1M4	97.3
8	1M8	95.4
9	1M7	95.3
10	1M11	96.8
11	1M14	94.8
12	100K5	7.34
13	100K8	8.44
14	100K7	7.81
15	1M9	99.1
16	100K2	7.52
17	100K16	8.24
18	1M16	96.1
19	100K13	8.12
20	1M13	96.6
21	1M12	95.4
22	100K6	7.74
23	100K9	8.06
24	100K12	9.25

25	1M15	97.9
26	1M1	96.3
27	1M5	96.9
28	100K4	8.44
29	1M6	96.7
30	1M10	100
31	1M3	99.9
32	100K3	7.64

Appendix 9: File transfer times for different file sizes at 10^{-5} BER using FTP

Sequence Number	File Size	Time in Seconds
1	100K1	11
2	1M2	142
3	100K15	13.4
4	100K14	11.6
5	100K10	10.4
6	100K11	12.9
7	1M4	132
8	1M8	137
9	1M7	135
10	1M11	135
11	1M14	139
12	100K5	11.3
13	100K8	11.9
14	100K7	10.9
15	1M9	144
16	100K2	12.9
17	100K16	10.9
18	1M16	144
19	100K13	12.9
20	1M13	138
21	1M12	134
22	100K6	10.9
23	100K9	13.2
24	100K12	11.6

25	1M15	139
26	1M1	151
27	1M5	143
28	100K4	12.7
29	1M6	131
30	1M10	139
31	1M3	146
32	100K3	10.7

Appendix 10: File transfer times for different file sizes at 10^{-4} BER using FTP

Sequence Number	File Size	Time in Seconds
1	100K1	233
2	100K15	304
3	100K14	197
4	100K10	675
5	100K11	251
6	10K15	17.2
7	10K8	5.76
8	10K11	11
9	10K13	14
10	100K5	310
11	100K8	272
12	100K7	209
13	100K2	199
14	100K16	697
15	10K3	6.31
16	10K12	15.8
17	10K10	7.55
18	10K14	13.4
19	10K4	10.6
20	10K9	11.8
21	100K13	285
22	10K2	16.4
23	10K7	8.71
24	100K6	426

25	100K9	203
26	10K5	13
27	100K12	248
28	10K6	9.04
29	100K4	598
30	10K1	16
31	10K16	9.37
32	100K3	318

Acronyms

BER	Bit Error Rate
FTP	File Transfer Protocol
H	Hypothesis
HSD	Honestly Significant Difference
N	the number of measurements in a data set
IP	Internet Protocol
MDP	Multicast Dissemination Protocol
NACK	Negative Acknowledgement
NMSU	New Mexico State University
q	Critical value for a studentized distribution
s	standard deviation in a set of measurements
PPP	Point-to-Point Protocol
SGLS	Space to Ground Link Simulator
t	Critical value for a t-distribution
t_i	time measurement for an experiment run
TCP	Transmission Control Protocol
z	Critical value for a normal distribution
α	significance level
μ	mean value for a set of measurements
ν	number of degrees of freedom
UDP	User Datagram Protocol