PERFORMANCE OF RELIABLE MULTICAST PROTOCOLS VIA SATELLITE AT EHF WITH PERSISTENT FADES

M. Koyabe and G. Fairhurst
Electronics Research Group
University of Aberdeen
King's College, Old Aberdeen
Aberdeen, AB24 3UE
Scotland

Tel: +44-(0)1224-272813
Fax: +44-(0)1224-272497
{koyabe, gorry}@erg.abdn.ac.uk

Abstract

Most of the existing reliable multicast transport (RMT) protocol techniques that are suited for satellite networks adopt various design techniques that may limit or enhance their scalability to large receiver groups in one-to-many topologies. Satellite networks (with high bandwidth capacity and multi-spot beam support at the Extra High Frequency (EHF) spectrum), may provide a complementary solution to scalability although at a cost. Prolonged link fades due to varying weather patterns predominantly cause packet loss and unnecessary retransmission (form source or receiver) when link is unavailable. Analysis of RMT reliability techniques over links with persistent fade, shows that hybrid Forward Error Correction (FEC) with Automatic Repeat Request (ARQ) show better performance and efficiency in terms of robustness, goodput and redundancy.

Key Words: Internet, Protocols, Fades, Satellite, Multicast

1 Introduction

The subject of reliable multicast transport (RMT) protocols [1, 2] has generated a lot of research interest [3]. Most of the existing RMT protocol techniques that are suited for satellite networks [4], adopt one of a set of design paradigms that either limit or enhance their scalability to large groups of receivers. Despite concerted efforts by researchers to define common techniques suited for varying topologies (such as one-to-many) [5], limited research work has been conducted to evaluate and analyse the performance of these RMT protocol techniques over satellite links with persistent fades.

Satellite networks [6], intrinsically, may offer a complementary alternative solution in enabling these classes of reliable multicast techniques to scale to large group of receivers [4]. High bandwidth availability (at the Extra High Frequency (EHF) spectrum) and the broadcast nature of satellite networks still motivates most content distributors and network providers to switch to satellite networks for service provision [7]. Satellites networks (especially in the single-hop case) also suffer less congestion between the sender and receivers (since there are significantly few or at times no intermediate routers causing congestion), as compared to terrestrial networks.

Apart from the inherent GEO satellite delay (≈250 ms) and link bit-error rates, reliable multicast support over satellite systems operating at EHF still poses some challenges. At EHF, satellite link channels (both uplink and downlink) can from time-to-time experience persistent link fades (caused by varying weather patterns). Variations in these weather patterns may cause either short or long fades on both the uplink and downlink channels [8, 9]. More so, persistent link fades (on both the uplink and downlink channels) may exacerbate performance degradation due to data loss and unnecessary transmissions (from either sender or receiver) when the links are unavailable.

The aim of this paper, therefore, is to evaluate and analyse (using simulation) the performance of reliable multicast techniques (that are suited for satellite networks) over satellite topologies experiencing persistent fades and linked to very large groups of receivers. The rest of this paper is organised as follows: in the next section we briefly discuss related research work that has been conducted. Section 3 outlines the three sub-classes of RMT protocol techniques that are suited for satellite networks. In section 4 we briefly describe our fade error model which was integrated in the simulator, whereas in section 5 we present performance parameters and experiment scenario used to generate our results. In section 6 we present our analysis results followed by conclusions and recommendation for further research work.

2 Related research work

A number of research studies have motivated our work in this paper. These studies have mainly been focused at evaluating or analysing performance of transport protocols over satellites or wireless links [10-14]. The transport Control Protocol (TCP) [15], which is a reliable unicast transport protocol, has been used mostly in these research studies over varying end-to-end network topologies. The advantage of using TCP in these studies has been attributed mainly to TCP's well-defined and better-understood protocol behaviour. Similar studies, although not over wireless links, have also been carried out on RMT protocols [11, 16-19]. A significant number of these protocols, however, have matured with time and are increasingly being deployed. There is also an on-going effort within the Internet Engineering Task Force (IETF) to standardise different classes of existing RMT [3].

3 Reliable Multicast Protocols

Several classes of RMT techniques have been proposed to the IETF for standardisation [3]. In this paper we compare the performance of RMT (i.e. one-to-many) that are suited for satellite networks [4]. In our work we examine examples of several fundamentally different protocols: Multicast File Transport Protocol (MFTP) [K. Robertson, 1998 #25], Multicast File Transport Protocol with Erasure Correction (MFTP/EC) [20] and a theoretical analytical File Transfer Protocol (aFTP) with similar characteristics to the digital fountain (DF) protocol [21]. The other reason for choosing these protocols was the availability of both the source code and implementation in the ns-Simulator [22]. All the three reliable multicast protocol sub-classes adopt a receiver-initiated error recovery schemes, but differ in the way feedback traffic is minimised. A brief description of these protocols follows next.

3.1 Multicast File Transport Protocol (MFTP)

Multicast File Transport Protocol (MFTP) [K. Robertson, 1998 #25], is a one-to-many reliable multicast protocol. MFTP uses a check-point based automatic repeat request (ARQ) reliability scheme [23]. The protocol partitions a file into packets and builds equally-sized blocks of packets. Each block consists of as many packets as bits fit into an IP-packet of maximum size. The source initially multicasts all packets (equivalent to the whole file) in the first pass (or data transmission phase). Receivers start receiving data packets and note the missing packets. After each file transfer pass, receivers send a NACK-bitmap listing the status (received/missed) for each data packet sent within the block. The sender collects all the NACK-packets and determines the set of data packets that were not correctly received and later retransmits in the subsequent file transfer pass. This procedure continues until all receivers have completely received the file. Receivers leave the multicast group as soon as they complete file reception.

3.2 Multicast File Transport Protocol with Erasure Correction (MFTP/EC)

Multicast File Transport Protocol with Erasure Correction (MFTP/EC) [16] is a continuous based ARQ reliable multicast protocol similar to MFTP. MFTP/EC uses erasure correction methods to perform loss recovery more efficiently. Both the sender and receivers partition the set of data packets into groups of ***k*** packets each and build blocks of ***n*** groups (where ***n*** > 10,000). Like MFTP, receivers in MFTP/EC send back NACK-bitmaps after a complete file transmission indicating whether or not all packets of a group have been received successfully. In the second and subsequent passes, the sender multicasts one redundancy packet for each NACK'ed group. Ideally, this makes the recovery passes in MFTP/EC only last up to a fraction $1/k$ of a full MFTP pass (although in general, MFTP/EC would need more passes than the basic MFTP protocol) [17]. In our simulation work we set the value of k = 32 for MFTP/EC. At the extreme case when k = 1, MFTP/EC effectively becomes MFTP.

3.3 Analytical File Transfer Protocol (aFTP)

aFTP is an analytical file transfer protocol used to compute mathematically the transfer of packets from a single source to several receivers This is representative of a simplex RMT protocol. In aFTP algorithm, the following functionality and assumptions were made:-

- The sender transmits the same file in a carousel manner after each subsequent *round* or *pass*[1], for infinite number of times until all the receivers successfully get the whole file.
- All receivers register to receive data packets at the same time and there is no feedback data (ACKs or NACKs) generated by receivers
- The time between the end and start of subsequent passes was negligible, and therefore each scheduled pass started immediately after the completion of the previous transmission pass.
- All packets were transmitted at the set data rate and there was no appreciable delay or jitter over the link due to congestion (either at receiver or sender) and all transmitted packets experienced no other loss except loss due to fades over the satellite link channels
- Receivers experiencing packet loss, resumed file reception at the end of each fade duration until the whole file was successful received.

4 Fade Error Model

The fade error model was based on a time-series of packet error rate (PER) derived from a time-series of E_b/N_0 provided by ONERA [8]. This fade error model was used to accurately estimate the occurrence and duration of a fade event (when all or no packets are lost) based on the packet error rate time-series.

In this model we consider a time-series data set taken over a period of time (i.e. days, months or even years) with minimum time-series time (*ts_min*) and maximum time-series time (*ts_max*) measured in seconds as shown in figure (1) below. Let the fade events *fade(i)*, *fade(i+1)*, *fade(i+2)*, … , *fade(n)* occur at times *fade_start_time(i)*, *fade_start_time(i+1)*, *fade_start_time(i+2)*, … , *fade_start_time(n)* with fade periods *fade_period(i)*, *fade_period(i+1)*, *fade_period(i+2)*, … , *fade_period(n)* for *i* = 1,2,3,4,..,n respectively.

We assume that the effect of fade mitigation techniques (FMT) on transition time between *no fade* state-to-*fade* state and *fade* state-to-*no fade* state is very small and therefore negligible [8]. In order to estimate accurately uncorrelated fades experienced at different weather cells, each initial starting time (on the Time-series) was randomly chosen over the entire time-series interval [*ts_min*, *ts_max*]. In this model, it was further assumed that each fade occurrence over the time-series interval lasted for a short duration of not more than one minute (~ 60 secs) similar to assumption made in [9].

To illustrate how the model works, let position ***A***, ***B***, ***C*** and ***D***, shown in figure (1) above represent possible initial start time positions, that can randomly be chosen in the interval [*ts_min*, *ts_max*]. ***A***

---

[1] In this work, we interchangeably use ***round*** or ***pass*** to refer to the number of subsequent transmission or retransmission of data from sender to receivers.

represents the start time before start of first fade; ***B*** represents start time within a fade phase; ***C*** represents start time between two fades and ***D*** represents start time after the last fade.

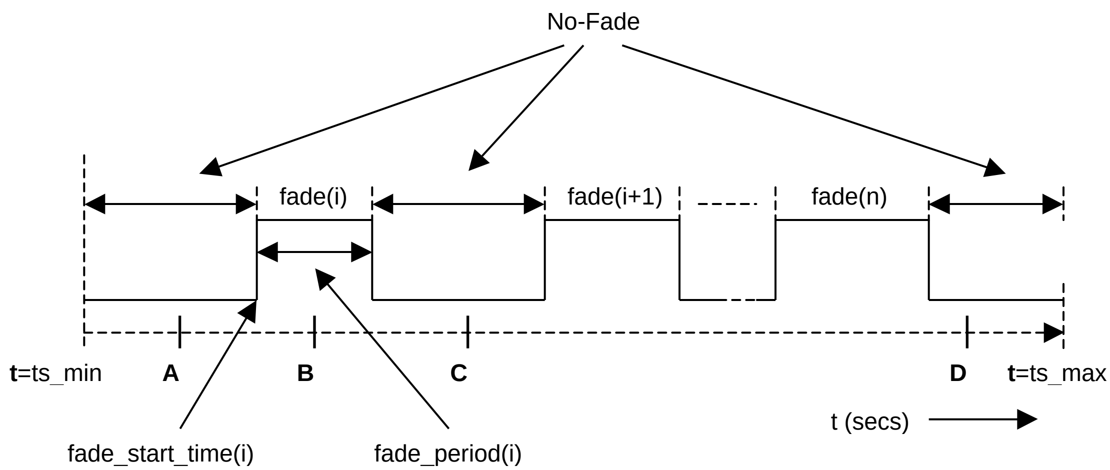

Figure 1: Sketch of fade occurrence and duration based on PER Time-Series

At the start of every session the algorithm keeps track of three parameters—current start time (*start_time*) which marks the start of the fade pattern on the time-series; the next fade duration (*next_fade_duration*); the next inter-fade duration (*next_no-fade_duration*)—all dependent on both fade periods (*fade_periods(i)*) and fade start times (*fade_start_time(i)*), *i* = 1,2,3,…,n over the time-series interval [*ts_min*, *ts_max*]. The initial start time is randomly generated at the start of the simulation (based on a heuristic chosen seed value). Next we show how the algorithm computes and determines the next fade and inter fade duration during simulation for the four cases ***A***, ***B***, ***C*** and ***D*** respectively.

Case A: when {*start_time* < *fade_start_time(1)*}
next_fade_duration = *fade_period(1)*
next_no-fade_duration = *fade_start_time(1)* – *start_time*
start_time = *fade_start_time(1)* + *fade_period(1)*

Case B: when {*start_time* >= *fade_start_time(i)* && *start_time* < (*fade_start_time(i)* + *fade_period(i)*)}
next_fade_duration = (*fade_start_time(i)* + *fade_period(i)*) – *start_time*
next_no-fade_duration = 0.0
start_time = *fade_start_time(i)* + *fade_period(i)*

Case C: when {*start_time* >= (*fade_start_time(i)* + *fade_period(i)*) && *i* != *n*}
next_fade_duration = *fade_period(i+1)*
next_no-fade_duration = *fade_start_time(i+1)* – *start_time*
start_time = *fade_start_time(i+1)* + *fade_period(i+1)*

Case D: when {*start_time* >= (*fade_start_time(n)* + *fade_period(n)*)}
next_fade_duration = *fade_period(1)*
next_no-fade_duration = (*ts_max* – *start*_time) + (*fade_start_time(1)* – *ts_min*)
start_time = *fade_start_time(1)* + *fade_period(1)*

In the case of position D, if the data transfer period exceeds the time-series interval, the algorithm wraps around the time-series to restart the process from the first fade. The main reason for wrapping around was to ensure consistency of the time-series recorded for a particular region.

5 Performance parameters and experiment Scenario

In this section we define some of the performance parameters and metrics used to analyse the results. Other researchers (e.g. [12, 17, 24]) have used most of these definitions in analysing similar experiments.

Duplicate Rate = [*Total number of duplicate packets received / Total number of receivers*]
Useful Packets = [*Data packets received that are not duplicates]*
Goodput[2] = [*Total useful data received (Bytes) / Total data transmitted by source (Bytes)*]
Cumulative % of Complete Receivers = (*Current total number of Complete Receivers / Total number of receivers) X 100*

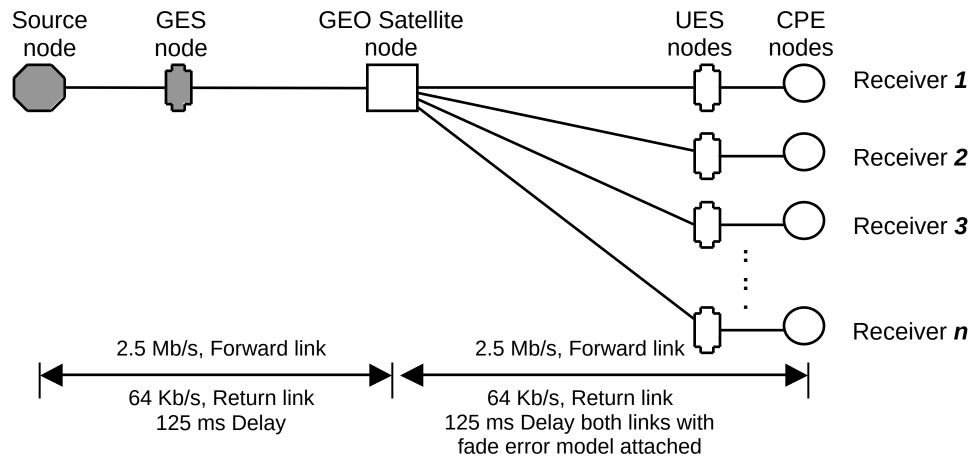

Figure 2: Simulation scenario of one-to-many network topology

Figure (2) above shows the scenario implemented in the ns-Simulator. The source node is connected to a Ground Earth Station (GES) is linked to receivers via a GEO satellite. The receivers, which in this case are Customer Premise Equipments (CPEs), are connected to a User Earth Station (UES) which is linked to the GEO satellite. The forward link and the return link were set to 2.5Mb/s and 64Kb/s respectively and subjected to persistent fade through our ns implementation of the fade error model.

6 Analysis Results

6.1 Reliability and robustness

Figures (3) and (4) show the percentage of complete receivers (as reliability and robustness measure) over file transfer time between aFTP, MFTP and MFTP/EC with increasing number of loss receivers and different packet sizes. In both cases, the MFTP/EC had a high completion rate after the first pass than aFTP and MFTP. In MFTP/EC, repair packets are transmitted during file transfer pass, unlike in aFTP and MFTP, where receivers have to wait for a whole pass before requesting/receiving repair packets.

[2] When analysing goodput for this work we consider the time interval to be minimum time between passes which is equivalent to a file transfer time.

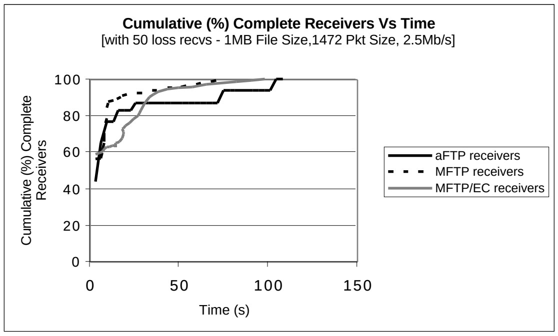

Figure 3(a): Completion performance with 50 loss receivers using 1472 Bytes packet size

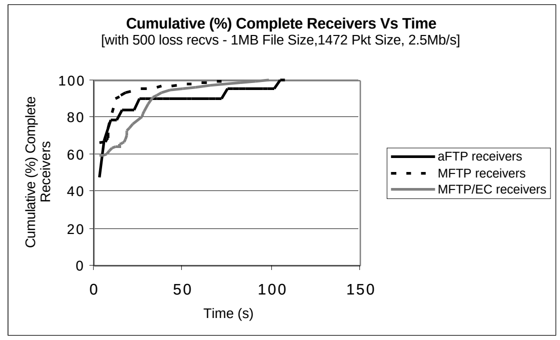

Figure 3(b): Completion performance with 500 loss receivers using 1472 Bytes packet size

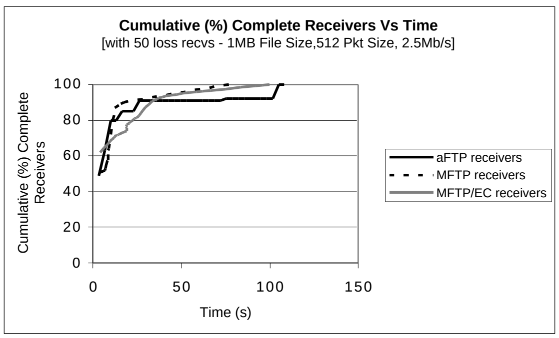

Figure 4(a): Completion performance with 50 loss receivers using 512 Bytes packet size

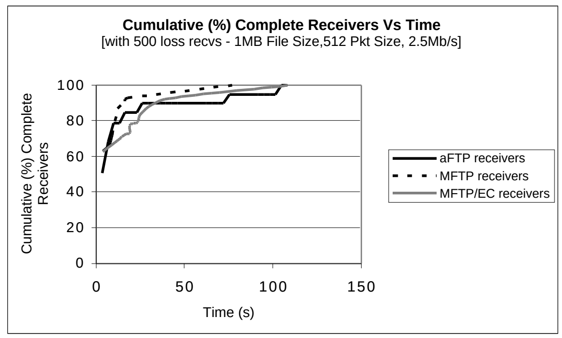

Figure 4(b): Completion performance with 500 loss receivers using 512 Bytes packet size

As time progresses, MFTP/EC had a slightly lower completion rate (in the subsequent file transfer pass) than aFTP and MFTP, since in the latter protocols, the majority of receivers that missed packets in the first pass completed in the second and subsequent pass which explains the surge in completion rate. However, aFTP receivers took a longer time to complete because aFTP's retransmission is based on data carousel where receivers have to wait for a whole file transfer period before receiving a missing packet. In contrast MFTP receivers completed much faster than, since there was no overhead incurred such as encoding and decoding of parity packets in MFTP/EC or data carousel in aFTP.

6.2 Goodput performance

Figure (5) and (6) show comparison of goodput performance over transfer period for aFTP, MFTP and MFTP/EC with increasing number of loss receivers and different packet sizes. In all cases the goodput per interval for MFTP/EC was predominantly higher than aFTP and MFTP at the start of file transfer. This was attributed by the fact that, MFTP/EC sender was able to respond to retransmission request from receivers (experiencing packet loss) during the file transfer pass, thus registering a high goodput compared to both aFTP and MFTP.

As time progresses, MFTP receivers had slightly higher goodput after the first file transfer pass due to retransmission of data packets that were not received in the first pass. As pointed out earlier, both MFTP and aFTP incur delay overheads due to encoding/decoding of parity packets and data carousel respectively causing longer completion times. In general, however, aFTP had a lower goodput than both MFTP and MFTP/EC, since the source always retransmits all packets (including duplicates) in each subsequent pass. This results in low goodput per interval, since receivers have to wait longer (i.e. a whole file transfer period) before receiving a missing packet.

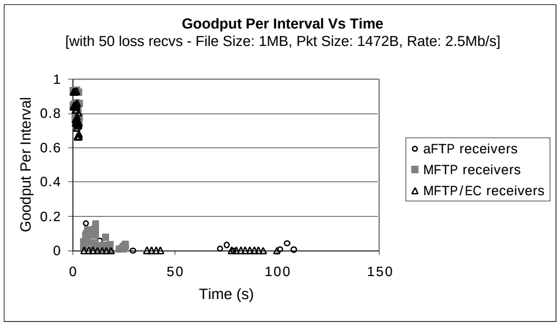

Figure 5(a): Goodput performance with 50 loss receivers using 1472 Bytes packet size

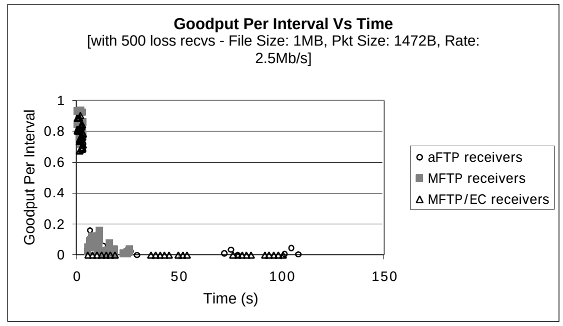

Figure 5(b): Goodput performance with 500 loss receivers using 1472 Bytes packet size

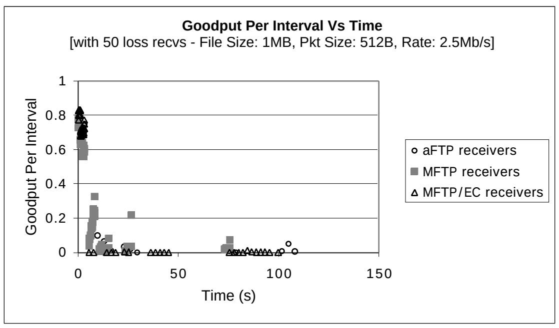

Figure 6(a): Goodput performance with 50 loss receivers using 512 Bytes packet size

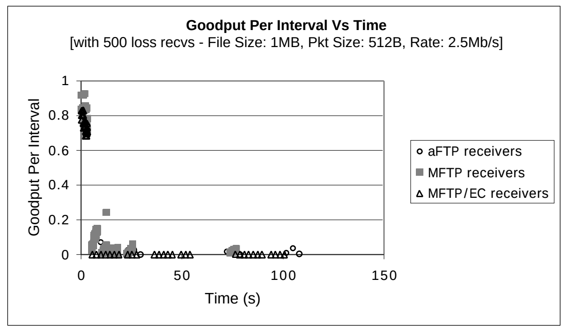

Figure 6(b): Goodput performance with 500 loss receivers using 512 Bytes packet size

6.3 Redundancy Performance

Figure (7) and (8) show the redundancy performance based on different reliability techniques (aFTP, MFTP and MFTP/EC) and increasing file size (using aFTP) respectively. The results (in figure (7)) show that MFTP/EC had the least redundancy overhead compared to both MFTP and aFTP. Using erasure correction, a single repair packet from a MFTP/EC source was able to repair several lost packets requested by different receivers thus resulting in less duplicates per receiver. MFTP receivers experienced slightly higher redundancy overhead, since the source retransmitted and the packets requested by receivers (resulting in high duplication as the receiver number increased). Redundancy overhead in aFTP was far much higher than in MFTP and MFTP/EC, since the source retransmitted the same packets irrespective of whether they were received or not.

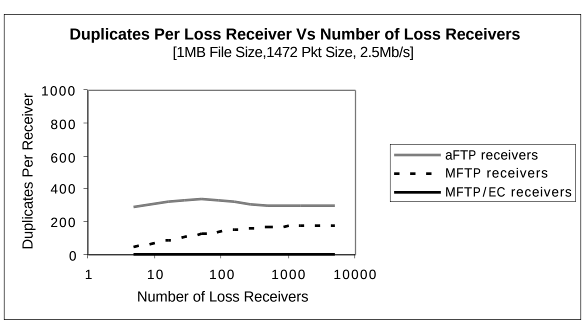

Figure (7): Protocols redundancy performance

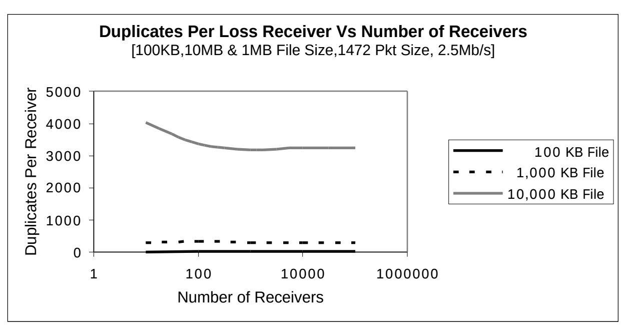

Figure (8): File size redundancy performance

Results in figure (8) show that there is a redundant penalty as file size increases. As the file increases, there is an equally significant increase in redundancy (in terms of duplicates per receiver).

7 Conclusions and further work

In this paper we have evaluated and analysed the performance of RMT protocol techniques (suited for satellite networks) over a persistence link fades (similar to the EHF spectrum fades). In the presentation, we have demonstrated and highlighted various design caveats of this class of RMT protocol techniques (suited for satellite networks) based on MFTP, MFTP/EC and theoretical aFTP. A detailed overview of our fade error model based on PER time-series was also presented. MFTP/EC protocol based on hybrid ARQ/FEC technique performed better than both MFTP and aFTP protocols that were based on check-point ARQ and carousel techniques. Although MFTP/EC performed better than MFTP and aFTP, this technique may have gained better performance if both the source and receivers had adequate information regarding link availability before transmitting data (i.e. when link is unavailable). More so, with large receiver groups, control information from receivers (in hybrid ARQ/FEC) incur unnecessary delay (due to link access delay). Designing an RMT protocol that counteracts these limitations is the next task in our reseach work.

8 Acknowledgements

The authors would like to thank ONERA and ALCATEL SPACE INDUSTRIES for the valuable data on propagation at EHF, and the European Commission IST Programme for funding this research work under the GEOCAST Project.

9 References

[1] C. Diot, W. Dabbous, and J. Crowcroft, “Multipoint Communication: A Survey of Protocols, Functions, and Mechanisms,” *IEEE Journal on Selected Areas in Communications*, vol. 15, pp. 277 - 290, 1997.

[2] K. Obraczka, “Multicast Transport Protocols: A Survey and Taxonomy,” *IEEE Communications Magazine*, vol. 36, pp. 94 - 102, 1998.

[3] IETF Reliable Multicast Transport Working Group, “http://www.ietf.org/html.charters/rmt-charter.html,” January 2001.

[4] M.W. Koyabe and G. Fairhurst, “Reliable Multicast via Satellite: a comparison survey and taxonomy,” *International Journal of Satellite Communication*, vol. 19, pp. 3 - 28, 2001.

[5] B. Whetten, L. Vicisano, R.Kermode, M.Handley, and S.Floyd, “Reliable Multicast Transport Building Blocks for One-to-Many Bulk-Data Transfer,” IETF, RFC3048, June 2001.

[6] G. Maral and M. Bousquet, *Satellite Communication Systems*, 2nd Edition ed. Toulouse: John Wiley & Sons, 1993.

[7] DTT Consulting, “Internet via Satellite'99 Management Summary,” , Hampshire, UK, Technical Report April 1999.

[8] French National Aerospace Research Establishment (ONERA), http://www.onera.fr/english.html January 2001.

[9] A. Paraboni and C. Riva, “A new method for the prediction of fade duration statistics in satellite links above 10 GHz,” *International Journal of Satellite* Communications, vol. 12, pp. 387 - 394, 1994.

[10] M. Liu, J.S. Baras, S.M. Payne, and H. Harrelson, “Modeling and Simulation of Large Hybrid Networks,” International Mobile Satellite Coneference, Ottawa, Canada, 1999.

[11] R. Bostrom, “A Study on the Performance of MFTP over Satellite,” *Electrical Engineering*. Stockholm, Sweden: Telia Research Lab, 1999.

[12] T. R. Henderson, “Networking over Next-Generation Satellite Systems,” *Department of Electrical Engineering and Computer Science*: University of California at Berkeley, 1999, pp. 142.

[13] S. M. Payne, “Reliable Multicast via Satellite,” *Department of Electrical and Computer Engineering*. Maryland: University of Maryland, 1999, pp. 110.

[14] B. S. Bakshi, P. Krishna, D. K. Pradhan, and N. H. Vaidya, “Performance of TCP over Wireless Networks,” International Conference on Distributed Computer Systems (ICDCS), 1997.

[15] W. R. Stevens, *TCP/IP Illustrated, Volume 1*, vol. 1. Massachusetts: Addison-Wesley, 1992.

[16] C. Hanle, “A Comparison of Architecture and Performance between Reliable Multicast Protocols over the MBone,” *Institute of Telematics*: Karlsnuhe, 1997.

[17] C. Hanle and M. Hofmann, “Performance Comparison of Reliable Multicast Protocols using the Network Simulator ns-2,” 23rd Annual Conference on Local Computer Networks (LCN), Boston, USA, 1998.

[18] S. Floyd, “TCP and Explicit Congestion Notification,” *ACM Computer Communication Review*, vol. 24, 1994.

[19] M. Hofmann, M. Rohrmuller, "Impact of Virtual Group Structure on Multicast Performance", 4th International COST 237 Workshop, Lisboa, Portugal, December 15-19, 1997.

[20] C. Hanle, “Feasibility Study of Erasure Correction for Multicast File Distribution using the Network Simulator ns-2,” IEEE/MILCOM, MA, USA, 1998.

[21] J.W. Byers, M. Luby, M. Mitzenmacher, and A. Rege, “A Digital Fountain Approach to Reliable Distribution of Bulk Data,” ACM SIGCOMM, Vancouver, Canada, 1998.

[22] U. L. N. S. (ns), “Network Simulator (*ns*),” http://www.isi.edu/nsnam November 2000.

[23] S. Lin and D. J. Costello, *Error Control Coding: Fundamentals and Applications*. New Jersey: Prentice-Hall, Inc., 1983.

[24] K. Varadhan, D. Estrin, and S. Floyd, “Impact of Network Dynamics on End-to-End Protocols: Case Studies in TCP and Reliable Multicast,” USC/Information Science Institute, California, Technical Report USC-CS-TR 98-672, April 1998.