Packet size and DTN transport service: Evaluation on a DTN Testbed*

Nikolaos Bezirgiannidis
Department of Electrical and Computer Engineering
Democritus University of Thrace
Xanthi, Greece
Email: nbezirgi@ee.duth.gr

Vassilis Tsaoussidis
Department of Electrical and Computer Engineering
Democritus University of Thrace
Xanthi, Greece
Email: vtsaousi@ee.duth.gr

***Abstract*—We present a study of DTN transport layer service, evaluating its performance during a file transmission from a Mars relay satellite towards Earth. The experiments are being held in a DTN Testbed designed and implemented for emulating and evaluating deep-space communications. Bundle Protocol truncates application data into bundles of different size and Licklider Transmission Protocol is used as the convergence layer protocol, encapsulating bundles into blocks and fragmenting blocks into segments. Bundle size, LTP block size and segment size are under consideration to achieve better transmission in terms of delivery time, overhead and memory occupancy at the sending node. Our most interesting conclusion is that memory is released faster when using small bundles and LTP blocks. We also conclude that segments should be close to the maximum boundary of underlying layer's MTU, and that ignoring these dependencies, file delivery time in a channel of steady bit error rate has little dependence from packet sizes.**

***Index Terms*—Delay Tolerant Networking, packet size, Bundle Protocol, LTP**

I. Introduction

Delay/Disruptive tolerant networking, practically, is a nice way to extend networking to areas where typical communication infrastructure does not exist and communication contacts are primarily opportunistic or otherwise characterized by long propagation delays. The two distinctive cases do have something in common: an end-to-end strategy cannot be implemented, at least in a way that was implemented in the traditional Internet. This fact has a rather obvious justification: a responsive sender-receiver interaction cannot be based on occasional communication snapshots and, beyond, a transmission strategy cannot be accurate and efficient under such circumstances.

However, delay-tolerant networking does not really imply that delay is not an important performance metric. Indeed, a delay/disruptive tolerant network should be able to operate even more efficiently when communication contacts exist. That is, in order for DTN to present itself as a complementary communication means, it should invest on maximizing transmission capability during communication contacts - which, in turn, will possibly allow for application completion within the operational cycle and before a new disruptive cycle. For example, imagine a space communication contact with duration of 20 minutes and disruption of 2 hours. It is crucial indeed to complete applications within the 20 minute cycle - otherwise, the application performance will fall, dramatically. This argument has not been highlighted adequately in the related literature.

Within this context, we investigate the impact of packet size on DTN performance. Packet size may influence the error probability, may disrupt the interaction among different network layers, impacts the overhead cost, and the memory utilization. In order to investigate the impact of the aforementioned parameters, we used a real DTN Testbed. The testbed was funded by ESA and its backbone is located in our lab and extends to MIT and Hellenic Aerospace Industry premises though HELLAS SAT. We developed the following scenario: A 10 MByte file is transferred from a Mars relay satellite to a DTN node on Earth through an error-prone channel with propagation delay equal to 10 minutes.

We found that segment size influences overhead greatly, while bundle size has a major impact on memory occupancy and release. Transmission time, on the other hand, is not significantly influenced by either segment or bundle size.

The paper has the following structure: Section 2 briefly describes related work in the area of packet size in wireless networks, and delay/disruption tolerant networks in particular. Section 3 presents the DTN and protocol background of our work and section 4 outlines our experimental methodology, including the DTN Testbed, scenario, parameters and metrics used. In section 5 we present the experimental results and in last section we discuss conclusions and future extensions of our work.

II. Related Work

The scientific area of packet size optimization over error-prone and low rate channels has been thoroughly investigated by researchers. The propagation delay and packet losses that

*This paper was funded by "SPICE-Space Internetworking Center" (Seventh Framework Programme, Capacities, Research Potential Scheme, Coordination and Support Action, FP7-REGPOT-2010-1, Grant Agreement Number 264226)

978-1-4244-7286-4/10/$26.00

appear in every case make data transfer lengthy, energy consuming and occasionally impossible. Different network layers have been under analysis, most of them referring to terrestrial networks. Medium Access Control improvement in packet size attracts the majority of relevant research [7]-[15]. X. Wang et al. focused on the impact of packet size on energy efficiency. [7]. Jun Yin et al. in their simulation [8] analyze the performance of the IEEE 802.11 distributed coordination function. They show that packet error decreases throughput and increases delay, and conclude that there exists an optimal packet length, which maximizes throughput under a given channel condition. Many interesting algorithms for optimizing packet sizes have been presented to adaptively improve the goodput of a system [9], [10]. There has been also some work done that concerns the transport layer packet size. Albuquerque et al. in [11] evaluate performance of TCP Reno connections as a function of TCP packet size. However, all of the mentioned research papers focus on links with small propagation delay, where the sender can receive feedback from the receiver and adjust the packet size accordingly. There are environments, such as the one studied in this paper, where such feedback is not available during one transmission round. A. Harris et al. [12] in one such environment, analyzed packet size adaptation in underwater acoustic networks, with high bit error rates and long propagation delay.

During the recent years, Delay Tolerant Networking has gained much attention especially in space communications. Thus, there has been some work done on the optimization of packet size in space specific protocols. R. Wang et al. in [13] present an experimental study of MAC layer packet size over space links, considering the cross-layer interactions, and using Space Communication Protocol Standards as a protocol suite. In [14], Wang et al. focus on cislunar communications and simulate file transfers using Bundle Protocol and Licklider Transmission Protocol. They measure file transmission time and goodput in relevance to one-way delay, with a maximum of 5 seconds propagation delay. However, in deep-space there are much more lengthy delays than near space and, therefore, major differences occur in network and protocol performance. A deep-space environment study is presented in [15] where the performance of CCSDS File Delivery Protocol (CFDP) is evaluated in weather sensitive Ka-band links. Using network outages, file size and CFDP protocol data unit (PDU) size as input parameters, they measure the storage required to complete 99.99% of file transmissions. They conclude, among others, that unlike PDU size and file size, pass outages do have significant impact on protocol behavior.

III. DTN Background and Protocols

A. Delay Tolerant Networking (DTN)

The existing Internet is based on TCP/IP protocol suite and assures end to end connectivity, provided that the latencies are relatively small, the connectivity is continuous and the bidirectional data rates are high. However, in many networks such as the interplanetary communications this is not the case; propagation delay can reach up to minutes or even hours, the data rates are asymmetric and relatively low and there is an intermittent connectivity, which gives DTN another interpretation, Disruption Tolerant Networking. Such networks have not just emerged. In the near past, each individual network used to have its own dedicated protocol suite and was detached from the global Internet. However, the idea to incorporate all of them to the Internet emerged the *Bundle Protocol (BP)*, a protocol to unite different kinds of networks and treat them as a global one.

B. Bundle Protocol (BP)

Bundle Protocol (specified in [1]) serves as a new protocol layer on top of heterogeneous region-specific lower layers and below the application layer and allows communication between applications in different areas and environments, forming a store-and-forward overlay network. It is an end-to-end experimental architecture, not fully implemented yet, missing at that stage a full implementation of end-to-end data delivery assurance. However, it is used widely in terms of experimental research and will eventually achieve commercial use.

Bundle protocol uses bundle as a protocol data unit which may be of various lengths. Ivancic et al. have used bundle sizes of 160 MB to transfer images from orbit to a ground station [4]. On the other hand in [2], Scott Burleigh suggests use of small bundles, less than 64KB long, to enable partial data delivery at application-appropriate granularity. It is obvious that the scientific community has not yet defined a commonly accepted formula of encapsulating application data into specifically sized bundles. In this paper we try to evaluate a wide scale of bundle sizes and measure the impact of this parameter on deep-space file delivery.

Main capabilities of Bundle Protocol include custody-based retransmission, ability to cope with intermittent connectivity, and ability to take advantage of scheduled, predicted, and opportunistic connectivity, in addition to continuous connectivity. When a node forwarding a bundle ”accepts custody”, it retains a copy of the bundle to re-forward it if necessary. Custody is released when notification is received that some other node (or the destination endpoint) has accepted custody of the same bundle or the bundle is explicitly deleted for some reason, such as lifetime expiration. In terms of deep-space communications, bundle custodian nodes occupy memory to store multiple bundles. It is critical that this memory is not held for longer periods than necessary, because the cost of space resources is excessively high. Thus, one of the main issues we examine is memory occupancy using different sizes of bundles.

Bundles are encapsulated into packets of the underlying transport layer, also known as convergence layer. There are numerous convergence layer protocols being developed by communication researchers, and most of them are still under standardization. One of the main utilities of convergence layer protocols is to ensure hop-by-hop bundle transmission. Bundle delivery reliability in such cases has a single-hop granularity and thus the evaluation of bundle transmission can be ex-

amined in each hop separately. One of the commonly used convergence layers is named *Licklider Transmission Protocol (LTP)* and its main functionality is described in the following section.

C. Licklider Transmission Protocol (LTP)

LTP is one of the mostly known protocols that are used in the convergence layer under Bundle Protocol and its detailed functionality is specified in [5]. Communication in LTP is separated in sessions, also known as *blocks*. Each LTP session has a sender and a receiver node and gathers into blocks bundles ready for transmission to the destination. Then, LTP fragments data into *segments*, which are encapsulated in lower layer frames. The last segment within a set of segments, called a *checkpoint*, requests an answer segment by the receiver, which reports the received packets in a *report segment*. If the data received is the complete LTP block, session ends with a *report-acknowledgement segment*. Otherwise, the report segment details the missing parts of the block, and in turn the sender retransmits the correspondent segments. The process is repeated as many times required until the block is received correctly.

Scott Burleigh, in [2], recommends grouping of multiple bundles destined to a specific node into the same block. However, this is a rather generic rule. For example in a topology of a few nodes with a single sender communicating with a single receiver, the required level of service may not necessarily get any beneficial treatment by this particular LTP stacking policy. Therefore, in this paper, we use a correspondence of one bundle to one LTP block, to avoid overhead.

LTP segment size is also under investigation here: currently there are no instructions in the protocol specification on appropriate segment length, other than Maximum Transmission Unit of the underlying link layer protocol, which inevitably sets an upper bound to segment size.

In figure 1 we illustrate the DTN transport encapsulation process, as described previously.

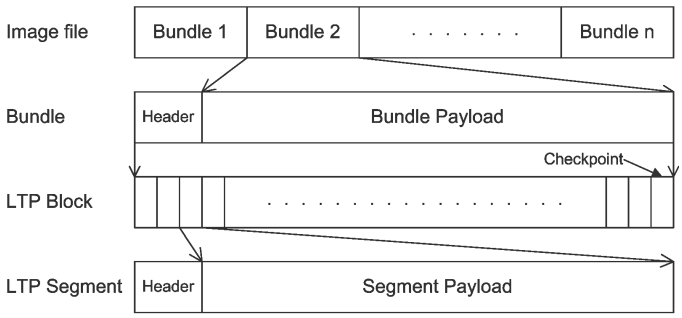

Fig. 1. DTN Transport Service: Packet Encapsulation

Although we use LTP in our experiments, the research and results may be extended to other convergence layer protocols with similar functionality that assure hop-by-hop data delivery.

IV. Experimental Methodology

A. DTN Testbed

Our experimental methodology has also influenced the design of the DTN Testbed that was implemented in our lab ([3]). The Testbed was developed in order to evaluate Delay Tolerant Networking environments such as the deep-space though it may be adjusted to delay/disruption tolerant terrestrial internets.

Our main research purpose was to evaluate real-time network behavior, using deep-space environment conditions. In addition, one of our main requirements being the actual protocol implementations and real protocol headers rather than simulation protocols, the use of a simulation tool was inappropriate. For these purposes, the DTN Testbed provided us with a perfect emulation environment to conduct our real-time experiments.

The Testbed currently consists of 12 geographically spread participating nodes, located in Space Internetworking Laboratory of Democritus University of Thrace, in Hellenic Aerospace Industry in Athens and in Massachusetts Institute of Technology. It includes emulated network links, real and modeled protocol implementations, including Bundle Protocol and LTP, and is enhanced with actual satellite link interfaces offered by HellasSat satellite. Network parameters and evaluation results can be easily controlled by a Central Management System through a Graphic User Interface. The dynamic control of network parameters along with scalability, transparency to upper layer protocols and flexibility to emulate various topologies and incorporate new protocols and mechanisms are its basic elements.

As there is a plethora of available protocol suites to evaluate, we chose the one that fits our evaluation scenario, which is described in the following session. The Testbed's dynamic adjustability allowed us to specifically optimize it to target our scenario and parameters. The software selected to implement the protocol suite we used was Interplanetary Overlay Network [2], and it was customized accordingly. For each experiment conducted, ION was configured according to parameter values in each case. Additionally, the tool that allowed us to incorporate link delays and channel errors was *netem*, included in Linux kernel and therefore integrated in each node of the testbed [6].

In figure 2 we present a preview of the DTN Testbed's protocol stack.

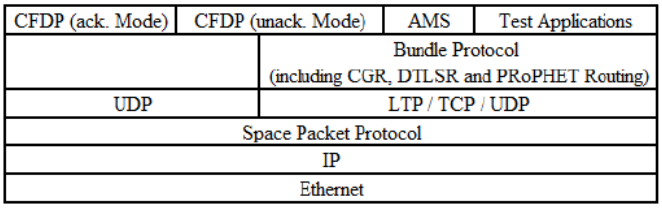

Fig. 2. Testbed protocol stack

B. Scenario and parameters

As mentioned previously, the scenario we examine emulates a file transmission from a Mars relay satellite to a node located on Earth (or an Earth satellite). Mars relay satellites according to JPL [16] have visual contact with Earth for about 2/3 of each orbit, or about 16 hours a day. They can gather data such as images, videos and other information from Rovers exploring Mars surface and then transmit it to the space agency on Earth for analysis. The file being sent in our experiments is 10MB, enough to store an image of the Mars surface. Connectivity is continuous as file transmission time is about an hour and definitely less than the uptime of the link.

One way light time from Mars to Earth is not steady but varies according to the distance between the two planets. With a minimum of approximately 9.4 minutes and a maximum of 17.8 minutes propagation delay ([16]), our choice of 10 minutes is not far from the lower limit. However, the order of magnitude in comparison with a typical propagation delay on Earth, permits us to argue that this quantitative divergence will have no qualitative difference on our results and deductions.

Bandwidth between the two nodes is asymmetric with downlink supporting 100 Kbits per second and uplink typically transferring from 10 to 400 bits per second, due to the reports' small size. Delivery time thus was primarily affected by downlink data rate as files were sent from Mars to Earth, while the opposite flow was used only to deliver small report segments.

Bit Error Rate (BER) in the majority of our experiments was equal to 10^{-6} which is a common and acceptable error rate through deep-space environments, as one can see in [15]. We also evaluated performance with higher error rates (BER=10^{-5}) but also with smaller (BER=10^{-7}) in order to compare the impact of BER. The realistic nature of the experiments obliged us to conduct time-consuming experiments, and therefore we report results from a statistically significant but not a huge number of experiments. Especially in cases with high Bit Error Rates, where file transmission lasted several round-trip-times, we conducted only the necessary number of experiments. In such cases, that number was calculated using statistical measures such as typical deviation among the first 20 experiments. *File Delivery Time*, however, with BER=10^{-5} needed a large number of experiments in order to achieve statistically acceptable results. This fact, along with the lengthy file transmissions, forced us to exclude measuring File Delivery Time using high error rates.

Bit Error Rate was in turn translated into Packet Error Rate (PER) in each experiment. PER values varied from 0.05% (segment size = 600 Bytes, BER=10^{-7}) to 10.9% (segment size = 1400 Bytes, BER=10^{-5}). The PER used was random with a uniform distribution, thus making it a typical channel error, which may not correspond to the characteristics of errors caused by external factors (weather conditions, etc.).

DTN bundle sizes used were 20, 50, 100 and 500 KBytes. LTP block size has been fixed accordingly to correspond to one single bundle, equaling the size of bundle. LTP Segment sizes used were 600, 800, 1000, 1200 and 1400 Bytes.

Emulation parameter values are listed in Table I.

TABLE I
SCENARIO PARAMETERS

Parameters	Values
File size	10 MB
Propagation delay	10min
Downlink rate	100Kbps
Uplink rate	Asymmetry (10-400 bps)
Bit Error Rates	10^{-5}, 10^{-6}, 10^{-7}
Bundle/LTP block size (KBytes)	20, 50, 100, 500
LTP Segment size (Bytes)	600, 800, 1000, 1200, 1400

C. Metrics

In order to evaluate file delivery through deep-space, we try to focus our attention on vital space communication issues. One of them is resource conservation. Energy consumption, along with telemetry and telecommand incessant row, makes channel resources invaluable and increases the importance of reducing transmission effort. Another important issue is quick file delivery, as observation data may need to be delivered, stored and further inspected by space agencies as soon as possible. Finally, we evaluate buffer storage exploitation. In terrestrial communications buffer storage is a basic factor when designing protocols. In deep-space networks, however, where hardware and components cost is extravagant, memory and buffer usage becomes most important. Examining the above deep-space communication issues, we mainly focus on three important metrics, which are listed below.

1) At first, we examine *Wasted Transmission Effort (WTE)*, which includes protocol overhead and retransmitted packets, as a percentage of pure application data (actual file size).

$$WTE = 100\% * \frac{\text{Total Bytes Sent} - \text{Application Data}}{\text{Application Data}} \quad (1)$$

2) The second metric we measure is *File Delivery Time (FDT)*, which represents the time needed to complete the 10MB file transmission to the last segment. During our statistical analysis, a percentage of files were delivered successfully through two round-trip-times. During these file deliveries, lost packets were transferred correctly through the first retransmission round. In many cases though, retransmitted packets were lost during their retransmission, leading to a second retransmission round and consequently to an major increase in FDT. This fact, along with the 10-minute propagation delay, led to a quite large deviation of FDT. Therefore, we measured the percentile of file transmissions, as the time needed to complete specific percentages.
3) Finally we present a third metric, *Memory Occupancy*, which measures the memory space detained at the sender

and shows its release through time with the delivery of bundles.

V. Experimental Results

In figures 3 and 4 we capture *Wasted Transmission Effort*, which consists of the overhead along with retransmitted packets. Changing both segment size and bundle/block size (see figure 3), we may argue that segment size is the parameter that affects WTE mostly. This can be easily explained, as the overhead produced by segments with 600 Bytes length is more than twice the overhead of 1400 Bytes segments. Bundle size and LTP block size, in contrast to segment size, have no significant impact on the overhead produced. It is obvious from figure 4 that, even with varying error rate, segment size remains the most important factor for *WTE*. This is despite the fact that retransmissions increase greatly.

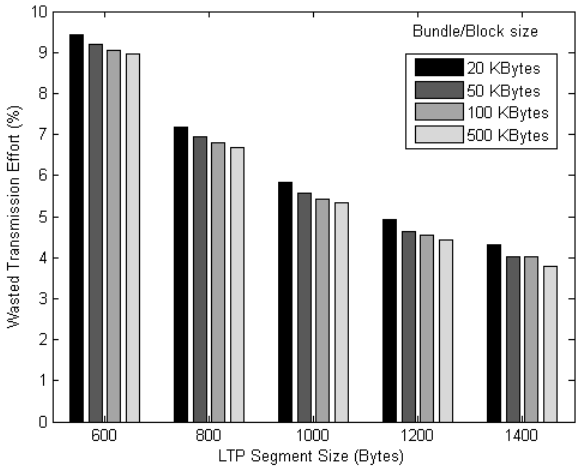

Fig. 3. WTE vs LTP segment size and bundle/LTP block size. BER = 10^{-6}

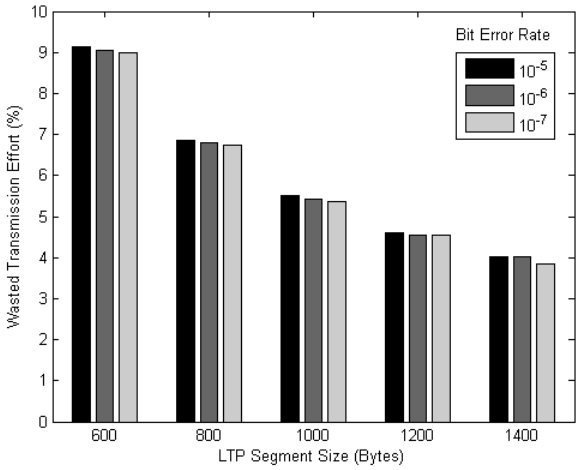

Fig. 4. WTE vs LTP segment size and BER. Bundle/block size = 100KBytes

File Delivery Time is studied next. During this set of experiments, we kept BER fixed and equal to 10^{-6} and varied segment size and bundle/block size. Our analysis included a statistically acceptable number of file transmissions, during which FDT percentile was measured. It appears from figure 5 that there is no significant difference when using fixed segment size and varying bundle/block size. The nearly horizontal line between 10% and 30-40% of percentile captures all experiments that were completed with only one retransmission round, while random loss of any retransmitted packet led to a significant time increase in data delivery completion through the rest of the experiments.

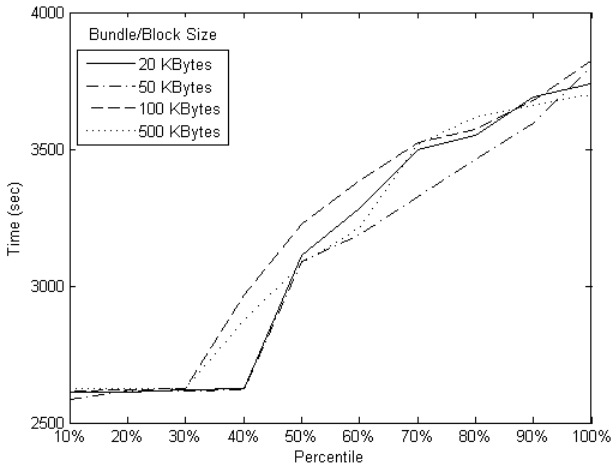

Fig. 5. File Delivery Time vs file transmission percentile. Segment size = 1400 Bytes, BER = 10^{-6}

We also note that if we isolate file deliveries that needed only one retransmission round (that is roughly 30-40% of total file transmissions), we could argue that bundle/block size has a minor impact on *File Delivery Time* (see figure 6).

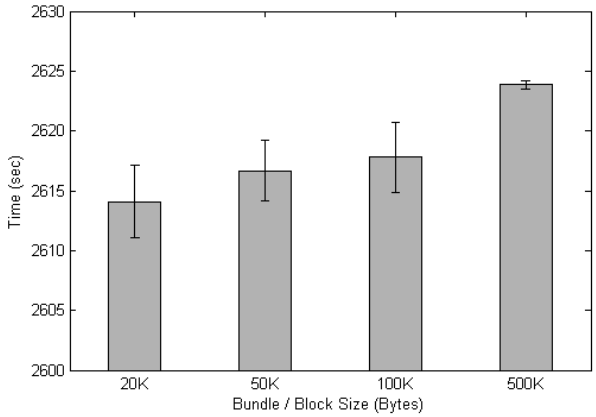

Fig. 6. Average File Delivery Time with 99% confidence interval vs bundle/block size for files completed with one retransmission. Segment size = 1400 Bytes, BER = 10^{-6}

This can be justified by the fact that the last retransmitted segment, and the block it belongs to, determine the file completion time. For example, if the last reported lost segment

contains bytes from 9,700,000 to 9,701,000, the acknowledgement reporting its loss will be sent by the receiver upon receiving the *checkpoint* (last segment) of the specific LTP block. This means that for a 20KB block size, when the 9,720,000th Byte is received, the destination node will send a segment reporting any lost bytes. For a 100KB block size, when the 9,800,000th Byte reaches destination, a similar report segment will be sent. As a result, report segment arrives later at the sender node and, consequently, lost packet is retransmitted with a small time delay, in comparison with the first case (20KB block size). In figure 6 we also illustrate the 99% confidence intervals that derived from our statistical analysis.

In the next pair of diagrams (figures 7 and 8), we examine *File Delivery Time*, keeping block size fixed and varying segment size.

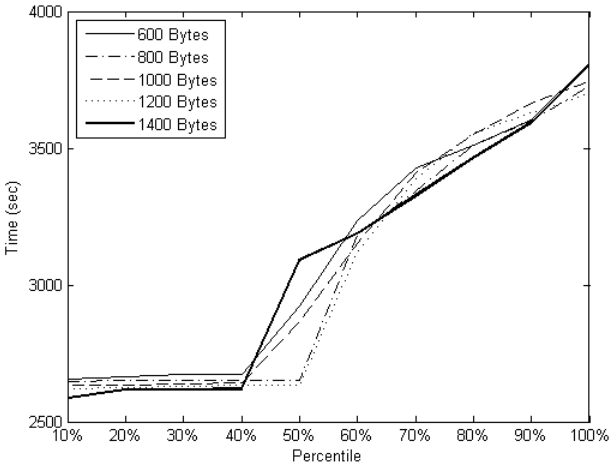

Fig. 7. File Delivery Time vs file transmission percentile. Bundle/block size = 50 KBytes, BER = 10^{-6}

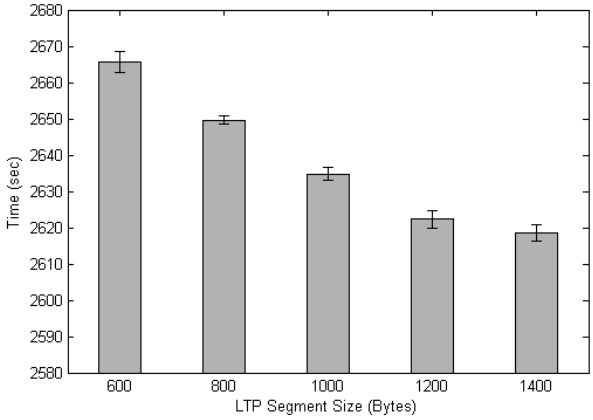

Fig. 8. Average File Delivery Time with 99% confidence interval vs segment size for files completed with one retransmission. Bundle/block size = 50 KBytes, BER = 10^{-6}

In figure 7 we measure the percentile of completed file transmissions. We can argue that there is no significant change when using different segment values. This is in contrast to the vast difference in previous diagrams presenting *Wasted Transmission Effort*. One might expected that greatly increasing overhead, transmission time would also increase in a more straight-forward way. Instead, there is only a slight effect in the statistical analysis of all file deliveries. However, overhead impact on data transmission time becomes more evident when we isolate the experiments that were completed with a single retransmission round. Average FDT of these experiments along with 99% confidence intervals are presented in figure 8. It is obvious that, in that case, mean FDT decreases as segment size increases. However, the difference in average time values is minor.

In figure 9 we present average file delivery times and 99% confidence intervals with BER = 10^{-7}, bundle and block size fixed and equal to 100 KBytes, and segment size varying from 600 to 1400 Bytes. During these file deliveries there was no second retransmission round and, therefore, small deviation and confidence intervals allowed us to measure average time values. It is obvious, therefore, that overhead is the main factor that affects transmission time, increasing it with the decrease of segment size.

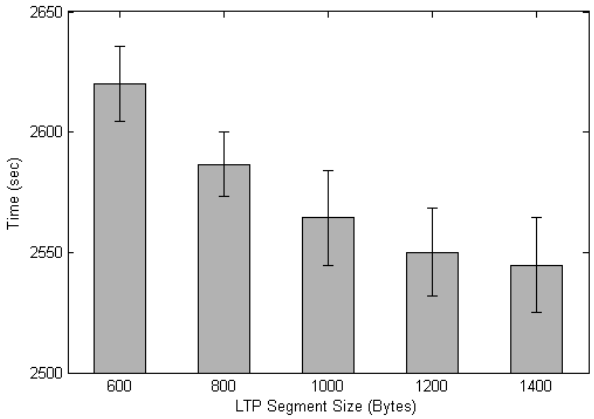

Fig. 9. Average File Delivery Time with 99% confidence interval vs segment size. Bundle/block size = 100 KBytes, BER = 10^{-7}

Finally, we measure our most interesting metric, *Memory Occupancy* at the sender. Our first observation is that keeping bundle/block size fixed and changing segment size has no significant impact on memory release rate. This is quite obvious in figures 10 (block size = 50KBytes) and 11 (block size = 20KBytes). During the first time period (roughly 0-1250 seconds), data are being transmitted from sender to receiver and report segments are being transmitted vice versa, though no report segment has arrived yet to the sender node. During the next time period, report segments arrive continuously at the sender. In case any of these segments report successful bundle delivery (with no lost segments), the specific bundles are released from memory, resulting in *Memory Occupancy* decrease. The horizontal lines between 2000-2500 seconds capture the time period between the first and the second transmission rounds. During this time slot, no report segment

arrives at sender and, therefore, no bundles are discarded from memory.

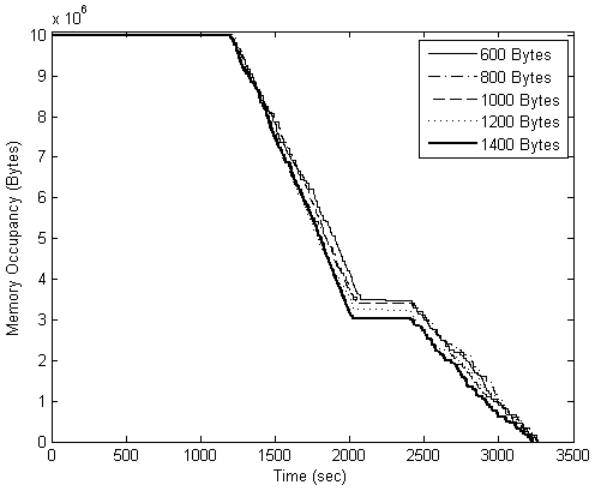

Fig. 10. Memory Occupancy through time vs LTP segment size. Bundle/block size = 50 KBytes, BER = 10^{-6}

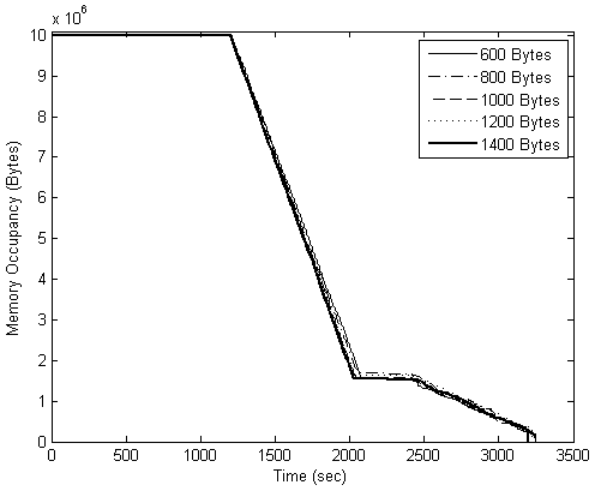

Fig. 11. Memory Occupancy through time vs LTP segment size. Bundle/block size = 20 KBytes, BER = 10^{-6}

Figures 10 and 11 show results from a single experiment conducted for each segment size value, and are not a statistical analysis result. This is mainly because of randomness: In each experiment, even conducted with the same segment size, different segments are lost and different bundles are released from memory, preventing us from statistically analyze a plethora of experiments. However, when comparing results from different experiments with the same segment size, the differences that appear are insignificant. So, we may argue that presenting a single experiment for each segment size is adequate to analyze and compare results between different segment sizes.

During the next set of experiments, we kept LTP segment size fixed and compared *Memory Occupancy* using bundle/block sizes from 20 to 500 KBytes. Figures 12 (segment size = 600 Bytes) and 13 (segment size = 1400 Bytes) picture the comparative results. As mentioned in the previous paragraph, a single experiment from each bundle/block length is depicted. It is obvious from figures 12 and 13 that there is a substantial difference in *Memory Occupancy* between experiments with different bundle/block sizes.

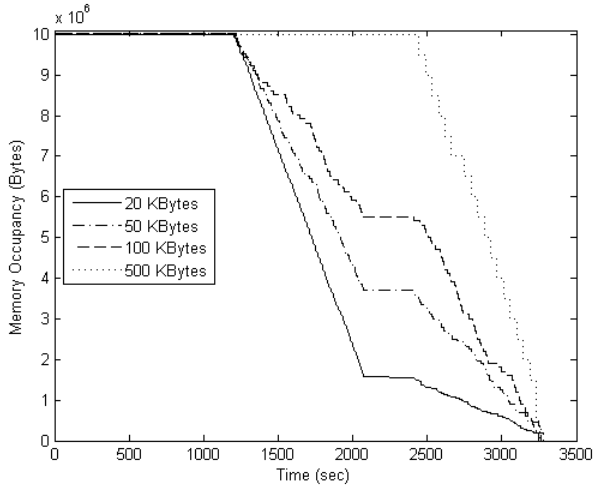

Fig. 12. Memory Occupancy vs bundle/block size. Segment size = 600 Bytes, BER = 10^{-6}

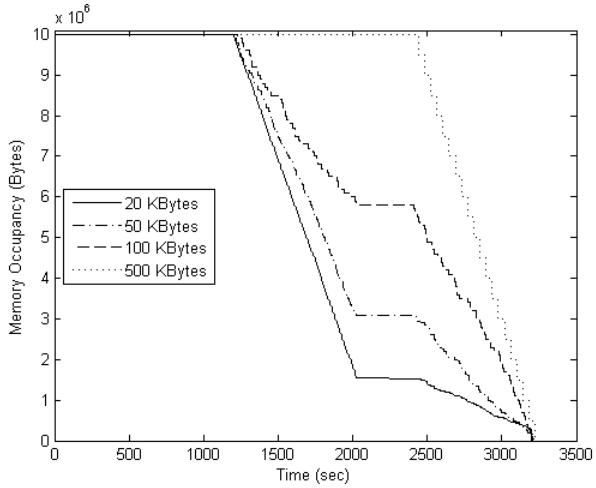

Fig. 13. Memory Occupancy vs bundle/block size. Segment size = 1400 Bytes, BER = 10^{-6}

This divergence can be justified by the granularity of stored and acknowledged data. In order to achieve reliable file delivery, sender node releases each bundle only on receipt of complete bundle delivery report. Therefore, a single segment loss results in memory conservation of the whole bundle that contains the specific segment. For example, when bundle size used is 500 KBytes and one 1400Byte-segment is lost, receiver node may not release that bundle until reception of the lost segment. As a result, when bundle size is small, there is

a quicker memory release in comparison with larger bundle sizes.

This is a very interesting observation and may be of great value to space agencies as well as space protocol designers. Fast memory release can be very useful, especially in space nodes, where memory components are expensive and where critical applications operate. In some cases, memory usage may be vital even for a space node's life.

VI. Conclusion - Future Work

In this paper, we evaluated the performance of DTN transport layer service in a DTN Testbed that emulates deep-space environments. File transmissions from a Mars relay satellite to Earth are studied using BP as a store-and-forward protocol and LTP as a convergence layer protocol, varying sizes of bundle, LTP block and segment. Despite the real-time emulation restrictions, we reached a number of conclusions, which are presented in the following paragraphs:

- Increase of LTP segment size has one major consequence: Overhead. Channel occupancy increases by roughly 5% of the application data when segment size is reduced from 1400 to 600 Bytes. As a result, when segment size is small, transmission effort and energy consumption increase at the sender node. Therefore, we may argue that optimal LTP segment size should be as close to the underlying protocol MTU as possible. This could be extended to other convergence layer protocols as well. The observation that small protocol frame size leads to increased overhead is independent of the convergence protocol used.
- *File Delivery Time* is not significantly affected by bundle/block size and segment size. Randomness in packet losses, along with long round-trip-time, boosts transmission time deviation. In a percentage of experiments, file is delivered completely in a single retransmission round (roughly 30-40%), while in the rest it requires more rounds. In cases where file delivery does not require a second retransmission round, such as in channels with BER=10^{-7} or in a percentage of file transmissions with BER=10^{-6}, we observed a clear pattern: Increasing bundle and block size leads to a minor increase in file delivery time, while increasing segment size decreases file delivery time. The time difference in both cases, however, is insignificant.
- From the evaluation of *Memory Occupancy* at the sender, we discovered that bundle size and LTP block size are significant parameters indeed. When bundle size and the correspondent block size is large, a lost segment forces the sender node to keep the entire bundle and block in memory. When multiple small bundles are used instead, sender keeps only the bundle that contains the lost segment, while the rest are released from memory. This is very useful in deep-space telecommunications mainly because of memory cost. Space component memory is released faster when small bundle and block sizes are used, and it can be used by other critical applications.

In this paper we cover only typical random errors. We intend to extend this work further and evaluate packet size impact on protocol behavior, when errors appear in bursts or, in any case, the error pattern corresponds to external factors such as weather conditions. An interesting future direction also is to investigate packet size in space communications, using more nodes, different topologies and alternate routes. Our initial conclusions about memory release, in particular, could be examined further in a more realistic scenario with a more complex topology.

References

[1] Keith Scott and Scott Burleigh, Bundle Protocol Specification, IETF RFC 5050, experimental, November 2007

[2] Interplanetary Overlay Network (ION) Design and Operation, V1.11, Jet Propulsion Laboratory, California Institute of Technology, December 2009, URL: https://ion.ocp.ohiou.edu

[3] E. Koutsogiannis, S. Diamantopoulos, G. Papastergiou, I. Komnios, A. Aggelis and N. Peccia, "Experiences from architecting a DTN Testbed" *Journal of Internet Engineering, v.3, n.1*, December 2009

[4] W. Ivancic, W. M. Eddy, D. Stewart, L. Wood, J. Northam, C. Jackson, "Experience with delay-tolerant networking from orbit", *4th Advanced Satellite Mobile Systems, 2008* pp. 173-178, August 2008

[5] M. Ramadas, S. Burleigh and S. Farrell Licklider Transmission Protocol - Specification, September 2008

[6] S. Hemminger, Network emulation with netem, Linux Conf Au, April 2005

[7] Xiaodong Wang, Jun Yin and Dharma P. Agrawal, "Effects of Contention Window and Packet Size on the Energy Efficiency of Wireless Local Area Network", *IEEE Wireless Communications and Networking Conference 2005, pp. 94-99* New Orleans, March 2005

[8] Jun Yin, Xiaodong Wang and Dharma P. Agrawal, "Optimal Packet Size in Error-prone Channel for IEEE 802.11 Distributed Coordination Function", *IEEE Wireless Communications and Networking Conference 2004* pp. 1654-1659, March 2004

[9] Dalei Wu, Song Ci, Hamid Sharif and Yang Yang, "Packet Size Optimization for Goodput Enhancement of Multi-Rate Wireless Networks" *IEEE Wireless Communications and Networking Conference 2007*, pp. 3575-3580, 11-15 March 2007

[10] Eytan Modiano, "An adaptive algorithm for optimizing the packet size used in wireless ARQ protocols", *Wireless Networks v.5, n.4*, pp. 279-286, July 1999

[11] Marcelo Albuquerque, Jae H. Kim, Sumit Roy, "Effect of packet size on TCP-Reno performance over lossy, congested links", *IEEE Military Communications Conference 2001*, pp. 705-710, October 2001

[12] Albert F. Harris III, Davide G. B. Meneghetti, Michele Zorzi, "Maximizing Channel Utilization for Underwater Acoustic Links", *Oceans 2007*, pp. 1-6, June 2007

[13] Ruhai Wang, Stephen Horan, Rama Chandrasekaran, "An experimental investigation of cross-layer optimal packet size in space Internet", *International Journal of Satellite Communications and Networking, v. 24 n. 6*, pp. 561-577, November 2006

[14] Ruhai Wang, Tiaotiao Wang, Xuan Wu, "Bundle protocol (BP) over Licklider transmission protocol (LTP) for cislunar communications", *Proceedings of the 28th IEEE conference on Global telecommunications*, pp. 3492-3496, 2009

[15] Jay L. Gao, John S. SeGu "Performance evaluation of the CCSDS file delivery protocol - latency and storage requirement", *IEEE Aerospace Conference 2005*, pp.1300-1312, 5-12 March 2005

[16] Jet Propulsion Laboratory, California Institute of Technology, Mars Exploration Rover Mission, http://marsrovers.jpl.nasa.gov/mission/communications.html