Performance Analysis of the AeroTP Transport Protocol for Highly-Dynamic Airborne Telemetry Networks

Kamakshi Sirisha Pathapati, Truc Anh N. Nguyen, Justin P. Rohrer
Department of Electrical Engineering & Computer Science
The University of Kansas
Lawrence, KS 66045
{kamipks,annguyen,rohrej,jpgs}@ittc.ku.edu

Faculty Advisor:
James P.G. Sterbenz

ABSTRACT

Due to the challenging network conditions posed by a highly-dynamic airborne telemetry environment, it is essential for the transport protocol to provide automated mechanisms that dynamically adapt to changing end-to-end performance on any path. The AeroTP multi-mode transport protocol provides service tailored to the requirements of the telemetry mission control and data packets, achieving better performance compared to the traditional TCP and UDP. We use ns-3 to simulate the AeroTP protocol's reliable and quasi-reliable modes and demonstrate the performance tradeoffs between the modes, as well as comparing their performance with TCP and UDP.

I. INTRODUCTION AND MOTIVATION

The highly dynamic airborne telemetry environment poses many unique challenging network conditions. One of the challenges is the constrained bandwidth caused by limited RF spectrum and the large quantity of data being sent over the network. The second challenge is the limited transmission range due to power and weight constraints of test articles (TAs). In addition, the high velocity of TAs poses the third challenge, the problem of mobility that results in highly dynamic topology changes. The fourth challenge is intermittent connectivity that arises due to the second and the third challenges. Unfortunately, the current TCP/IP-based Internet architecture is not appropriate for this environment [1] and there are several issues to be solved at the network and transport layers [2]. Given these constraints, in order to make the network resilient, we need to have cross-layer enabled protocols at the transport, network, and MAC layers that are uniquely designed to address the challenges posed by the aeronautical telemetry network. These protocols also have to be TCP/IP compatible with those devices located on the airborne nodes and with the control applications. With that in mind, in the context of the Integrated Network Enhanced Telemetry (iNET) program for Major Range and Test Facility Bases (MRTFB) across United States [3], we developed the Airborne Network Telemetry Protocol (ANTP) suite [4, 5]. The suite includes AeroTP – a TCP-friendly transport protocol that supports multiple reliability modes, AeroNP – an IP-compatible network protocol, and AeroRP – a routing protocol that takes advantage of location information to mitigate the short contact durations between high-velocity nodes. The different modes of AeroTP include reliable, nearly-reliable, quasi-reliable, unreliable connection, and unreliable datagram modes [6].

The AeroTP transport protocol in our ANTP suite was introduced in [6] and further developed in [5]. This paper extends the simulation and evaluation of its different modes and demonstrates the performance tradeoffs of each mode as well as comparing their performance with TCP and UDP. The rest of this paper is organized as follows: Section II presents some background and related work. Section III gives a detailed description of AeroTP simulation model. This is followed by our simulation results and an analysis on the performance of different modes in comparison to each other as well as to TCP in Section IV. Finally, Section V concludes the paper with directions for future work.

II. BACKGROUND AND RELATED WORK

Ideally, reliable data transfer transmits data end-to-end with low delay and with no errors or losses. But, transmission in a network is often prone to delay, limited bandwidth, and multiple errors along the path towards the destination. Bit errors are the most common in wireless channels because of the channel's vulnerability to noise and interference. Packet losses are caused because of congestion, switching between multiple paths within the network, and packet-drops resulting from the occurrence of bit errors in the packet. To avoid the errors caused by congestion, congestion control and avoidance algorithms are used. When implicit congestion notification is used they reduce the window size each time congestion is detected. Packet drops at the receiver may be caused because of corrupted packets. Error recovery schemes are often a solution to correct the errors in the received data packet. The Automatic Repeat Request (ARQ) mechanism uses ACKs and retransmissions to ensure all the lost packets are successfully delivered to the destination. The Forward Error Correction (FEC) is an alternative error-control mechanism that sends redundant information in each packet, allowing it to correct bit errors at the receiver without requesting a retransmission from the source.

A. Drawbacks of Traditional Protocols

Although TCP and UDP are the most commonly used transport protocols they fail to perform efficiently in a challenged wireless environment. In wireless networks packet losses are inevitable; link outages, lossy channel characteristics, unstable connectivity, delay, and congestion are a few examples of challenges that cause packet loss. A wireless channel is often subjected to interference and channel fading, resulting in packet loss and packet corruption. TCP assumes every packet loss is caused by congestion in the network and invokes its congestion control algorithm. This decreases the congestion window by a fraction (usually half), thus causing inefficient use of bandwidth. Schemes such as split-TCP connections and local retransmissions were developed to circumvent the problem caused by TCP's assumption of congestion being the only cause for packet loss [7].

TCP uses ACKs to provide reliable data transmission and retransmissions. The source retransmits a TCP segment to the destination when a timeout occurs while waiting for an ACK. A connection setup is performed through a three-way handshake between the source and the destination pair of nodes. This takes one round-trip time (RTT) before data may be sent, which causes significant performance degradation in a telemetry network because of short contact duration between nodes. By using a slow start algorithm, TCP takes many RTTs to ramp up the sending rate before it can fully utilize the available bandwidth. This results in a significant amount of wasted capacity in an environment that often has episodic connectivity.

Because of dynamic topologies, link outages are common. The congestion control algorithm is invoked during short link outages, causing an increase in the number of retransmissions. The connection is terminated in case of longer link outages. This causes difficulty in restoring links and finding alternate paths to the destination [4]. TCP also does not provide any QoS differentiation for prioritizing the type of data being transmitted.

SCPS-TP (Space Communications Protocol Standards – Transport Protocol) [8] is an extension to TCP, used particularly for satellite communications, developed to address problems posed by asymmetric links. SCPS-TP addresses some similar problems to those of telemetry networks although it is not fully suitable for highly dynamic networks. This is in part because it relies on channel condition information which is either pre-configured or discovered over time from the network [9].

Although UDP is a simpler protocol than TCP, it does not offer any guarantee for data delivery, so it is unreliable. Unlike TCP, UDP does not have a connection set-up mechanism and does not provide congestion or flow control or data retransmissions. UDP also does not provide differentiated levels of precedence or QoS for the classes of information required in the telemetry environment.

B. AeroTP Overview

AeroTP is a connection-oriented TCP-friendly protocol that performs efficient end-to-end data transfer between the edges of the telemetry network, while providing QoS for various classes of data used and smooth interoperability with TCP and UDP at the gateways. It performs transport layer functions such as connection-setup and management, transmission control, and error control. The protocol operates with five different transfer modes providing services that are connection-oriented and connectionless, and range from completely reliable, to partly reliable, and unreliable:

- ***Reliable connection*** mode uses end-to-end acknowledgement semantics from source to destination to guarantee correct data delivery.
- ***Near-reliable connection*** mode is highly reliable, but does not guarantee correct delivery. The gateway uses split ARQ (custody transfer) and immediately returns TCP ACKs to the source.
- ***Quasi-reliable connection*** mode completely eliminates ACKs and ARQ. It uses open-loop error recovery mechanisms such as FEC and erasure coding to achieve statistical reliability.
- ***Unreliable connection*** mode uses no error correction mechanism at the transport layer. It relies exclusively on the FEC of the link layer to preserve data correctness.
- ***Unreliable datagram*** mode is a stateless mode which provides no assurance of reliable delivery.

III. AEROTP SIMULATION MODEL

In this section we describe the ns-3 simulation model of AeroTP. The purpose of this model is to allow us to compare its performance with other transport protocols, as well as refine its performance.

C. AeroTP Segment Structure

An AeroTP segment (shown in Figure 1) is structured for a bandwidth-constrained network so it does not encapsulate the entire TCP/UDP and IP headers, but is capable of converting to the TCP/UDP format at the gateways. To satisfy the end-to-end semantics it keeps certain fields in common with the TCP/UDP headers such as the source-destination port numbers, TCP flags, and the timestamp. The sequence number uniquely identifies AeroTP segments for reordering them at the receiving edge and for error-control purposes. The HEC (header error check) field performs a strong CRC on the header to detect bit errors caused by wireless channel, thus making sure the packet is correctly transmitted to the destination. In case the payload gets corrupted, AeroTP performs FEC on the payload. The payload CRC is used in the absence of a link-layer frame CRC and enables the measurement of bit-error-rate for error correction depending on the transfer mode used.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Source Port Number        |     Destination Port Number   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Sequence Number                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           Timestamp                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|resv |  Mode   |ECN|   Flags   |         Payload Length        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
/ Optional fields for FEC, Erasure Coding, ... /  TP HEC CRC-16 /
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
\                                                               \
/                            Payload                            /
\                                                               \
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
/              Payload FEC Parity Trailer (Optional)            /
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Payload CRC-32 (Optional)                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 1: AeroTP data segment structure

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Source Port Number        |     Destination Port Number   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Sequence (ACK) Number 0                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           Timestamp                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|resv |  Mode   |ECN|   Flags   |         Payload Length        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
/ Optional fields for FEC, Erasure Coding, ... /  TP HEC CRC-16 /
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          ACK Number 1                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          ACK Number ...                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          ACK Number N                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
/              Payload FEC Parity Trailer (Optional)            /
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Payload CRC-32 (Optional)                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2: AeroTP MACK segment structure

D. AeroTP Operation

As a connection-oriented protocol, it is essential to define and maintain consistent states at the sender and the receiver to establish a connection for data transfer. The states either remain the same or evolve to another depending on the events and actions that happen within the protocol during a communication session. Figure 3 shows the AeroTP reliable mode packet flow-diagram, in which S is the source, D is the desitnation, and TmNS represents the telemetry network and figure 4 shows the state transition diagram.

Similar to TCP, the AeroTP source-destination pair uses control messages (ASYN, ASYNACK, AFIN,

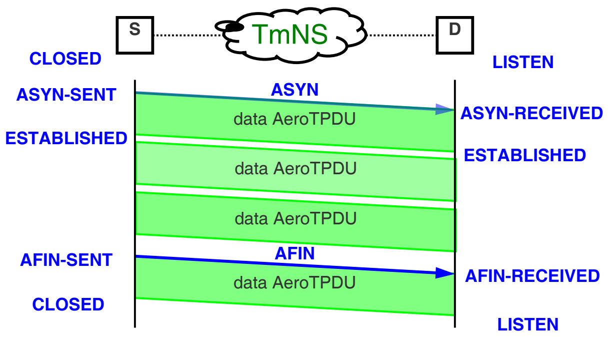

Figure 3: AeroTP connection management

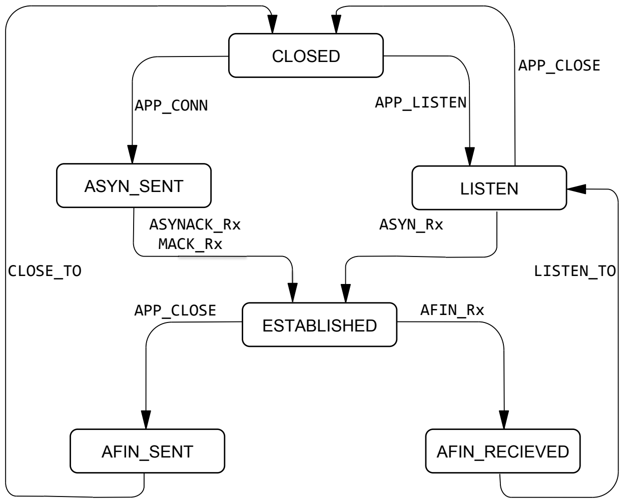

Figure 4: AeroTP state transition diagram

and AFINACK) for opening and closing a connection. The difference is an opportunistic connection establishment, in which data and control overlap, is chosen over the TCP's three-way handshake, thus saving a round-trip-time that is otherwise wasted. Initially the sender and the receiver are in the CLOSED state. A connection is initiated by the sender through an APP_CONNECT message from the application. The sender then sets the ASYN bit in the AeroTP header and transmits it with or without data depending on the data being available in the send buffer and moves to the AFIN_SENT state. The receiver receives an APP_LISTEN request from the application and moves to the LISTEN state, and upon receiving the ASYN message it moves to the ESTABLISHED state, and acknowledges the ASYN by setting the ASYN bit and the MACK bit simultaneously. The sender moves to the ESTABLISHED state as long as the ASYNACK or a simple MACK is received, so that the connection does not have to terminated in case the ASYNACK gets lost. While in the ESTABLISHED state, the sender and the receiver exchange AeroTPDU and ACKs. After the sender is finished with sending all the data, an AFIN message (with AFIN bit set in the header) is sent to the receiver and the sender moves to the AFIN_SENT state. Upon receiving the AFIN message the receiver moves to the AFIN_RCVD state and transmits an AFINACK. To make sure that the receiver has acknowledged all the data including retransmissions, the receiver begins a timer in AFIN_RCVD state which goes to a LISTEN state after a time long enough so that all the retransmissions have likely been received from the sender. The sender also maintains a close timer, which expires after a certain time interval that is long enough to likely receive acknowledgements for all data packets. This way the sender is guarantees delivery of all the data packets with high probability.

In the reliable mode, error-control is achieved by implementing a selective-repeat ARQ mechanism. To reduce the overhead incurred, AeroTP aggregates multiple ACKs at the receiver into a single packet before transmitting them to the sender, as TCP does using the SACK option [10]. The number of ACKs to be aggregated is a tunable parameter, and the optimal value depends on the probability of error or loss in the channel, as well as the rate at which packets are sent. AeroTP also uses a timer to guarantee that ACKs will not be delayed long enough to trigger unnecessary retransmissions.

Table 1: State Transition Definitions

State	Description
CLOSED	State in which no connection exists and no data is transferred
LISTEN	State in which a receiver is ready to listen to any incoming data
ASYN_SENT	ASYN message sent by the host initiating connection
ESTABLISHED	A steady state in which data transfer takes places
AFIN_SENT	AFIN message sent to indicate no new data being sent
AFIN_RECEIVED	AFIN message received and AFINACK is sent as an acknowledgement
APP_CONN	Request issued by the application to initiate connection by sending ASYN
APP_LISTEN	Request issued to the receiving host to move to the LISTEN state
APP_CLOSE	Request to intiate closing a connection by sending AFIN
ASYN_RX	ASYN received, indicating a connection has been requested
ASYNACK_RX	ASYNACK received, indicating connection request has been granted
MACK_RX	Single or multiple packet ACK received
AFIN_RX	AFIN received, indicating end of any new data, initiating connection close
CLOSE_TO	A timeout allowing outstanding retransmissions before going to CLOSED state
LISTEN_TO	A timeout to go to LISTEN state so that the receiver can receive all data packets

IV. SIMULATION RESULTS

We have implemented the fully-reliable (ARQ) and quasi-reliable (FEC) described above as ns-3 simulation models from section III . This section presents the simulation results from running these models. We compare the performance of AeroTP in the reliable connection and quasi-reliable modes with TCP and UDP protocols using the ns-3 open-source simulator [11]. The selective-repeat ARQ algorithm is used to provide reliable edge-to-edge connection between nodes for the reliable mode, and FEC (as discussed above) is used for the quasi-reliable mode of the AeroTP protocol. The network in this simulation setup consists of two nodes communicating via a link that is prone to losses. One node is configured as a traffic generator, and the other as a traffic sink. The traffic generator sends data at a constant data rate of 4.416 Mb/s (3000 packets/s with an MTU of 1500 B). The path consists of a 10 Gb/s link representing the LAN on the TA, a 5 Mb/s link with a latency of 10 s representing the mobile airborne network, and a second 10 Gb/s link representing the LAN at the ground station. Bit errors are introduced in the middle link with a fixed probability for each run, and the performance for each probability of bit-errors is shown in the plots described in the next section. A total of 1 MB of data is transmitted during one single simulation between the two nodes. The link is made unreliable by introducing losses using an error model varying bit-error probabilities ranging from 0 to 0.0001 for each of the protocols. Each simulation case is run 20 times and the results averaged to obtain the data needed for comparison.

E. *Fully-reliable mode performance*

In *fully-reliable* mode, AeroTP uses ARQ as its reliability mechanism. This trades off additional latency (in the case of lost or corrupted packets) and overhead of the reverse channel, for reliability. The advantage to this mechanism is that given enough time, every lost packet can be retransmitted. In

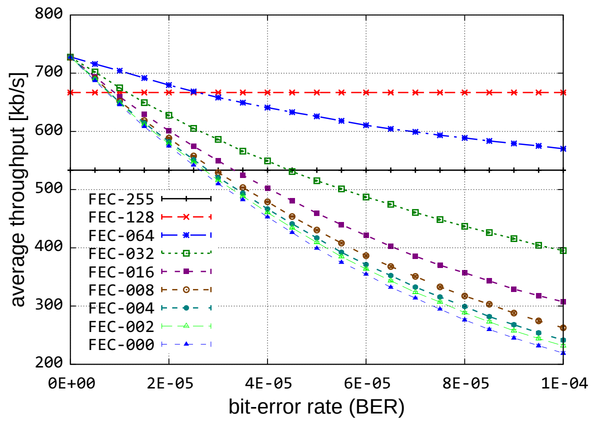

Figure 5: Average goodput

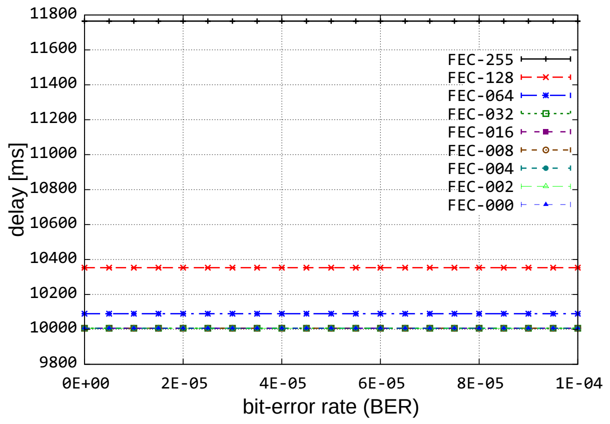

Figure 6: Average delay

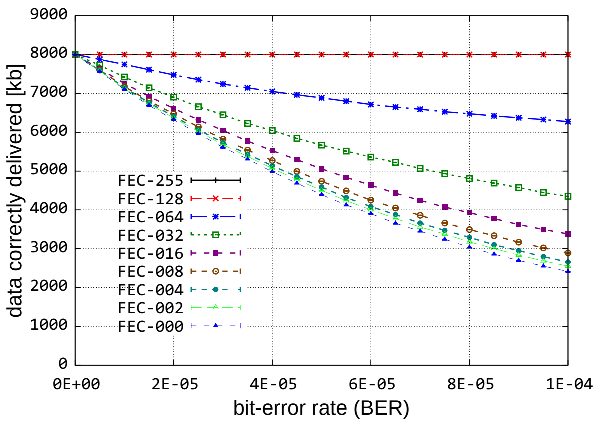

Figure 7: Cumulative goodput

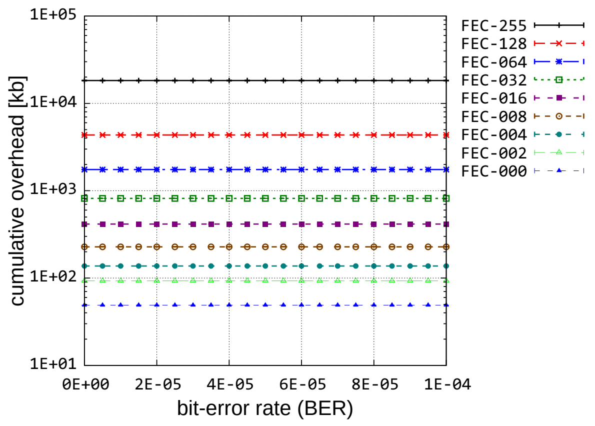

Figure 8: Cumulative overhead

our model we are able to adjust the amount of bandwidth required by adjusting the number of ACKs aggregated into a single packet before it is transmitted. We found this to have a negligible effect on performance, and so have omitted the set of plots showing adjustments to this parameter to save space. The overall performance of the fully-reliable mode can be seen in our last set of plots.

F. Quasi-Reliable Mode Characterization

In *quasi-reliable* mode, AeroTP uses FEC as its reliability mechanism. This trades overhead on each packet for reliability. The advantage to this mechanism is low delay, because no retransmissions are required to correct errors. Our first set of plots compares varying FEC strengths, from zero FEC 32 bit words per packet, to 256 FEC words. In all cases 1500-byte packets are used, thus as the number of FEC words in each packet is increased, the number of bytes of data in each packet decreases, meaning that more packets are required to transfer the same amount of data. Figure 5 shows that the throughput

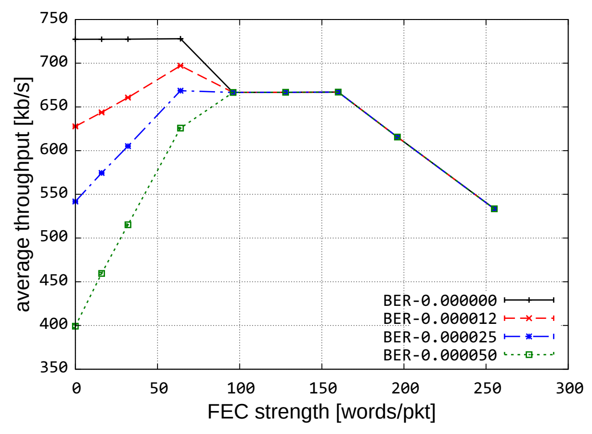

Figure 9: Average goodput

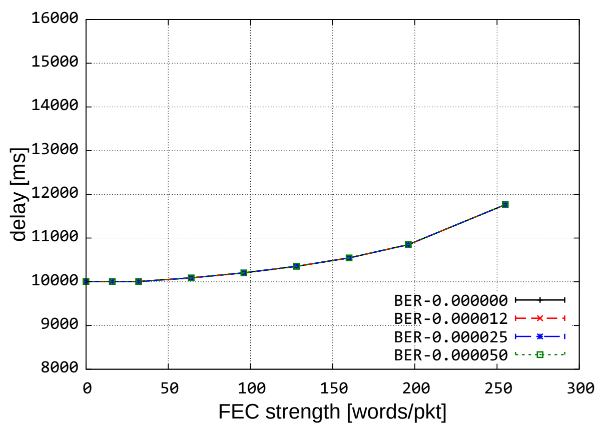

Figure 10: Average delay

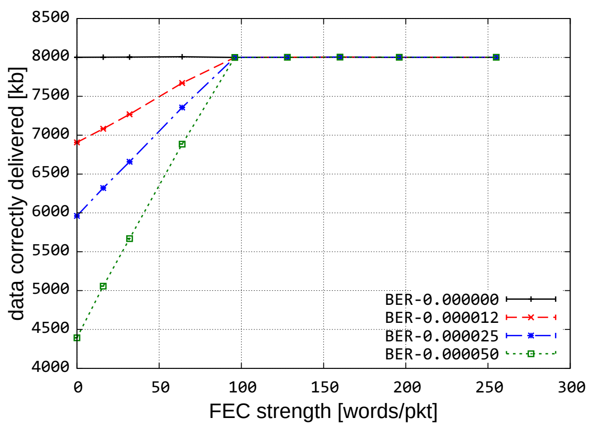

Figure 11: Cumulative goodput

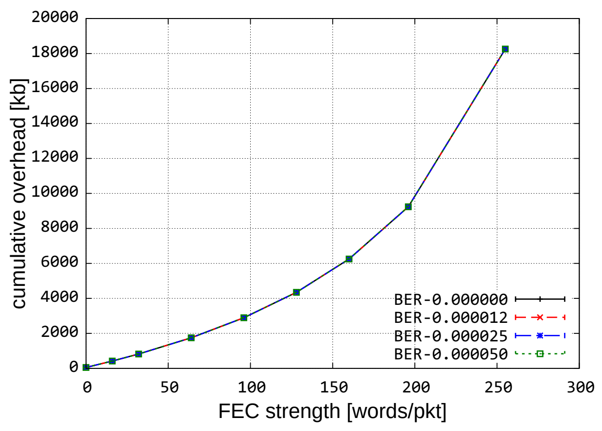

Figure 12: Cumulative overhead

decreases due to uncorrectable packets as the error-rate increases, however this effect can be mitigated by increasing the FEC strength. For very high FEC strengths (128 and 256), there was virtually no decrease in performance across the range of error-rates tested, however the performance is decreased at low error-rates due to the high level of overhead. Due to the fact that retransmissions are not involved, the latency of data transmission is not affected by packet errors as shown in Figure 6. However, as very high levels of FEC result in link saturation this translates into added latency due to queuing delay. Similar to the throughput plot, Figure 7 shows the total amount of data transmitted. Depending on the FEC strength, this quantity decreases as the error-rate increases, except for very high FEC strengths (128 and 256) all errors are able to be corrected, at the rates tested. Lastly in this set of plots, we show the overhead imposed on the network by using the FEC mechanism at various strengths (Figure 8). This quantity is significant (note the log y-axis scale), however it is not affected by the error rate.

The next set of plots continue to characterize the *quasi-reliable* mode. Figure 9 shows that for error rates greater that zero, higher FEC strengths result in higher throughput up to a point. For the 128-word and 256-word FEC strengths, the amount of FEC bytes being sent begins to saturate the link, resulting

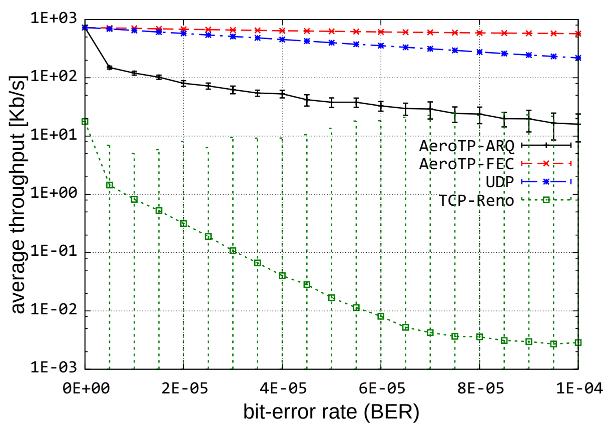

Figure 13: Average goodput

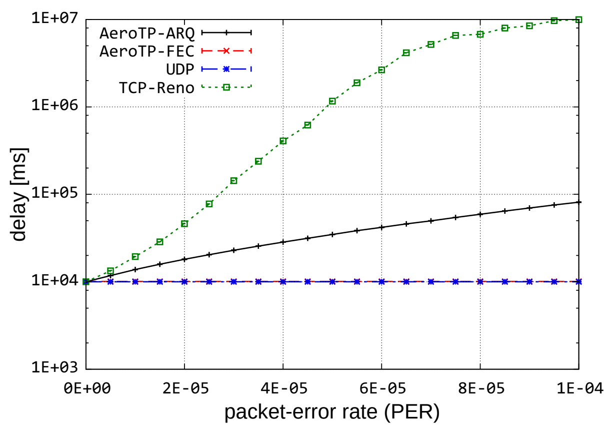

Figure 14: Average delay

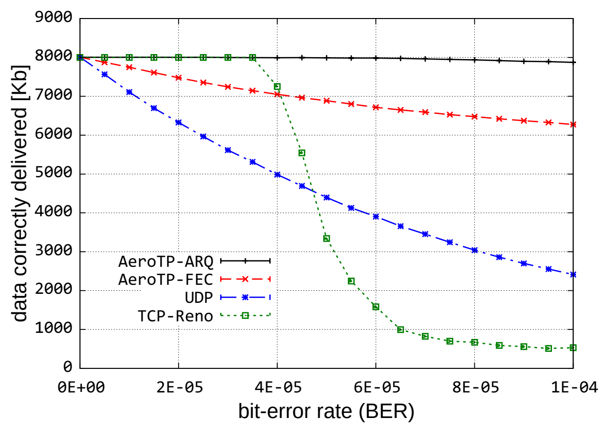

Figure 15: Cumulative goodput

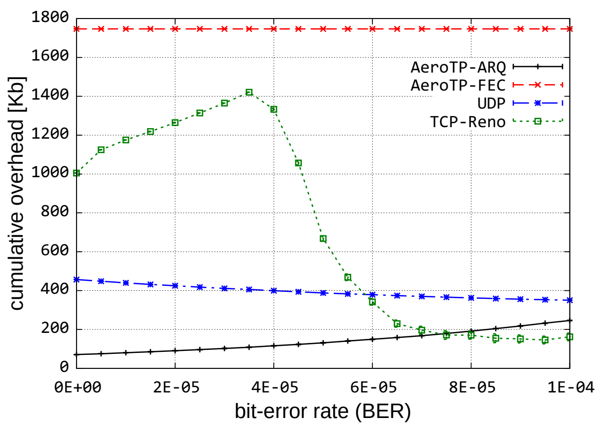

Figure 16: Cumulative overhead

in reduced throughput of data. Figure 10 shows that for all error rates, higher FEC strengths increase delay slightly as the link becomes saturated. Figure 11 shows that an FEC strength of 96 words/packet or greater is able to correct all errors at the error rates tested. Figure 12 shows the increase in overhead resulting from increased FEC strength. The increase is linear with respect to the amount of data being sent, but since we have chose to quantify FEC strength with respect to the number of packets transmitted it appears exponential, due to the fact that an increase in FEC strength results in an increase in the number of packets sent to transmit a give amount of data using maximum-size packets. Future models may change this quantification in favor of one relative to the amount of data being transmitted.

G. Performance Comparison over Lossy Links

Figure 13 shows that AeroTP reliable-mode is able to achieve significantly better performance than TCP, which backs off substantially as the BER (bit-error rate) increases. TCP also becomes highly unpredictable in its performance, as shown by the error bars. At the same time TCP's end-to-end delay increases by 3 orders-of-magnitude doubles with a BER of 1×10^{-4}, while AeroTP increases less than 1 order-of-magnitude as shown in Figure 14. Over the course of the simulation, both TCP and AeroTP are able to deliver the full 1 MB of data transmitted for low error rates $<$0.000035, but above that TCP performance drops rapidly while AeroTP is still able to deliver nearly all the data at the highest error rates as shown in Figure 15. In the same plot we see that UDP looses a percentage of the data due to corruption as the BER increases, and that the AeroTP quasi-reliable mode losses a much smaller percentage. Lastly in Figure 16 we see that the performance improvement of the AeroTP reliable-mode is achieved with much lower overhead than TCP, while quasi-reliable mode does incur significant overhead, but does not cause any increased delay as the BER increases.

V. CONCLUSIONS AND FUTURE WORK

In this paper we presented the ns-3 simulation model of AeroTP, and results showing it's significant performance advantages in a lossy environment. We also show the tradeoffs enabled by having both a fully-reliable and a quasi-reliable mode. We are in the process of implementing this protocol to run in a linux environment along with the rest of the ANTP suite [12]. This will enable testing on mobile platforms as well as in real-world traffic conditions. In the future we will also be simulating and implementing gateways to allow translation between the traditional Internet protocol stand and the ANTP suite.

ACKNOWLEDGEMENTS

The authors would like to thank the Test Resource Management Center (TRMC) Test and Evaluation/Science and Technology (T&E/S&T) Program for their support. This work was funded in part by the T&E/S&T Program through the Army PEO STRI Contracting Office, contract number W900KK-09-C-0019 for AeroNP and AeroTP: Aeronautical Network and Transport Protocols for iNET (ANTP). The Executing Agent and Program Manager work out of the AFFTC. This work was also funded in part by the International Foundation for Telemetering (IFT). We would like to thank Kip Temple and the membership of the iNET working group for discussions that led to this work.

REFERENCES

[1] "iNET Needs Discernment Report, version 1.0." Central Test and Evaluation Investment Program (CTEIP), May 2004.

[2] "iNET Technology Shortfalls Report, version 1.0." Central Test and Evaluation Investment Program (CTEIP), July 2004.

[3] "iNET System Architecuture, version 2007.1." Central Test and Evaluation Investment Program (CTEIP), July 2007.

[4] J. P. Rohrer, A. Jabbar, E. Perrins, and J. P. G. Sterbenz, "Cross-layer architectural framework for highly-mobile multihop airborne telemetry networks," in *Proceedings of the IEEE Military Communications Conference (MILCOM)*, (San Diego, CA, USA), pp. 1–9, November 2008.

[5] J. P. Rohrer, A. Jabbar, E. K. Çetinkaya, E. Perrins, and J. P. Sterbenz, "Highly-dynamic cross-layered aeronautical network architecture," *IEEE Transactions on Aerospace and Electronic Systems (TAES)*, vol. 47, October 2011.

[6] J. P. Rohrer, E. Perrins, and J. P. G. Sterbenz, "End-to-end disruption-tolerant transport protocol issues and design for airborne telemetry networks," in *Proceedings of the International Telemetering Conference*, (San Diego, CA), October 27–30 2008.

[7] H. Balakrishnan, V. N. Padmanabhan, S. Seshan, and R. H. Katz, "A comparison of mechanisms for improving TCP performance over wireless links," *IEEE/ACM Trans. Netw.*, vol. 5, no. 6, pp. 756–769, 1997.

[8] R. C. Durst, G. J. Miller, and E. J. Travis, "TCP extensions for space communications," in *MobiCom '96: Proceedings of the 2nd annual international conference on Mobile computing and networking*, (New York, NY, USA), pp. 15–26, ACM Press, November 1996.

[9] J. P. Rohrer and J. P. G. Sterbenz, "Performance and disruption tolerance of transport protocols for airborne telemetry networks," in *International Telemetering Conference (ITC) 2009*, (Las Vegas, NV), October 2009.

[10] M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow, "TCP Selective Acknowledgment Options." RFC 2018 (Proposed Standard), Oct. 1996.

[11] "The ns-3 network simulator." `http://www.nsnam.org`, July 2009.

[12] M. AL-Enazi, S. A. Gogi, D. Zhang, E. K. Çetinkaya, J. P. Rohrer, and J. P. G. Sterbenz, "ANTP protocol suite software implementation architecture in python," in *International Telemetering Conference (ITC)*, (Las Vegas, NV), October 2011. (to appear).