Space Link Extension Protocol Emulation for High-Throughput, High-Latency Network Connections

Nicole Tchorowski*
Lafayette College, Easton, PA 18042, USA
Robert Murawski†
NASA Glenn Research Center, Cleveland, OH 44105, USA

Abstract

New space missions require higher data rates and new protocols to meet these requirements. These high data rate space communication links push the limitations of not only the space communication links, but of the ground communication networks and protocols which forward user data to remote ground stations (GS) for transmission. The Consultative Committee for Space Data Systems, (CCSDS) Space Link Extension (SLE) standard protocol is one protocol that has been proposed for use by the NASA Space Network (SN) Ground Segment Sustainment (SGSS) program. New protocol implementations must be carefully tested to ensure that they provide the required functionality, especially because of the remote nature of spacecraft. The SLE protocol standard has been tested in the NASA Glenn Research Center's SCENIC Emulation Lab in order to observe its operation under realistic network delay conditions. More specifically, the delay between then NASA Integrated Services Network (NISN) and spacecraft has been emulated. The round trip time (RTT) delay for the continental NISN network has been shown to be up to 120ms; as such the SLE protocol was tested with network delays ranging from 0ms to 200ms. Both a base network condition and an SLE connection were tested with these RTT delays, and the reaction of both network tests to the delay conditions were recorded. Throughput for both of these links was set at 1.2Gbps. The results will show that, in the presence of realistic network delay, the SLE link throughput is significantly reduced while the base network throughput however remained at the 1.2Gbps specification. The decrease in SLE throughput has been attributed to the implementation's use of blocking calls. The decrease in throughput is not acceptable for high data rate links, as the link requires constant data a flow in order for spacecraft and ground radios to stay synchronized, unless significant data is queued a the ground station. In cases where queuing the data is not an option, such as during real time transmissions, the SLE implementation cannot support high data rate communication.

I. Introduction

Future, high data rate space communication networks require not only high data rate wireless communication links, but reliable, high throughput ground networks that can support the forwarding of user data between remote users and the ground station terminals. The Consultative Committee for Space Data Systems (CCSDS) Space Link Extension (SLE) protocol has been proposed for the high-throughput, reliable ground network traffic for the NASA Space Network (SN) Ground Segment Sustainment (SGSS) program. The SLE protocol is needed in order to transmit non-standard layer-2 protocols throughout the NISN, which generally do not contain IP information needed to route them throughout the network. To support the next-generation high throughput space communication links, a new CCSDS SLE implementation has been

*Student Member AIAA
†AIAA Member, NASA Contract Employee working for Vantage Partners, LLC

developed which must support the SGSS data rate requirements of 300 *Mbps* data transfer and future 1.2 *Gbps* Ka-Band satellite communication links.

Before use in operational systems, the proposed SLE protocol must be tested in realistic scenarios which include non-ideal link characteristics, such as network delay, jitter, or packet loss. Even with adverse network conditions, the SLE connection must maintain high throughput as the satellite and ground station modems require a constant stream of data in order to maintain synchronization. If the connection is not maintained, the synchronization between modems in the spacecraft and on the ground could be lost, requiring resynchronization and potentially leading to packet loss. This issue can be alleviated given a sufficiently large data buffer which could maintain the constant data stream; however, for real time traffic, buffering will not help.

In this paper, the SLE standard has been tested in the NASA Glenn Research Center's SCENIC emulation lab. These tests show its operation under realistic network delay conditions between the Mission Operation Center (MOC) and ground stations over the NASA Integrated Services Network (NISN). The NASA Communication Service Office (CSO) states that the maximum expected round trip time (RTT) delay for a continental United States (CONUS) connection is 120 *ms*. Both a baseline network connection and an SLE connection were tested with RTT delay in the network link ranging from 0 *ms* to 200 *ms*. This range was chosen to accommodate longer delays as some international connection delays may extend beyond what is expected for CONUS traffic. Tests carried out focused on how the current protocol implementation reacted in a link during these delay conditions. Results show that with realistic delay in the link, the SLE implementation would have a decrease in throughput up to 99%, while the base network tests suffered no loss in throughput.

Another SLE protocol implementation has also been examined in another work for its ability to operate in high data rate missions. ESA's study[1] on the protocol studied the use of TCP Reno, TCP Cubic, or UDT as options for the underlying protocol. When packet loss was present in their network tests, the throughput in the links dropped significantly, where TCP Reno dropped from 826.26 *Mbps* to .641 *Mbps* and TCP Cubic dropped from 925.212 *Mbps* to 3.065 *Mbps* for a .1 percent packet loss in the network. The drop in data rates due to packet loss is similarly seen in our own tests. This study focused on a different implementation of the SLE protocol than the implementation that this paper studies, and does not include testing of the SLE throughput for higher data rates.

II. SLE Protocol

The SLE recommended standards as defined by the CCSDS was developed to forward user-generated layer-2 frames between remote users the ground stations which transmit through the space communication link. For space communications, the user-generated layer-2 frames are generally CCSDS standard protocols which cannot be processed by commercial off the shelf (COTS) network switches. Furthermore, space-communication protocol stack may not utilize an IP-based network layer required to route packets through a IP-based layer-3 network. The user-generated layer-2 frames typically do not contain IP information and are CCSDS standards. Commercial switches cannot process CCSDS protocols, while routers would require IP information for routing, therefore a tunneling protocol is required for these packets to be routed through a layer-3 ground communication network, such as the NASA Integrated Services Network (NISN). SLE provides this service, through encapsulating the frames into a standard TCP/IP packet stream, so that intermediate routers can forward the traffic to their destination.

A typical protocol stack of where the SLE protocol operates within the network between the Mission Operation Center (MOC), NASA Integrated Services Network (NISN) and the Ground Station (GS) is shown in Figure 1.

It is important to test that the protocol can handle adverse network conditions in an emulation environment before it is used in real scenarios. An emulation environment allows for software emulation models to interact with hardware in real time, which gives a realistic estimation on how the system will behave in real scenarios. One of the important features to check if whether the SLE protocol can maintain the suggested levels of throughput in realistic network conditions. Satellite modems used for space communication must maintain a constant stream of data to maintain synchronization. Therefore, it is imperative that the data source, either at the spacecraft or ground station, maintain the required level of throughput. The radios on-board the spacecraft and ground must always remain in synchronization, which requires that data is constantly sent between the two. If the SLE protocol is unable to send packets out at the required speed, there may be a break in the transmission of the frames to between spacecraft and ground. This will cause a

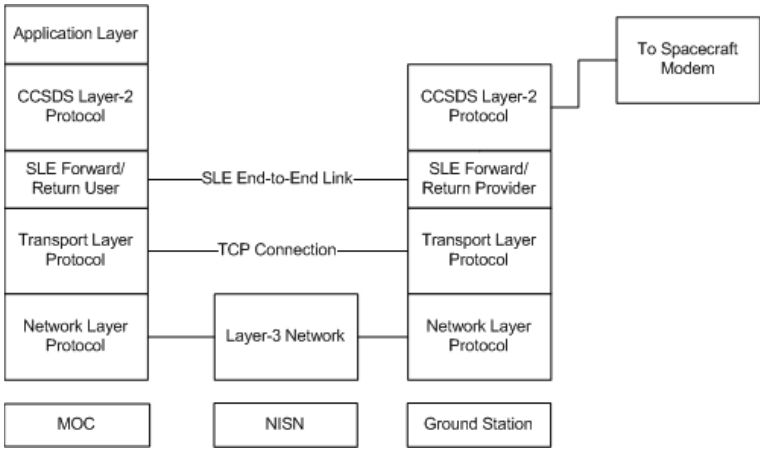

Figure 1. Network Stack Diagram with SLE Tunnel

loss of synchronization, and packets may be lost while the radios resynchronize. A data buffer at the ground station may alleviate this issue, but it would not be possible for missions which require a high data rate or real time data transfer.

The SLE protocol involves two links, the forward link, which has data sent from the ground network to the spacecraft[2] , and the return link, which sends data from the spacecraft to ground[3] . For the forward link, the Forward Communications Link Transmission Unit (FCLTU) protocol was used, while the return link used the Return All Frames (RAF) protocol. During the testing of the SLE protocol for this paper, both SLE Forward and Return links were tested. The SLE protocol allows for both blocking or non-blocking calls to the underlying transport layer protocol. The API Interface for the Forward CLTU Service [4] published by the CCSDS describes these options in detail. It states that:

"A service user is not required to wait for a CLTU-TRANSFER-DATA return before transmitting the next CLTU-TRANSFER-DATA invocation."

The CLTU-TRANSFER-DATA command will send a Protocol Data Unit (PDU) to the transport layer for transmission, and wait for a return from the CLTU-TRANSFER-DATA command from the transport layer that indicates if the packet is successfully received, or any error conditions that occurred. In addition, the standard also lists three options for the implementation of the CLTU-TRANSFER-DATA returns:

1. *an implementation can always pass invocations for which a check has failed to the application*
2. *an implementation can prevent queuing of invocations by withholding an invocation until the previous invocation has been confirmed by the application. In that case, it can always generate the appropriate return when needed; or*
3. *an implementation can decide to pass invocations to the application on a case by case basis*

Based on our observations, we believe the current implementation uses the second option, which can be seen in Figure 2. This implements a blocking scheme, where a CLTU-TRANSFER-DATA invocation can only be called once there is a confirmation that the receiver has received the current frame. In a blocking scheme, the SLE protocol will send the CLTU-TRANSFER-DATA command to send a PDU to the transport layer, and then block all further processing until the CLTU-TRANSFER-DATA command returns. As you will see in Section IV, this results in significantly reduced throughput in the link, and renders the SLE implementation unusable for high data rate missions.

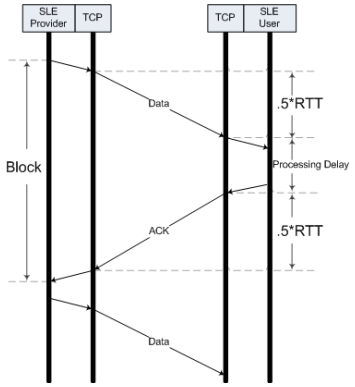

Figure 2. Diagram showing the flow of data between SLE User and SLE Provider

III. SCENIC Emulation Lab

A. Network Configuration

The configuration for testing of both the base network connection and the SLE protocol is shown in Figure 3. Two IBM x3550 M4 servers running Red Hat Enterprise Linux (RHEL) 6.4 run the proposed SGSS SLE Implementation, which was developed by Ingenicomm. The network connection between the two servers was established through a Brocade MLXe4 switch, and was carried over a 10*Gbps* fiber link. The delay that was added into the network was added through the Linux netem command. The configuration of the servers can be seen in Figure 3.

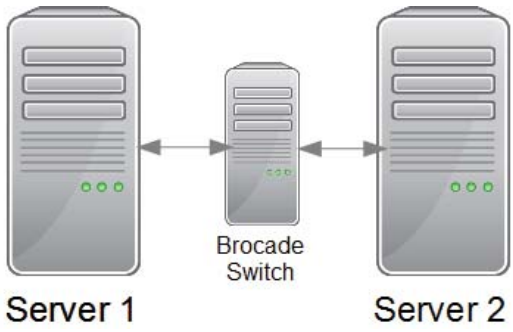

Figure 3. The network connection with the switch as the connection between the two servers

B. Baseline Network Test

In order to determine the basic performance characteristics of the SCENIC Emulation Lab setup and the Configurable Ground Systems (CGS) software, a baseline network test was performed. The Ingenicomm software was configured to open a TCP socket between the two servers and send packets between them, while tracking performance metrics. It is important to note that both this network connection and the SLE network connection both use the same underlying TCP protocol settings, so performance issues resulting from the TCP protocol will be present during both test scenarios. The CGS system provides delay measurement blocks, which generate data at a specified bitrate and record the delay from the sender to the receiver. These blocks were configured to generate data at 1.2 *Gbps*. Bash scripts were written in order to periodically pull performance metrics such as delay and jitter from the CGS system every second, for a total of five minutes.

Tests were initially carried out with no delay added into the link, and then delay was added through the Linux kernel up to a maximum round trip time (RTT) of 200ms. The NASA Communication Service Office (CSO) Services Document (CSD) states that the maximum RTT for Mission Critical or Real-Time Critical IP traffic within the Continental United States is 120ms. The selected RTT of 200ms provides coverage for this maximum delay. The average data rate for each RTT delay was recorded, and can be seen in Figure 4.

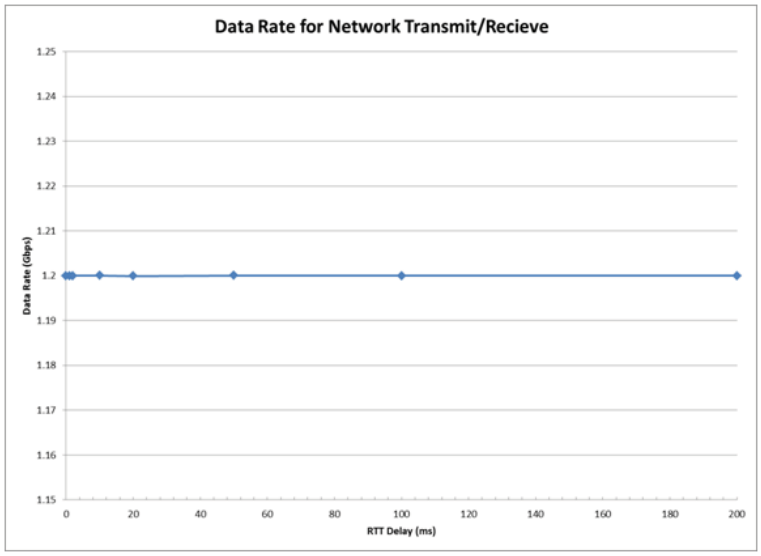

Figure 4. Base network throughput vs. delay from 0ms delay to 200ms delay.

As you can see, the baseline network configuration was able to provide the specified 1.2 *Gbps* data rate throughout a delay range of $0ms$-$200ms$.

IV. Space Link Extension (SLE) Network Performance Tests

Two performance tests were carried out to test the SLE network performance. First, the SLE protocol was tested in a similar manner as the base network connection test, with data generated at a rate of $1.2Gbps$, and increasing amounts of delay added into the network link. This test was run in order to examine the SLE protocol's reaction and ability to handle delay in the network link. The data rate of 1.2Gbps is shown here, which is higher than the SGSS requirement of 300Mbps. The 1.2Gbps data rate used is representative of future high-data rate Ka-Band satellite communication links, which is desirable for future SCaN services. Tests at lower data rates were also performed and produced similar results. The second test consisted of a link with no delay, with varying data rates from $.5Gbps$ to $1.6Gbps$ and was conducted in order to see what the throughput capabilities of the SLE implementation are.

A. SLE Network Delay Tests

The SLE performance test required configuration of the CGS software to create an SLE User on one server and an SLE Provider on the other. The SLE data service was also set up to receive pseudo-random data from a data generator and send this data at a throughput of 1.2 *Gbps*, with 7000 byte frames. The data generator was setup to provide 1.2 *Gbps* of pseudo-random data to the SLE protocol in 7000 byte application layer packets. The data generated for each frame packet was composed of pseudo-random data and a timestamp used to estimated link delay. For each test the SLE buffers were setup to allow the SLE protocol to utilize the maximum SLE PDU length of $64kB$.

Tests were run again following the same previous method. Over the range of 0s delay to 200ms delay, each test was run for five minutes, and network statistics were recorded every second. The observed throughput of the SLE protocol for the forward and return links are shown in Figures 5 and 6, respectively. As you can see, the throughput of the SLE connections is significantly reduced as the RTT delay of the network is increased.

Figures 5 and 6 also show the maximum theoretical throughput of the link given our assumption that blocking calls are being used for the transfer of SLE PDUs to the underlying transport layer protocol. Since the SLE protocol specification allows for a maximum SLE PDU size of $64kB$, the theoretical maximum throughput for the SLE connection is given in Equation 1.

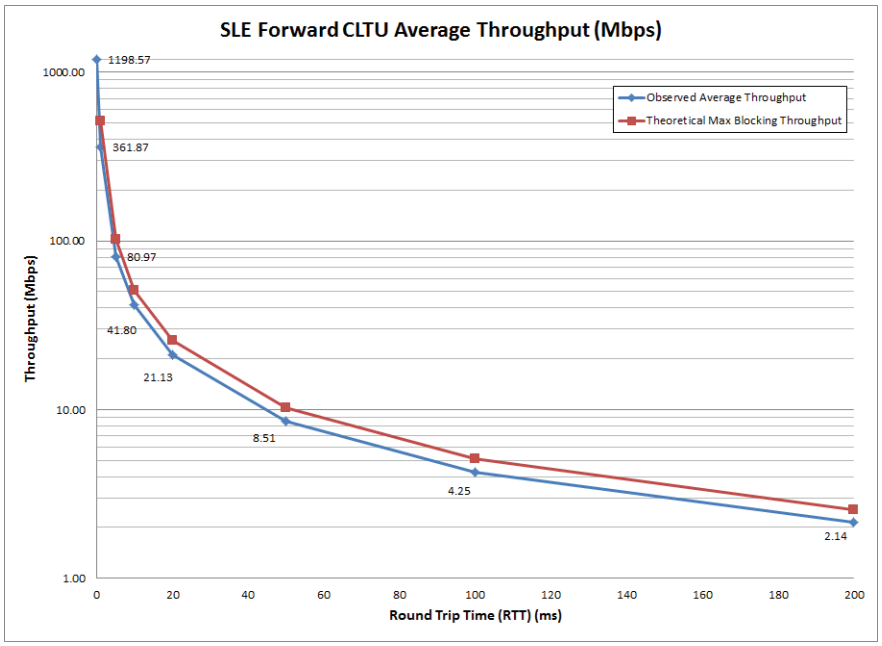

Figure 5. SLE Forward Link Connection throughput vs. delay from 0ms delay to 200ms delay.

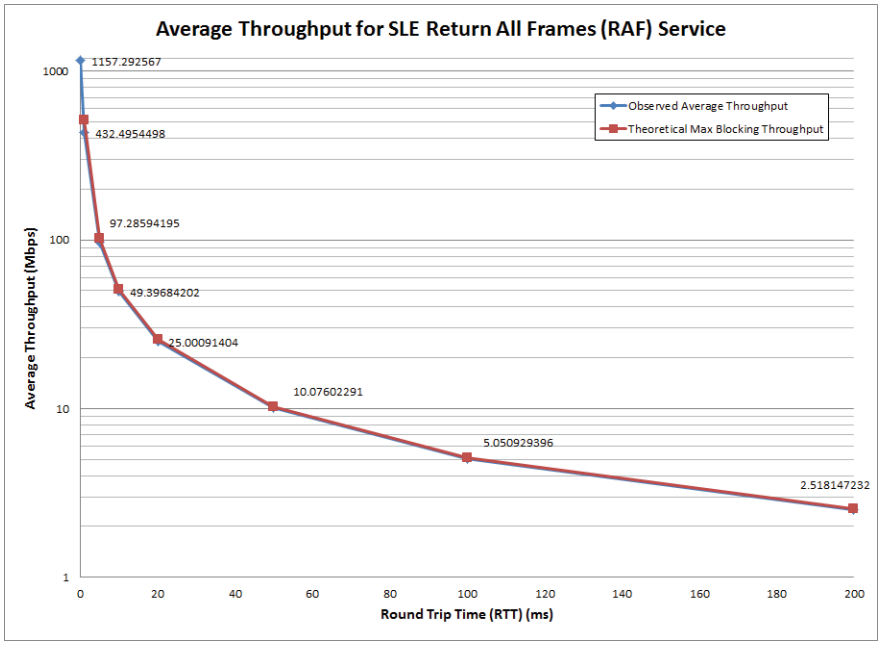

Figure 6. SLE Return Link Connection throughput vs. delay from 0ms delay to 200ms delay.

$$R = S/T, \tag{1}$$

where R is the maximum achievable data rate, S is the SLE PDU size transferred to the transport layer, and T is the expected RTT of the network connection.

B. Space Link Extension (SLE) High Data Rate Performance Test

In addition to testing the link in network connections with increasing RTT delay, we also tested the performance of the SLE protocol at higher data rates. This allowed us to see the maximum data rate at which the SLE protocol was able to operate.

The following tests were set up similarly to those seen in Section IV.A, with 64 kB PDUs and data rates from .5$Gbps$ to 1.6$Gbps$. The data rates were tested in increments of .1$Gbps$,and each test ran for 5 minutes, and the delay in the link was found every second. The results for all the tests are shown below:

The increase in delay from 1.3$Gbps$ to 1.4$Gbps$, and then the subsequent decrease from 1.4$Gbps$ to1.5$Gbps$ was of interest. A plot of the 1.4$Gbps$ test is shown for the point where the 1.4$Gbps$ test ran into increased delay. Figure 9 is a more detailed view of Figure 8.

As can be seen, when the data rate drops around 790 seconds, the delay increases dramatically. As the data rate is increased to send the delayed packets through, the delay drops as the built up queue is sent over the link. This is due to a packet drop and a TCP back off which will reduce the sliding window, and reduce throughput.

V. Conclusion

Our initial observations point to this being one of two issues: either the TCP transport layer is not properly increasing the TCP congestion window to allow for more data to be transmitted in this high bandwidth delay product environment, or the CGS SLE API is not sending the data to the transport layer fast enough support a high data rate.

To rule out the first case, as you can see above in Section IV.B we have tested the network connection with other network performance software. Namely, with the network transmit / receive blocks within the CGS framework. These tests were run with the same underlying TCP settings on the same network interfaces. As you can clearly see in Figure 1, the performance of this CGS function does not suffer the same performance hit as the CGS SLE API implementation.

The base network test and the SLE connection tests were carried with the same underlying TCP protocol. While the base network connection was well able to handle RTT delay well up to 200ms, while the SLE connection was not. It appears that the implementation of the SLE protocol uses blocking calls, an option that is listed under the CCSDS SLE Forward Link Standards.

The blocking option interacts between the SLE protocol and the TCP transport layer. A CLTU-TRANSFER-DATA command will send a PDU created by SLE to the transport layer. The CLTU-TRANSFER-DATA command will return from the transport layer and indicate if the packet was successfully received or return an error. The SLE protocol will block all processing until the CLTU-TRANSFER-DATA return command is received, which only occurs once the TCP connection returns an ACK. In high bandwidth links such as the 1.2Gbps link, this will significantly reduce throughput. In this situation, only one packet of L bytes can be sent every X seconds, where X is the RTT of the link. The throughput would then be reduced to L/X, a much lower throughput than if a non-blocking system was used.

Furthermore, observations on the SLE protocol operating in a 0ms RTT delay environment was also of note. The SLE protocol under the 1.4Gbps throughput test began to suffer from delay and decreases in throughput, as seen in Figure 7. These figures show the reaction from the TCP protocol when a packet is dropped, or an ACK is not received from the destination during the time-out window. The subsequent downsizing and build up of the TCP sliding window can be seen more clearly in Figure 9 . In high bandwidth/delay links, TCP requires a large sliding window, which will contain many unacknowledged packets. When an ACK is received out of order, the sliding window will decrease in size and be slowly readjusted back to original TCP sliding window size, which takes additional time, and will introduce more delay into the connection. Unless a significantly large queue has been built at the ground station, this service outrage will introduce too much delay into the link, which could cause a drop in connection, a loss of synchronization of radios, and cause packets to be dropped.

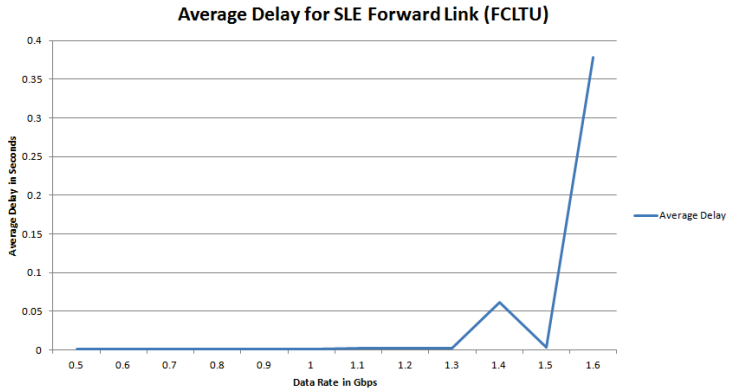

Figure 7. SLE Forward Link Connection delay for throughput from 500Mbps-1.6Gbps.

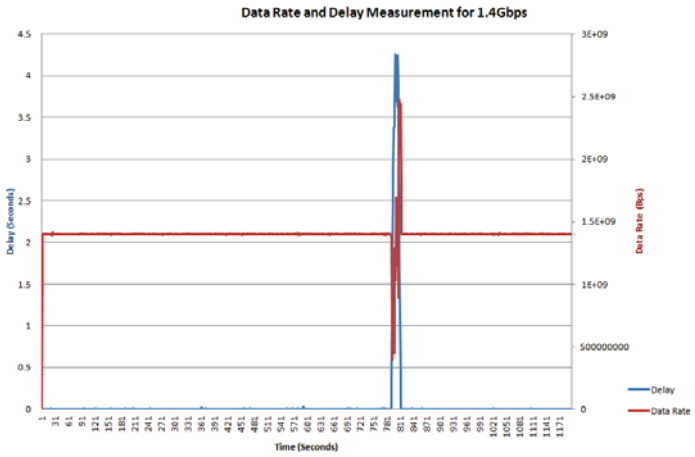

Figure 8. SLE Forward Link Connection throughput when operating at 1.4Gbps.

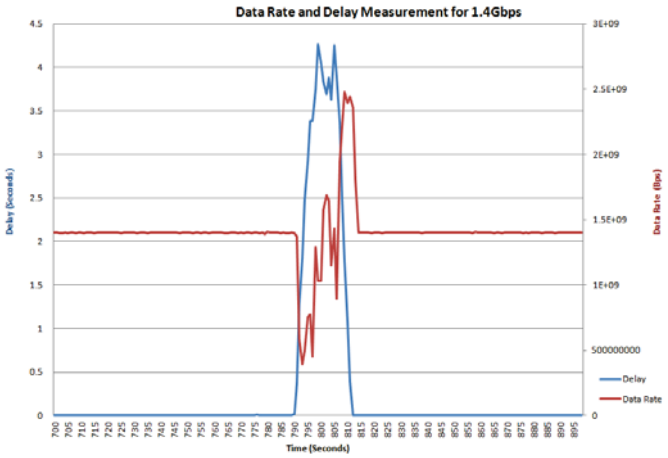

Figure 9. SLE Forward Link Connection throughput detail when operating at 1.4Gbps.

A. Future Work

The SLE protocol will be further tested in a network with additional realistic network conditions, such as packet loss and jitter. The inclusion of these conditions is important, as the underlying TCP protocol will react to the loss of a packet by throttling back the connection to ensure that all packets are received. This reaction to other network imperfections may cause further loss of throughput for the SLE connection. Furthermore, the SCENIC lab will be expanded to include end-to-end wireless communication emulation for the space communication link as well.

Acknowledgments

This work was performed as an intern for the NASA Space Communication and Navigation (SCaN) Internship Program (SIP) with Dr. Robert Murawski as a mentor. The authors would like to acknowledge NASA Glenn Research Center (GRC) SCENIC Emulation Lab and Program Manager, Richard Slywczak, for providing the facilities used in this research.

References

[1]Götzelmann, M., Karch, M., Lucia, D., Abello, R., and Dreihahn, H., "High Rate Telemetry Transfer - CCSDS SLE And Beyond," *AIAA SpaceOps 2012*, 2012.

[2]CCSDS Recommended Standard, "Space Link Extension - Forward CLTU Service Specification," *CCSDS 912.1-B-3*, July 2010.

[3]CCSDS Recommended Standard, "Space Link Extension - Return All Frames Service Specification," *CCSDS 911.1-B-3*, January 2010.

[4]CCSDS Recommended Practice, "Space Link Extension - Application Program Interface for the Forward CLTU Service," *CCSDS 916.1-M-1*, October 2008.