NASA/TM—2015-218453

Control and Non-Payload Communications (CNPC) Prototype Radio—Generation 2 Security Architecture Lab Test Report

Dennis C. Iannicca
Glenn Research Center, Cleveland, Ohio

James H. McKim
Wyle Information Systems, Cleveland, Ohio

David H. Stewart and Suresh K. Thadhani
Jacobs Engineering, Cleveland, Ohio

Daniel P. Young
DB Consulting Group, Inc., Cleveland, Ohio

May 2015

NASA STI Program . . . in Profile

Since its founding, NASA has been dedicated to the advancement of aeronautics and space science. The NASA Scientific and Technical Information (STI) Program plays a key part in helping NASA maintain this important role.

The NASA STI Program operates under the auspices of the Agency Chief Information Officer. It collects, organizes, provides for archiving, and disseminates NASA's STI. The NASA STI Program provides access to the NASA Technical Report Server—Registered (NTRS Reg) and NASA Technical Report Server—Public (NTRS) thus providing one of the largest collections of aeronautical and space science STI in the world. Results are published in both non-NASA channels and by NASA in the NASA STI Report Series, which includes the following report types:

- TECHNICAL PUBLICATION. Reports of completed research or a major significant phase of research that present the results of NASA programs and include extensive data or theoretical analysis. Includes compilations of significant scientific and technical data and information deemed to be of continuing reference value. NASA counter-part of peer-reviewed formal professional papers, but has less stringent limitations on manuscript length and extent of graphic presentations.

- TECHNICAL MEMORANDUM. Scientific and technical findings that are preliminary or of specialized interest, e.g., "quick-release" reports, working papers, and bibliographies that contain minimal annotation. Does not contain extensive analysis.

- CONTRACTOR REPORT. Scientific and technical findings by NASA-sponsored contractors and grantees.

- CONFERENCE PUBLICATION. Collected papers from scientific and technical conferences, symposia, seminars, or other meetings sponsored or co-sponsored by NASA.

- SPECIAL PUBLICATION. Scientific, technical, or historical information from NASA programs, projects, and missions, often concerned with subjects having substantial public interest.

- TECHNICAL TRANSLATION. English-language translations of foreign scientific and technical material pertinent to NASA's mission.

For more information about the NASA STI program, see the following:

- Access the NASA STI program home page at http://www.sti.nasa.gov

- E-mail your question to help@sti.nasa.gov

- Fax your question to the NASA STI Information Desk at 757-864-6500

- Telephone the NASA STI Information Desk at 757-864-9658

- Write to:
  NASA STI Program
  Mail Stop 148
  NASA Langley Research Center
  Hampton, VA 23681-2199

NASA/TM—2015-218453

Control and Non-Payload Communications (CNPC) Prototype Radio—Generation 2 Security Architecture Lab Test Report

Dennis C. Iannicca
Glenn Research Center, Cleveland, Ohio

James H. McKim
Wyle Information Systems, Cleveland, Ohio

David H. Stewart and Suresh K. Thadhani
Jacobs Engineering, Cleveland, Ohio

Daniel P. Young
DB Consulting Group, Inc., Cleveland, Ohio

National Aeronautics and
Space Administration

Glenn Research Center
Cleveland, Ohio 44135

May 2015

This report is a formal draft or working paper, intended to solicit comments and ideas from a technical peer group.

This report contains preliminary findings, subject to revision as analysis proceeds.

Level of Review: This material has been technically reviewed by technical management.

Available from

NASA STI Program
Mail Stop 148
NASA Langley Research Center
Hampton, VA 23681-2199

National Technical Information Service
5285 Port Royal Road
Springfield, VA 22161
703-605-6000

This report is available in electronic form at http://www.sti.nasa.gov/ and http://ntrs.nasa.gov/

Control and Non-Payload Communications (CNPC) Prototype Radio—Generation 2 Security Architecture Lab Test Report

Dennis C. Iannicca
National Aeronautics and Space Administration
Glenn Research Center
Cleveland, Ohio 44135

James H. McKim
Wyle Information Systems
Cleveland, Ohio 44135

David H. Stewart and Suresh K. Thadhani
Jacobs Engineering
Cleveland, Ohio 44135

Daniel P. Young
DB Consulting Group, Inc.
Cleveland, Ohio 44135

Summary

NASA Glenn Research Center, in cooperation with Rockwell Collins, is working to develop a prototype Control and Non-Payload Communications (CNPC) radio platform as part of NASA Integrated Systems Research Program's (ISRP) Unmanned Aircraft Systems (UAS) Integration in the National Airspace System (NAS) project. A primary focus of the project is to work with the FAA and industry standards bodies to build and demonstrate a safe, secure, and efficient CNPC architecture that can be used by industry to evaluate the feasibility of deploying a system using these technologies in an operational capacity. GRC has been working in conjunction with these groups to assess threats, identify security requirements, and to develop a system of standards-based security controls that can be applied to the current GRC prototype CNPC architecture as a demonstration platform.

The security controls were integrated into a lab test bed mock-up of the Mobile IPv6 architecture currently being used for NASA flight testing, and a series of network tests were conducted to evaluate the security overhead of the controls compared to the baseline CNPC link without any security. The aim of testing was to evaluate the performance impact of the additional security control overhead when added to the Mobile IPv6 architecture in various modes of operation. The statistics collected included packet captures at points along the path to gauge packet size as the sample data traversed the CNPC network, round trip latency, jitter, and throughput. The effort involved a series of tests of the baseline link, a link with Robust Header Compression (ROHC) and without security controls, a link with security controls and without ROHC, and finally a link with both ROHC and security controls enabled. The effort demonstrated that ROHC is both desirable and necessary to offset the additional expected overhead of applying security controls to the CNPC link.

1.0 Introduction

NASA is currently working with the Federal Aviation Administration (FAA), RTCA Inc. (a Federal Advisory Committee), and other organizations to develop technologies and procedures to allow Government (public) and commercial (civil) unmanned aircraft (UA) to operate safely in the National Airspace System (NAS). One aspect of this effort is to design and test a concept prototype network that

will allow a UA to fly in the NAS while being controlled by a Pilot in Command (PIC) from a Control Element (CE) located elsewhere on the ground, thus creating an Unmanned Aircraft System (UAS). An essential element of this new national capability is the radio communications channel linking each UA to a network of fixed ground stations (GSs). Data communication necessary for flight is referred to as Control and Non-Payload Communication (CNPC) and is exchanged between each UA and a GS to ensure safe, reliable, and effective UA flight operation.

This paper focuses on the research effort to better understand the requirements for secure communications over the concept prototype network and assesses standards-based cryptographic security controls that allow for confidentiality, integrity, non-repudiation, and anti-replay capabilities. The overarching goal of secure communications is to provide safety of flight.

A lab test network of several interconnected systems was setup to emulate GRC's UAS flight lab environment (Figure 1). The lab test network consisted of one UA connected to one GS using actual Rockwell Collins prototype radios providing the air-ground RF link. Networking equipment to support the mobile routing of information from the CE to the UA was also used to verify that messaging from a PIC to a UA is achieved.

To implement the security controls, a standards-based cryptographic security extension to the Internet Protocol (IP) called IPsec is used to secure the communications transmission through the network and over the air-ground RF link. Cryptographic techniques inherently add an overhead to the transmission of information and can adversely affect the throughput and overall capacity of the network so techniques like header compression are used to offset the overhead of such cryptography. The purpose of the experiment is to characterize the security overhead against the baseline Mobile IPv6 architecture and to evaluate whether the ROHC is successful in allowing us to offset the additional overhead of the cryptographic security controls.

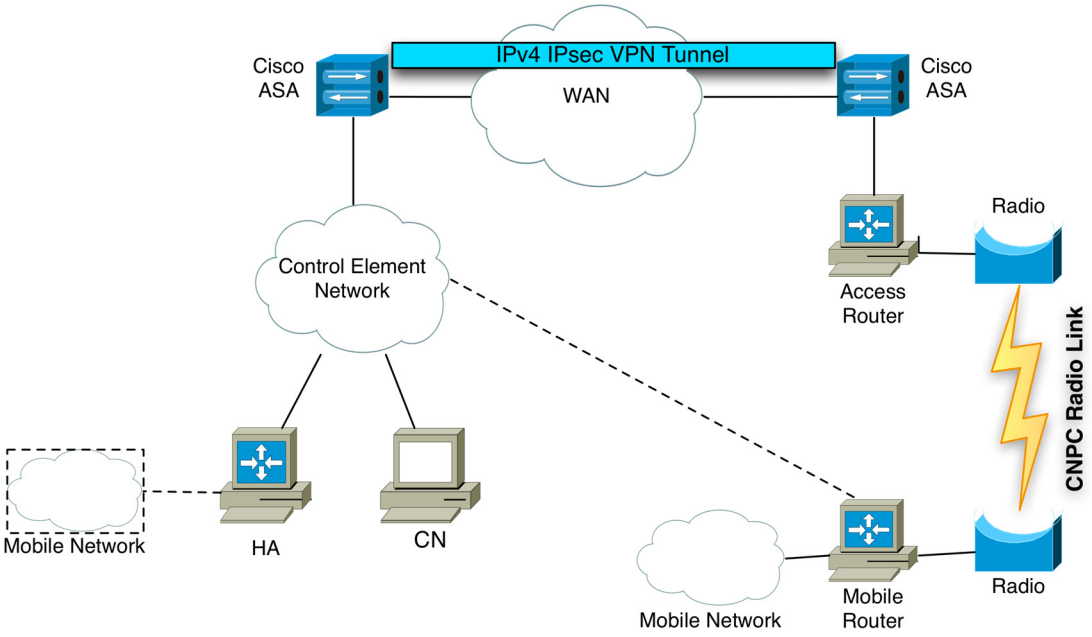

Figure 1.—UAS Security Test Bed.

2.0 Test Environment Description

2.1 Overview

The security test bed is configured to mimic the mobile IPv6 environment used by the prototype flight lab. The architecture consists of three major components: the control element network, the ground station, and the mobile node or router aboard the aircraft. Within the control element network is the mobile IPv6 home agent and one or more correspondent nodes that need to communicate with the mobile node.

A home agent (HA) is placed in the control element network, or "home network", and is used as a central communications point for contacting the mobile node. The HA serves to maintain a binding database of a mobile node's current foreign network address, or care-of address (CoA). The bindings map the home address, a known address in the home network, and a mobile network (denoted here as the Avionics Network), to the shifting care-of address in the foreign network in order to transparently reroute traffic from external users.

Each ground station (GS) serves as a broker providing a UAV with its foreign network IPv6 address. This is done by means of an IPv6 access router (AR) which communicates with the aircraft through the prototype radio. Router advertisements or other dynamic host configuration methods can then be used to provide the UAV with a network address. The ground station does not need to be owned by the same agency as the aircraft and will have address space associated with the terrestrial service provider used at that location.

The aircraft communicates to the ground station using the same prototype radio. A mobile IPv6 router attached to the radio allows the NEMO network to communicate with the home network seamlessly. This allows the various components behind the mobile router to be addressed directly by using their "home" address. Users would not need to know the addressing schemes used at the various ground stations.

2.2 Internetworking

In a production environment, communication between endpoints would utilize IPv6 end-to-end, without the need for any encapsulation of traffic within legacy IPv4 packets for transmission across the Internet. However, the perimeter network in use does not currently route IPv6 traffic natively. In order to emulate a native IPv6 environment, "6in4" (IP Protocol 41) tunnels were created between the home agent and the ground station computer system to allow for utilization of the readily available IPv4 backhaul communications networks. While there are several alternative ways to configure the test architecture to transmit IPv6 packets across an IPv4 network, the 6in4 approach reduced configuration complexity and eliminated the need for devices between the home agent and ground station computer to be IPv6-compatible.

The home network and the ground stations are each protected by a dedicated firewall system that serves as both an IPv4 network firewall as well as a VPN gateway to secure the transmission of data across the Internet. Each GS utilizes an IPv4 IPsec VPN connection to communicate with the home network and carry the 6in4 traffic between the home agent and the ground station computer where it is unencapsulated back to native IPv6 packets.

While the security test bed environment implements this "6in4" encapsulation in order to ensure complete compatibility with the production flight lab architecture, the simulated terrestrial IPv4 IPsec VPN link over the "Internet" has no design relevance to the CNPC security architecture recommendations under evaluation by this experiment. The additional overhead of the VPN gateways and IPv4 VPN link observed were negligible.

2.3 Addressing Scheme

The home network and ground stations were designed to support a dual-stack IPv4/IPv6 environment in order to support direct communications with legacy devices such as UPS battery backup network interfaces or out-of-band management interfaces on the ground station computers that may not support IPv6. In addition, an IPv4 addressing scheme was needed to serve as an endpoint for the 6in4 tunnels between the home agent and the ground station computer.

The ground station computers were configured to use the Router Advertisement Daemon "radvd" in order to support IPv6 neighbor discovery across the RF link by multicasting periodic Router Advertisement (RA) packets every 10 sec. The remote aircraft computer, upon receipt of one of these RA packets, would be able to self-configure its IPv6 Care-of Address (CoA) and begin communicating with the ground station, and thus the home agent, at the network layer.

2.4 Network Mobility Concept

The Mobile IPv6 (Ref. 1) with NEMO (Ref. 2) design allows the aircraft to maintain seamless network-layer communications between the pilots's ground control station and the onboard avionics systems as the unmanned aircraft transitions between ground stations. This design may result in a few dropped packets, but established sessions should continue on without needing to reconnect. This method eliminates the costly procedure of tearing down and re-establishing active sessions.

The Mobile IPv6 network shown in Figure 1 starts configuring after the CNPC radios connect. Then the Ground Station Router (GSR) (aka AR) sends a Routing Advertisement (RA) via the CNPC radios to the Mobile Router (MR). The MR using the RA creates a CoA, providing the MR with a global address that allows communication between the MR and HA.

The MR then sends a Binding Update (BU) that contains the MR's Home Address (HOA) and CoA to the HA, providing the HA with MR's latest point of attachment to the network. The HOA is a global address from the home network (located in the Control Element) address space and is used by the HA to identify the MR. After authenticating the BU, the HA replies with a Binding Acknowledgement (BA) and creates an IPv6 dynamic tunnel between the HA and the MR.

As an example, when a ping is initiated from the Correspondent Node (CN), it has a destination address in the Avionics Network behind the MR[1] (located in the Unmanned Aircraft). Due to the HA advertising this address space any traffic intended for an address on or behind the MR will always be routed to the HA first, for this reason the CN forwards the ping request to the HA. HA's advertising of this address space eliminates the need for and delay of routing table updates that are usually needed when a router changes it point of attachment to a network, thus eliminating connectivity outages due to routing table convergence.

The HA encapsulates, then forwards the ping request to the MR's CoA via the MIP6 tunnel between the HA and MR. Subsequently the MR removes the tunnel header and forwards the original ping request to its final destination, triggering a ping reply as the response. The ping reply follows the same route back for the return trip to the CN.

2.5 CNPC Radio

The prototype CNPC radios used to provide connectivity for the simulated air-ground link of the UAS Security Test Bed are the same as the radios used in the UAS flight-testing environment. Due to spectrum restrictions, these radios have bandwidth limitations that increase latency due to packet queuing and fragmentation.

[1] For testing purposes the Avionics address is actually assigned to a virtual interface on MR.

The waveform of the radio is Time Division Duplexing (TDD) with 50ms time frames (20 frames/ sec). Each 50 ms time frame has an uplink and corresponding downlink transmission. Due to data requirements, the uplink (ground-to-air) and downlink data payload sizes are different (55 bytes for the uplink and 103 bytes for the downlink per frame). The radio configuration was identical for all tests, using 10 of the available 20 frames.

2.6 IPsec

IPsec is a security extension to the Internet Protocol (IP) that provides end-to-end security protection of network layer traffic. It allows for mutual authentication of nodes at the establishment of a session and renegotiation of cryptographic keys used during an ongoing session. Mechanisms are available to provide confidentiality, data integrity, data-origin authentication, non-repudiation, and anti-replay capabilities. IPsec can be used in two different modes: transport, which protects data communications between two hosts, and tunnel which protects communications between two gateways allowing for secure communications between one or more hosts on each side of the tunnel. In addition, IPsec offers transport and tunnel mode with two protocols: Authentication Headers (AH) and Encapsulating Security Payloads (ESP).

For testing secure transmission over the CNPC network, IPsec ESP tunnel mode was used to encapsulate packets that were transmitted between the CN (behind the HA) and the avionics network (behind the MR). Using ESP in tunnel mode was a requirement of integrating IPsec with the UMIP (Ref. 3) mobile IPv6 implementation used by the current architecture. In order to attempt to offset the overhead of the IPsec ESP encapsulation with our ROHC implementation, a null encryption cipher was used along with an HMAC-SHA256 authentication cipher implementation that produced a 96-bit truncated integrity check value (ICV) in order to expose the clear text inner IPv6 header to the ROHC library's profile analyzer. At the time of testing, confidentiality of the CNPC link was not an expected system requirement so it was deemed sufficient to perform the testing using only data-integrity and authentication which would still protect the CNPC packets and ensure PIC to UA commands and UA to PIC telemetry was not modified in transit.

For the purposes of this phase of testing due to problems integrating the UMIP mobile IPv6 implementation with the Strongswan (Ref. 4) IKEv2 daemon, testing was unfortunately limited to using static keying and statically defined security associations using the Linux IPsec-tools "setkey" tool rather than being able to perform testing using the IKEv2 dynamic keying protocol. A set of security associations was statically defined between the HA and MR and used by the UMIP daemon for dynamically allocating the tunnel resources. While the performance overhead of the IKEv2 key exchanges and security association establishment was not able to be tested, the static security associations provided an accurate representation of the overhead of IPsec ESP encapsulated data in transit for the purposes of the rest of the test goals.

A secure tunnel is established by UMIP between the HA and the MR, by utilizing the pre-established security association at the HA and the MR. All packets transmitted over this tunnel are authenticated at the source and verified at the destination. IPsec ESP creates an overhead in the IP packet structure and results in a delay with potential impacts to system throughput. This is especially noticeable in transmission over the air-ground link through the CNPC radio and ROHC was expected to help offset the cryptographic overhead associated with the security encapsulation.

2.7 Robust Header Compression (ROHC)

ROHC is a mature specification based on a lengthy history of experiments to reduce the overhead of network protocol headers. There has been significant interest in compressing headers in situations where communication links have limited capacity and where the ratio of header to payload is large.

In the UAS CNPC waveform, reducing this overhead is of critical importance. The anticipated nature of this communication will be a large number of relatively short messages, each message with an associated set of protocol headers. There is a degree of real-time sensitivity—the traffic should be moved

forward expeditiously. The headers themselves will be large and there will be multiple "nested" headers (i.e., extension headers).

The ROHC specification defines a process that will - under ideal circumstances, compress the IPv6 header, including all the extensions that must be used in the test configuration, in most of the messages (most of the packets) to one or two bytes. It does this by removing invariant components in the header and providing ways to enable the de-compressor to anticipate predictable components (fields), obviating the need to pass those fields across the communication link.

In the ROHC specification, in addition to the framework specification, there are several protocol specific specifications or profiles. The objective of each profile is to identify protocol header field's particular to that protocol to optimally compress it. In an implementation the state of a "session" is maintained between the two pair of ROHC compressor and ROHC de-compressor operating at the two endpoints. For header fields that are invariant between packets (for instance source and destination addresses), that information need only be communicated once. All subsequent packets that are part of that "stream" can omit that information. Similarly, for some other fields that change predictably (for instance counters); the information need only be communicated once. For fields that change rarely, the compressor need only communicate when that field changes. In this latter case, a single packet will be slightly larger (but still far smaller than an entire uncompressed header packet). The ROHC profiles try hard to minimize overhead, and for some protocols are elaborate and complex.

"The Open Source ROHC Library" (ROHC Library) (Ref. 5) was used for tests identified in this report. The library started as a fork of the ROHC Project 2003 at the Lulea University of Technology in Sweden but has since greatly evolved into a community-supported open source project with little resemblance to the original project's code base. While investigating ROHC work, other alternate, easily available, (e.g., open source) versions were found to be less complete or of lower quality. There are also two commercial implementations that were investigated but could not practically be used due to cost or peculiarities in their licensing.

The primary testing concern was with the IP and the UDP ROHC profile specifications in an effort to compress the IP, ICMP, ESP, and UDP headers. In a production system, a TCP profile specification would be needed to help reduce TCP's 20 byte header overhead but the TCP profile implemented in the ROHC library at the time of testing was too immature and unstable to finish the evaluation. It was decided to forego attempts to use the TCP profile for this phase of testing and restricted our test traffic to ICMP and UDP packets.

3.0 Test Procedures

3.1 Network Setup

The following steps were used to configure the components of the Mobile IPv6 network.

- Upload configuration files to the radios that set up Link Streams and establish Link Stream identification numbers.
- Start the radio connection software that brings up the Linux tunnel interfaces on the GS Router and Mobile Router and establishes a data connection to the radio interface using the configured Link Stream IDs. The radio connection software is also where the ROHC option is enabled if needed.
- Configure and start the mip6d daemons on the home agent and mobile router. This has to be done after Step 2, because the mip6d daemon uses the tunnel interface that is created in Step 2. The configuration file of the mip6d daemon, determines if the MIP6 tunnel uses IPsec ESP or ordinary IPv6-in-IPv6 encapsulation for tunneling the data traffic.
- Start tcpdump captures on the following interfaces:

Correspondent Node Interface:

tcpdump -i eth1—capture the pings and udp-pings response and request packets.

Home Agent Interface:

tcpdump -i any—capture all traffic passing thru the HA for reference, in case of unexplained event in data.

tcpdump -i six0—capture all IPv6 traffic passing thru HA on the way to the ground station access router.

GS Router interface:

tcpdump -i six0—capture all IPv6 traffic passing thru GS Router from the HA.

tcpdump -i rftun0—the aforementioned Linux tunnel established between the computer and the radio. This should capture all traffic being passed to/from the radio connection software for the CNPC radio.

tcpdump -i eth2—capture all traffic being sent/received to/from the CNPC radio. This is one of two captures that will show the packet modifications made by the ROHC library.

Mobile Router Interface:

tcpdump -i rftun0—capture all traffic being passed to/from the radio connection software for the CNPC radio.

tcpdump -i any—capture all traffic passing thru the HA for reference, in case of unexplained event in data.

tcpdump -i eth2—capture all traffic being sent/received to/from the CNPC radio. This is one of two captures that will show the packet modifications made by the ROHC library.

3.2 Test Configurations

These four test configurations were used throughout the testing process. Table 1 is provided for quick reference.

- **Radio**
  In this test, the MR connects via the radio, the MIP IPsec option is off and the ROHC is off. The Radio test is the baseline test, exhibiting the latency and throughput of the system without the overhead of IPsec or the decrease in bandwidth of the ROHC.
- **Radio, IPsec**
  In this test, the MR connects via the radio, the MIP IPsec option is on and the ROHC is off. The Radio, IPsec test reveals the effects of the increased overhead of the IPsec tunnel.
- **Radio, ROHC**
  In this test, the MR connects via the radio, the MIP IPsec option is off and the ROHC is on. The Radio, ROHC test demonstrates the benefits of the ROHC algorithm without the increased overhead of the IPsec tunnel.
- **Radio, IPsec, ROHC**
  In this test, the MR connects via the radio, the MIP IPsec option is on and the ROHC is on. The Radio, IPsec test validates the benefits of the ROHC algorithm with the increased overhead of the IPsec tunnel. This test represents the configuration that is expected to be deployed in a future CNPC system.

TABLE 1.—TEST CONFIGURATION SUMMARY

Attributes / Test Name	Radio	IPsec	ROHC
Radio	ON	OFF	OFF
Radio, IPsec	ON	ON	OFF
Radio, ROHC	ON	OFF	ON
Radio, IPsec, ROHC	ON	ON	ON

3.3 Performance Testing

Ping utilities were used to generate the test traffic. Two different utilities were used. Both feature adjustable payload size and adjustable generation rate. After different payload sizes and packet intervals were tested a payload size of 40 bytes and the rate was fixed at two packets per second (0.5 sec interval) were selected. These payload parameters were selected to prevent dropped packets.

Ping6 is the traditional ping program, enhanced to work using IPv6. It sends Internet Connection Management Protocol (ICMP) echo request packets to a destination IPv6 address. The destination, on receipt will echo the packet back to the source as an ICMP echo reply. The originator records the round trip time of the replies (or lack of replies). Ping was originally developed only to test network connectivity and might well be considered the second oldest Internet application—it is been around for a long time.

Udp_ping was written to emulate the anticipated type of traffic the CNPC network will see when real equipment is configured to use the CNPC links. The initial concern with using ICMP traffic was with how the ROHC algorithm would handle ICMP traffic. ROHC defined several profiles, each of which is specific to a particular type of traffic. The profile used for ICMP traffic is different from the one used for UDP traffic. As UDP traffic will predominate in actual use scenarios, it was felt using UDP test traffic would yield more realistic measurements in test simulations.

During testing, it was found that an artifact of the udp_ping program, that it carefully maintains synchronization with the system clock, had a significant effect on how traffic flows through the simulated network. Our overall understanding of how traffic might potentially interact with the CNPC RF radio links was increased. Note, this observation is specific to the particular model of radios used in the test, but the considerations might well apply to any radio that depends on closely time synchronized operations.

3.4 Data Capture

The *tcpdump* utility was used to monitor the test traffic at several points in the demonstration network. Arguably this could be described as the Swiss-army-chainsaw of network traffic monitoring utilities, it features a wealth of configurable options that allow capture of different types of network traffic coupled with accurate recording (logging) and time stamping of that traffic. Post test, the recorded logs were re-processed by tcpdump to extract data used to produce the charts and measurements.

4.0 Data Analysis/Results

4.1 Brief

The analysis of our testing shows that the baseline ping packet structure that is transmitted over the RF CNPC radio interface was adversely impacted by an 18.7 percent or 24 byte increase in size due to the addition of IPsec ESP tunnel mode encapsulation. This resulted in an observed reduction in throughput by 18.3 percent.

The addition of ROHC to the ESP encapsulated IP packet structure shows that the transmitted packet over the RF CNPC radio interface was positively impacted by 30.2 percent or 8 byte reduction in size due to the addition of ROHC along with IPsec ESP. This resulted in an observed increase in throughput by 29.6 percent.

In our testing the base payload was a 40 byte UDP packet that was generated at the CN and routed to the MR using a ping utility. As the 40 byte payload is prepared for transmission across the network to its destination, addressing and other network parameters are added for routing. This is considered overhead as it is not part of the payload information.

For the baseline test the payload plus overhead structure is depicted in Figure 2. The baseline test was performed to characterize the network performance absent any security control and also absent any header compression techniques.

The next test added security control to the baseline UDP packet and it is clear that the overall packet size has increased in size by 24 bytes, see Figure 2. The increase in size and addition of security has a negative impact to system throughput and delay in our CNPC prototype network.

For the ROHC only test, ROHC was analyzed by implementing it with the baseline payload only—absent any security control. The packet structure for the payload and headers shows that the overall packet size is drastically reduced when using ROHC, see Figure 3. ROHC is able to reduce the Outer IP Header, Original IP Header and UDP Header from 88 bytes down to 4 bytes as shown in figure 4. This is a significant reduction in overhead and should reap great benefits when using security controls. It is concluded that ROHC is an excellent mechanism for header compression and positively impacts system throughput and delay.

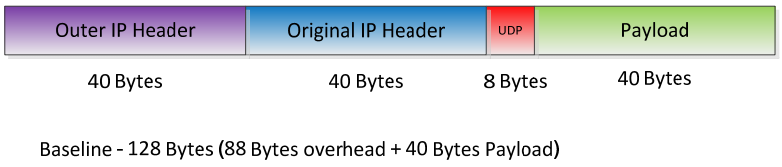

Figure 2.—Radio.

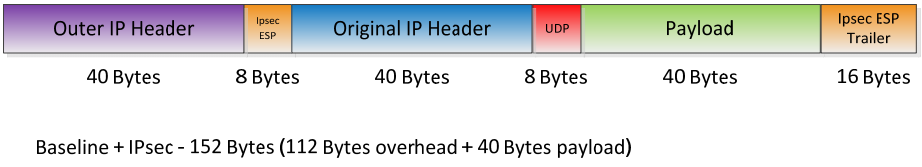

Figure 3.—Radio, IPsec.

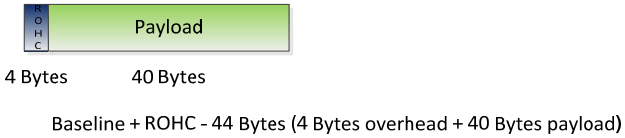

Figure 4.—Radio, ROHC.

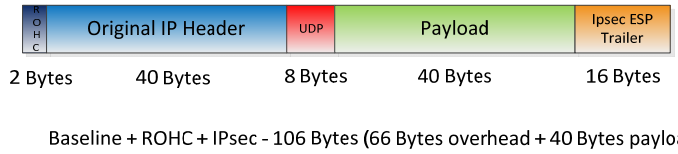

Figure 5.—Radio, IPsec, ROHC.

Finally, ROHC and security control were added to the baseline payload. This test demonstrates how security controls along with ROHC will work together in the network and how the system throughput and delay will perform. The packet structure of the payload with security control and ROHC is depicted in Figure 5. It is a reduction of 8 bytes in size from the packet structure in Figure 3 with IPsec and no ROHC.

The above information comes from the packet captures as shown in Appendix A. The data in Appendix A shows sample individual packets from the *tcpdump* data captures at different points on the network as the packets travels from the CN to the MR.

4.2 Loopback Only—Packet Throughput and Return Time

While this is not part of the formal testing, it's instructive to see how traffic would be conveyed in an ideal network—one that has minimal latency, essentially unlimited bandwidth and high reliability. For a set of different configurations, a per-configuration test was run during which traffic was sent bidirectionally between two endpoints. The adjusted parameters were:

- The inclusion or exclusion of IPsec
- The inclusion or exclusion of Robust Header Compression (ROHC)

For reference purposes, these tests were also run bypassing the radio, although the results are not apropos to this discussion of traffic through the radio and largely not included in this paper, this section being the sole exception.

Sending data traffic through a limited bandwidth radio link imposes characteristics on the traffic we are unaccustomed to in our ordinary world of fat, fast data pipes. A significant delay is imposed on the traffic as it transits the radio link. The radios and waveforms used in this test also function to sharply limit link bandwidth. Outside the radio link, we are accustomed to LANs with on the order of gigabits per second link capacity. The link through the radio provides on the order of kilobits per second, or some hundreds of bytes per second. The amount of information we would like to send through the radio links is large, compared to the link's capacity, so using the link efficiently is of critical importance.

Other factors complicate the issue more. For interoperability, anticipated future expansion (i.e., many users, many aircraft, large geographic region), and several other factors, the underlying (layer 3) protocol over which all ground station to UA traffic must flow has been specified as IPv6 in order to future-proof the system concept due to the growing inability to obtain IPv4 address space. IPv6 has a relatively larger overhead to other protocols. Additional overhead was introduced by the addition of Mobile IPv6 and IPsec ESP. These are necessary if aircraft are to be able to move between the individual ground station cells managed by a communications service provider. IPsec adds additional overhead to the traffic that must pass through the radio link.

To remediate the link capacity consumed by this overhead as much as possible, the ROHC protocol is used to compress the overhead, the large packet headers, used by IPv6, Mobile IP, and IPsec.

A brief discussion of the test components:

During each test run, two sets of test data lasting 10 min were sent between the endpoints. Traffic was generated at the rate of two packets per second, 40 byte payload. The first set was of ICMP echo request and concomitant echo reply traffic. Then 10 min of UDP echo request and reply traffic.

The traffic was monitored and measurements taken at several points in the test network; at both radios, and at other intermediary nodes as well as the source and destination of the test traffic. Logging the time ‘request’ for traffic originating at the ground station endpoint and logging the time ‘reply’ that the same end point receives was of primary interest.

Measuring the effective throughput at the radio is of primary interest. This latter measurement can only be derived, and is obtained by counting the amount of data (i.e., the number of bytes) that pass through the radio during each one second “slot” of the test. Because of the low per second data rate through the radio, quantization (i.e., low counts that might imply an incorrect measurement) is an issue. This measurement is thus imprecise and so in the chart displaying the measurement; it has been smoothed to suppress what would otherwise be a very jagged and less informative trace.

Figure 6 serves to demonstrate an ideal behavior of traffic between two endpoints. No artifacts are imposed by limited bandwidth links, lack of flow control, or high latency. Note the spikes: these are evidence that as a bona-fide layer-two link, occasionally other traffic will be conveyed through the radio link. In our test environment, this is typically broadcast traffic generated outside of the experiment. Similar spikes in all the subsequent tests were observed. There was not enough of it to be problematic so there was no great effort to suppress it.

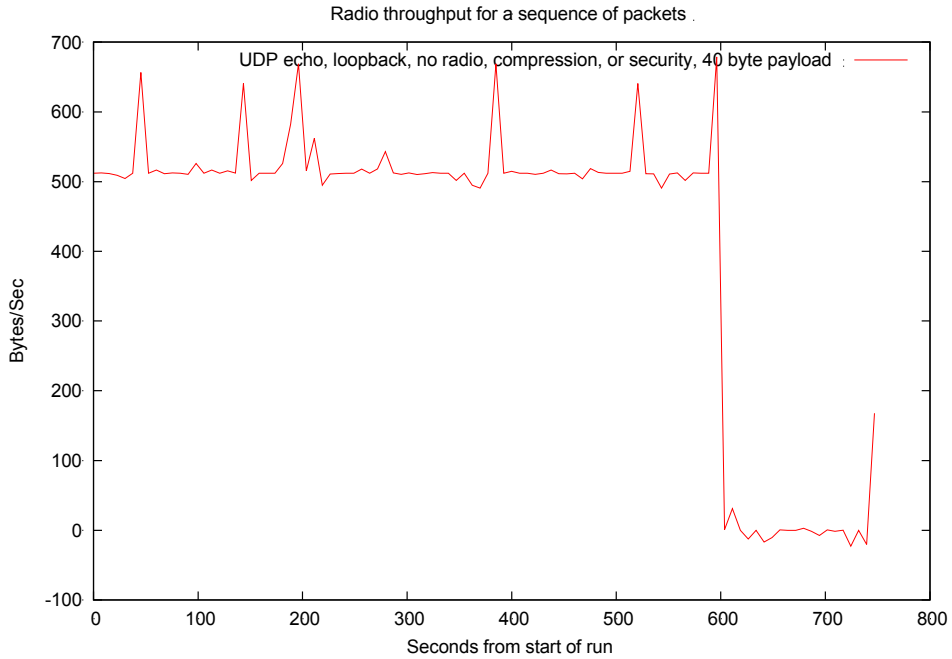

Figure 6.—Radio throughput for a sequence of packets. UDP echo, loopback, no radio, compression or security, 40 byte payload.

Figure 7 and Figure 8 depict traffic over time for ICMP echo (ping) and UDP echo. The jaggedness is an indication of the jitter. Given the scale, the jitter is fairly low. Regarding the initial spike in Figure 7; the ping traffic generator was restarted early on, resulting in a couple invalid readings that are manifested as the spike prior to the 50 sec mark, and should be ignored.

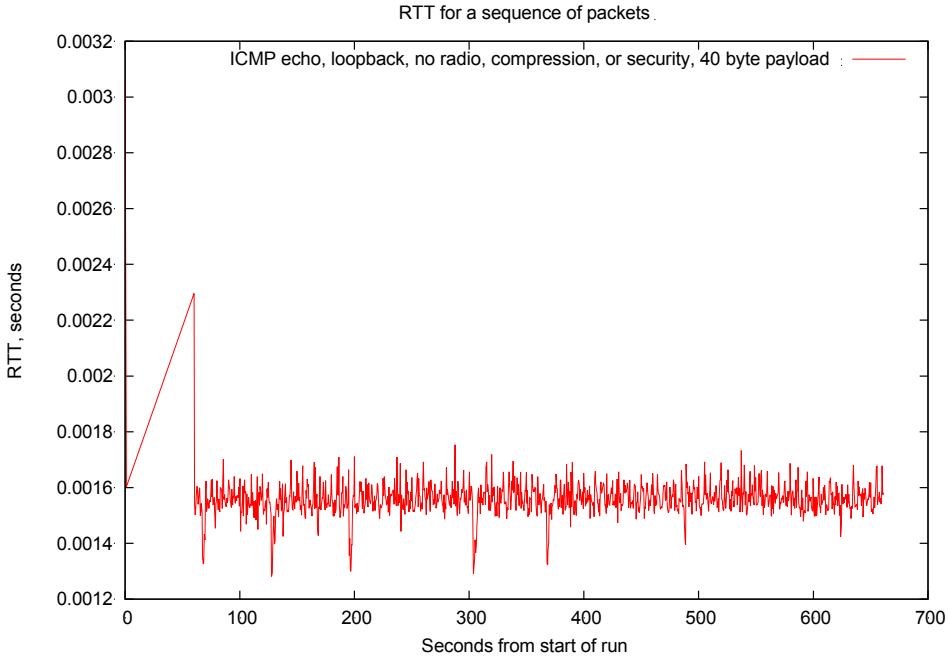

Figure 7.—RTT for a sequence of packets. ICMP echo, loopback, no radio compression or security, 40 byte payload.

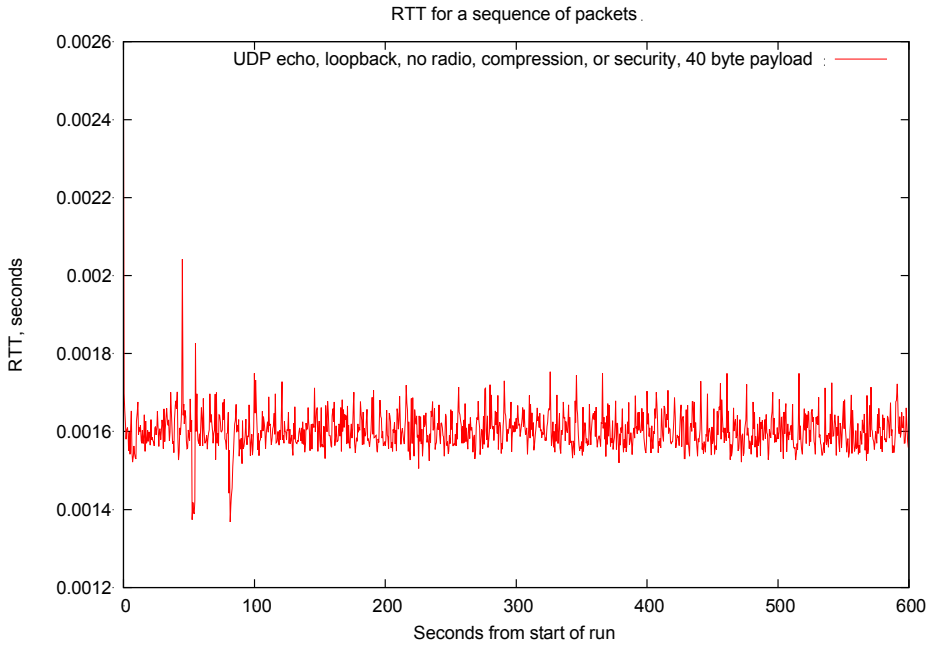

Figure 8.—RTT for a sequence of packets. UDP echo, loopback, no radio, compression, or security, 40 byte payload.

Figure 9 depicts RTT versus time plotted with outliers removed. Outliers were defined as a RTT that indicated a particular packet had to be queued for the next slot.

Figure 10 is a look at a subset of the RTT, the first 30 sec. Particularly during this time, RTT can be affected by additional resource consumption during startup.

For this loopback test there were no radios that were waiting for time slots; so nothing particularly interesting to see. View Figure 16 for instance, for an example of interesting attributes of the radios used for this test.

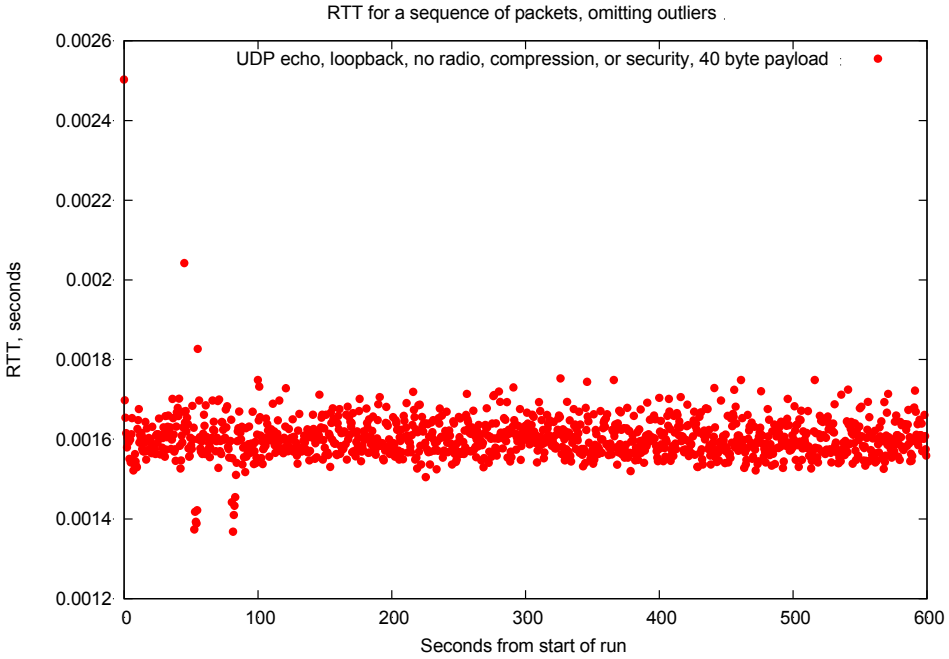

Figure 9.—RTT for a sequence of packets, omitting outliers. UDP echo, loopback, no radio, compression, or security, 40 byte payload.

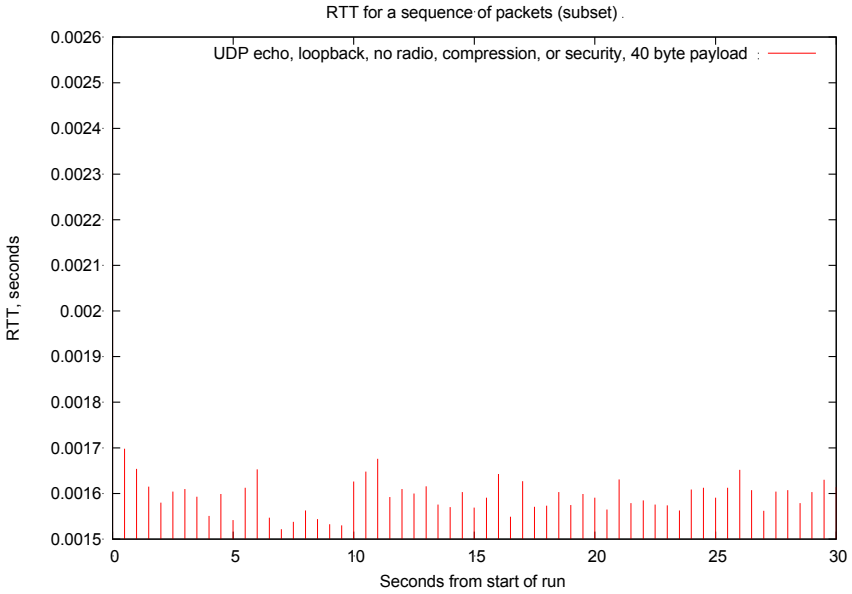

Figure 10.—RTT for a sequence of packets, omitting outliers. UDP echo, loopback, no radio, compression, or security, 40 byte payload.

Finally, RTTs of a subset of the first 30 sec were charted. These were done primarily to examine the effects of ROHC startup. Briefly, in order for ROHC to compress headers, the compressing half at the transmitting radio must convey to the decompressing peer at the other radio the initial state, or initial value of the headers it is "compressing".

Thus, there is always an unavoidable larger-than-normal cost of initializing a link to use ROHC - the first few packets are not only uncompressed, they are indeed larger than an equivalent non-ROHC packet. Given the constrained and close to saturated radio link, this additional cost, seen only at startup, needs to be considered.

This frequency distribution chart conveys a sense of how jittery the traffic is. Jitter can be caused by problems in the network, like congested links. It can also be caused by problems on either of the endpoints (again, constrained resources, like busy CPUs, can cause the problem). Finally, applications can be the source of jitter if they are not carefully designed to control timing. In later charts, what looks to be unusual interaction between one of the traffic generating applications (ping6) and the timing associated with the radio waveform is shown in Figure 14.

In a general sense, jitter can be considered the variability in the latency of packets between two points, two nodes. Jitter can be expressed by plotting a frequency distribution, or by determining the standard deviation of a set of latency measurements.

In an optimal network, this frequency distribution chart would be a single bar, a narrow standard deviation, indicating no jitter. In more problematic configurations, the distribution would be spread out across a range of round trip times.

Figure 11 and Figure 12 plot the frequency distributions for the UDP stream and the ICMP stream. As expected, the two charts are fairly similar.

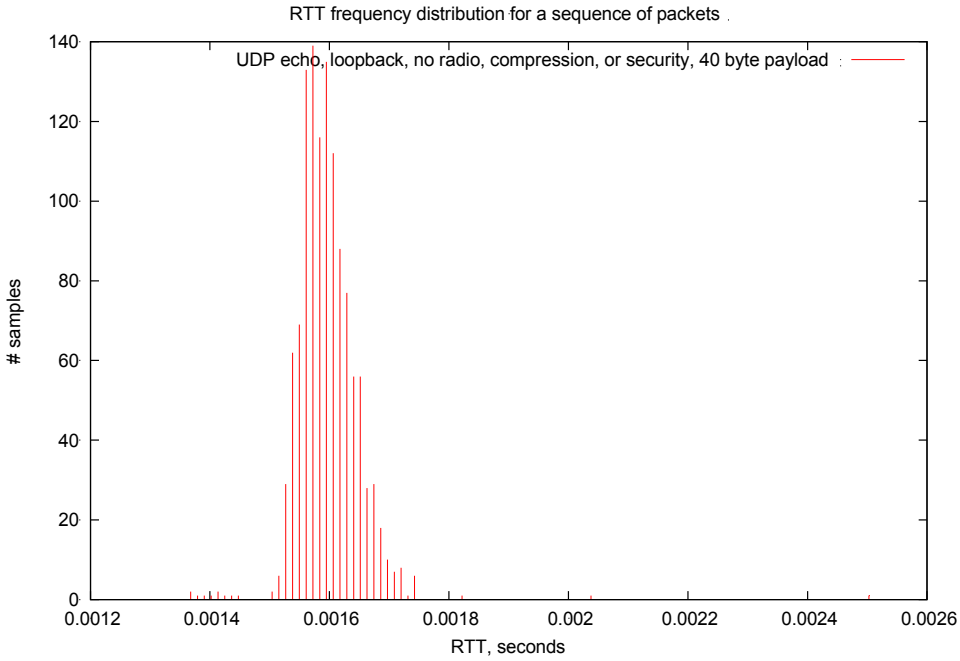

Figure 11.—RTT frequency distribution for a sequence of packets. UDP echo, loopback, no radio, compression, or security, 40 byte payload.

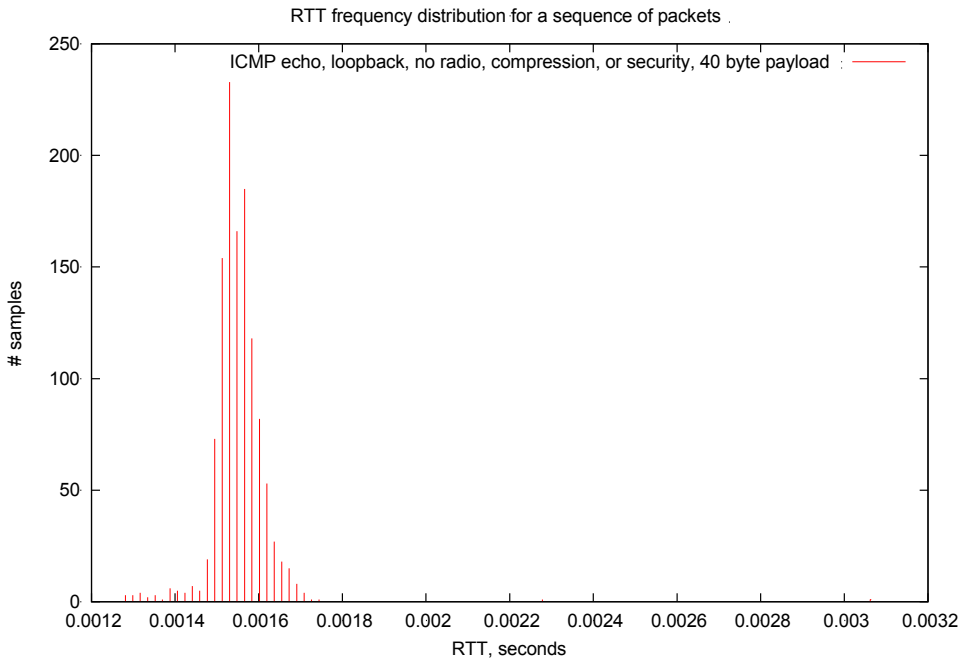

Figure 12.—RTT frequency distribution for a sequence of packets. ICMP echo, loopback, no radio, compression, or security, 40 byte payload.

4.3 Radio—Packet Throughput and Return Time

This configuration was the simplest form of traffic through the radio. No IPsec was used, and no ROHC was used. The expectation was that this test should demonstrate an inefficient use of the radio link, for an equivalent amount of application payload, a relatively large amount of the radio link capacity should be used. Since the payload size was specified, so as to never saturate the link, no packets were dropped. In a real world situation, dropped packets would be inevitable when the radio link becomes saturated. Applications would need to be aware of this and either accept data loss, or go to extraordinarily inefficient means to recover the lost data (bearing in mind, the lengthy ~0.5 sec RTTs, and the fact that the links will be operating close to capacity means recovery is problematic and could cause cascading data loss/recovery problems with other data streams flowing through the link).

The upward spikes in Figure 13 likely represent unexpected additional traffic, as discussed earlier. Compare this to the throughput when ROHC is enabled (Figure 25).

In Figure 14 there's a notable staircase, or ramp, in the trace in the charts of RTTs vs run time for any test that measures ICMP traffic through the radios. The effect is an artifact of the interaction between the ping6 application (the ICMP traffic generator) and the way the radios operate when transmitting packets.

The waveform in the radios has a pair of communicating radios synchronized such that when one is transmitting, its peer is listening. The pair of radios switches back and forth at an interval of 0.1 sec to enable bidirectional communication.

When an ICMP traffic test is run, the standard ping6 application to generate the ICMP traffic is used. This application does not attempt to stay closely synchronized to any particular rate of packet generation—it was designed for debugging network problems, not running tests where precise timing measurement is important.

When ping6 is run, and is configured to generate a packet every 0.5 sec, it comes close but slowly drifts. It tends to often be a bit slower. When pausing between generating packets, ping6 does not account for the time it takes to generate the packet or perform other housekeeping tasks.

The actual time the transmitting radio forwards one of these ICMP packets is influenced by both when a slot is available, and when the packet arrives. Sometimes those packets reach the radio close to when the radio has a slot ready to send something, and sometimes a bit later.

Figure 13.—Radio throughput for a sequence of packets. UDP echo, via radio, no compression, no security, 40 byte payload.

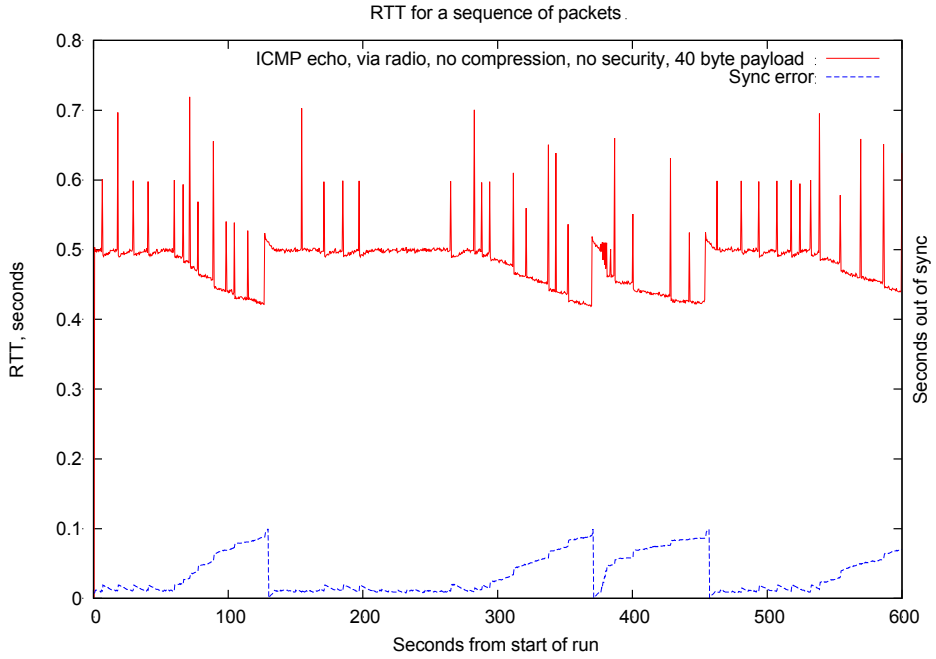

Figure 14.—RTT for a sequence of packets. ICMP echo, via radio, no compression, no security, 40 byte payload.

For all the charts that depict RTTs for ICMP traffic, a plot of ping6 timing error is overlaid on the chart. This plot is the blue trace. That is, the amount of time a ping packet is offset from some correct time (in effect the time used by the radios to determine when each radio can transmit, the time when a slot is available to forward the ICMP packet).

Since measuring radio timing cannot easily be measured, an inference can be made that both the radios and the node generating the ICMP traffic are synchronized to the same time source to a fair degree of accuracy.

Note the error creeps towards the 0.1 sec maximum, then drops to zero. That sudden drop indicates the error has become so large, that it "laps" the next time slot availability, thus becomes (for a while) an error close to zero, using the next slot.

No staircase effect in the UDP traffic charts is shown since the UDP traffic generator takes care to always stay synchronized with the system clock with respect to when it schedules the generation of its traffic. Because the system clock and the radio clocks are synchronized to the same source, the UDP test traffic is effectively synchronized with the timing scheme used by the radios; the UDP traffic always arrives at the same point in slot availability time at the transmitting radio.

Figure 15 shows the same RTT chart, but for the UDP traffic. The UDP traffic generator tries hard to be both precise and accurate. For whatever reason, the peculiar staircase effect does not appear. Figure 15 also shows indications of the transmitting radio enqueuing packets. The majority of the traffic has a 0.45 sec RTT, but occasionally a packet has to wait for the next slot, 0.1 sec away. More rarely, it has to wait two or even three slots. As these are indications of radio link saturation, any RTT larger than 0.45 sec is not wanted.

As depicted in Figure 16, removing the outliers, spreading the remaining data points across a larger range, reveals an interesting pattern. Note the grouping of times at 0.001 sec intervals (e.g., 0.452, 0.453, etc.). This is likely to be related to some sort of process scheduling algorithm on some system involved in the test. 0.001 sec intervals (process scheduler clock "ticks") are often used in operating system process scheduling. Given time and resources, this would be an interesting observation to investigate in greater depth.

This effect does not greatly affect the measured results, but viewing it in this chart helps us understand why the overall timing is not cleaner, less jagged.

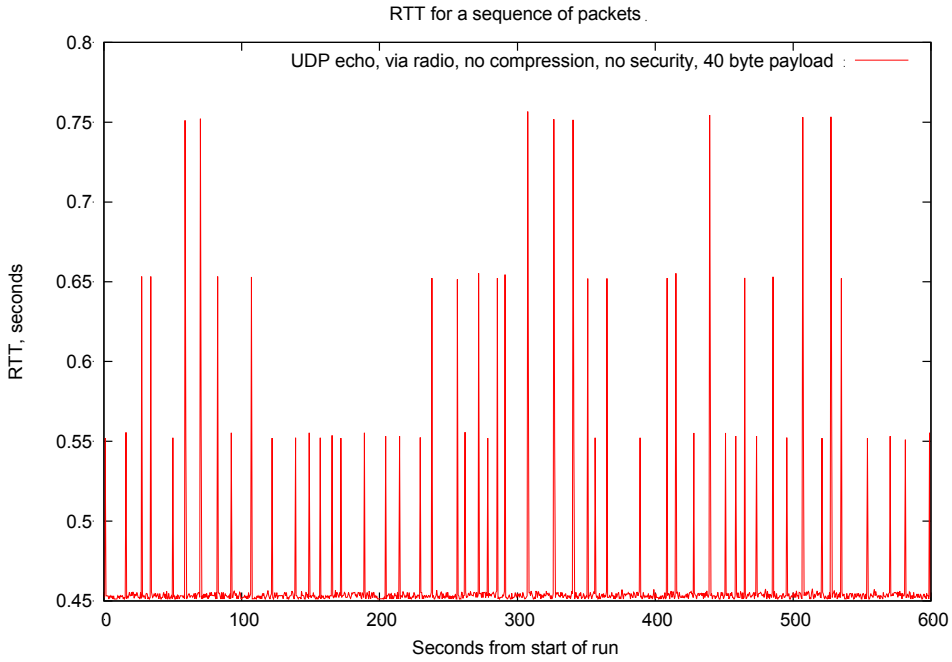

Figure 15.—RTT for a sequence of packets. UDP echo via radio, no compression, no security, 40 byte payload.

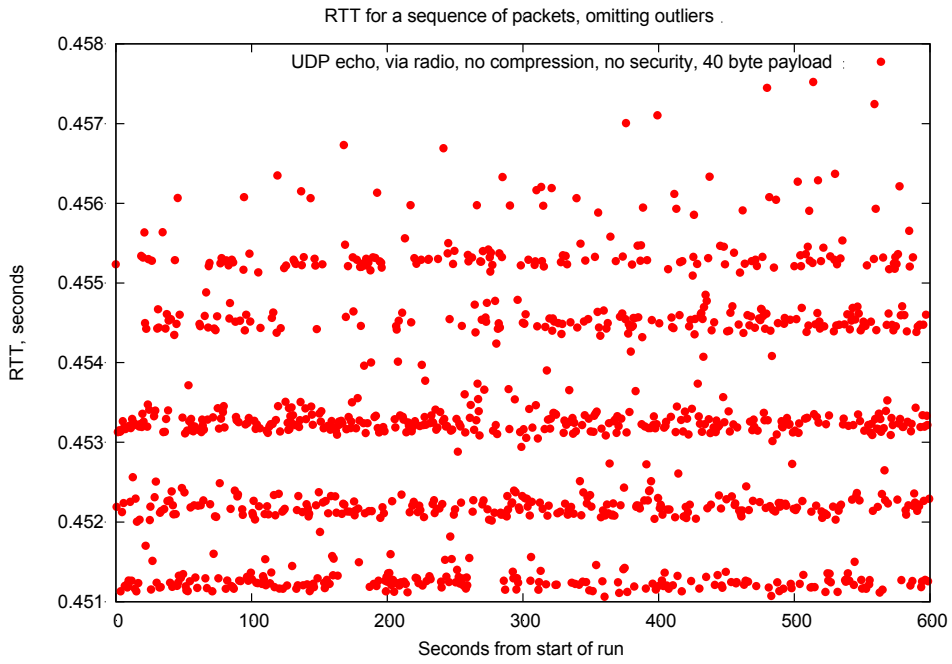

Figure 16.—RTT for a sequence of packets, omitting outliers. UDP echo, via radio, no compression, no security, 40 byte payload.

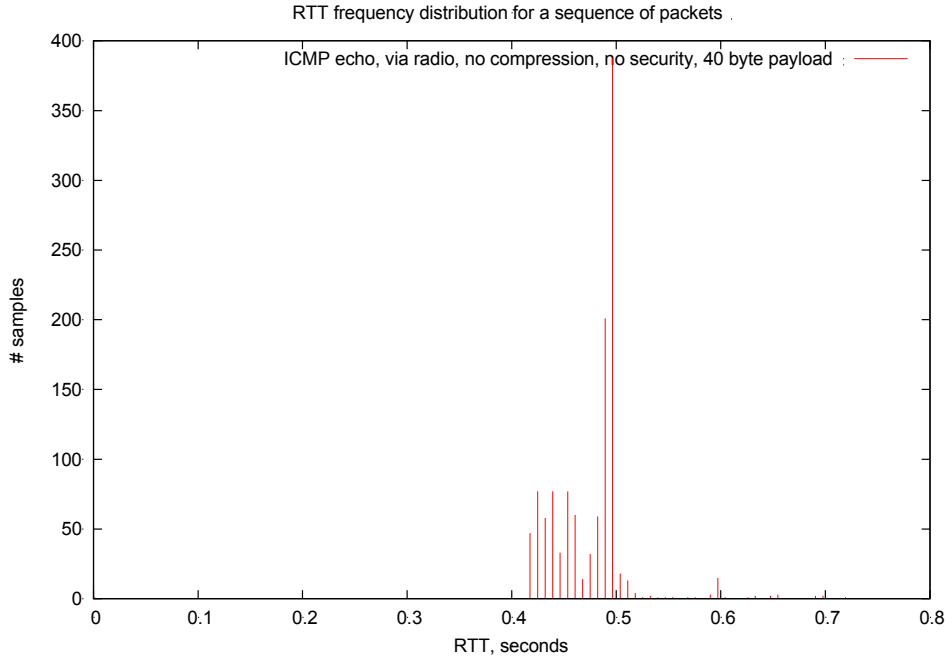

Figure 17.—RTT frequency distribution for a sequence of packets. ICMP echo, via radio, no compression, no security, 40 byte payload.

The ICMP echo frequency distribution in Figure 17 shows a bit of jitter. This might well be an artifact of the staircase/ramp effect seen when running ICMP test traffic.

The UDP echo traffic for the same test configuration has significantly less jitter. There is a small set of outliers at 0.55 and another at 0.65. These are an indication of traffic (packets) that had to be temporarily enqueued at the transmitting radio due to lack of space in the next available radio transmission slot.

The radios have a limited amount of space to buffer packets to be transmitted. So long as that buffer space is not exhausted, the packet will be held and transmitted in the next slot. The slots are spaced at 0.1 sec intervals, which is exactly what the frequency distribution indicates. If that buffer space is exhausted, the radio discards the new packet. There is no flow control (in the sense of TCP flow control), so care must be taken to cause packets not to be dropped, as recovery is problematic (and could cause catastrophic failure).

4.4 Radio, IPsec—Throughput and Return Time

This is the "worst case" configuration. The packets have the greatest amount of non-payload overhead, and no compression is employed to help keep the packets to a manageable size as they are sent through the radio link.

In this configuration, IPsec is enabled. This adds additional overhead to the IP packets that ultimately transit the radio link, in that an IPsec extension header is also included (in the overhead).

As with Figure 13, the throughput is fairly constant, with occasional spikes, likely due to aforementioned broadcast traffic. The throughput has gone up to around 0.31 sec, which indicates inefficient use of the link capacity.

Figure 18 indicates little deviation.

Figure 19 depicts the throughput, which was mostly consistent, with a few peaks.

As seen before, the peculiar staircase pattern (Figure 20).

And once again, in Figure 21, an indication of how most of the UDP traffic managed to be sent directly, but some packets had to wait multiples of 0.1 sec for one, two, three, and sometimes even four radio waveform slots.

Once again, that quantization due to process schedule timing on some subsystem (Figure 22).

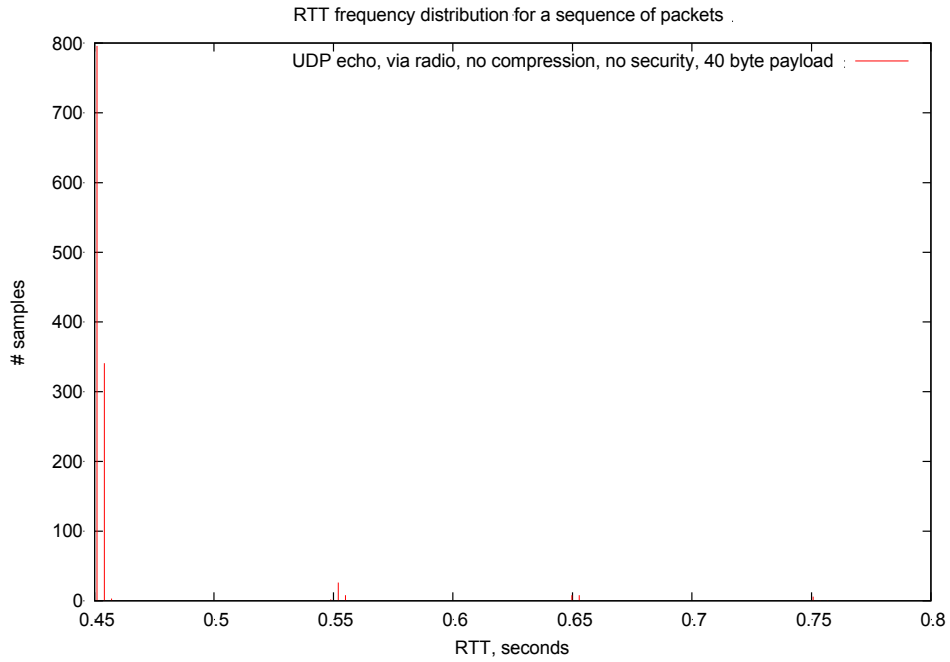

Figure 18.—RTT frequency distribution for a sequence of packets. UDP echo, via radio, no compression, no security, 40 byte payload.

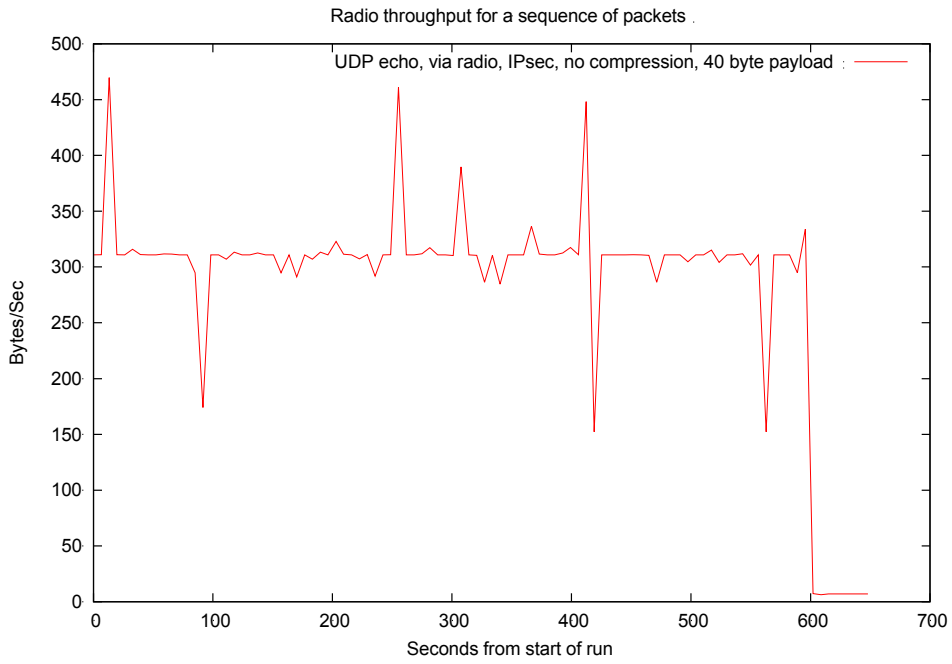

Figure 19.—Radio throughput for a sequence of packets. UDP echo, via radio, IPsec, no compression, 40 byte payload.

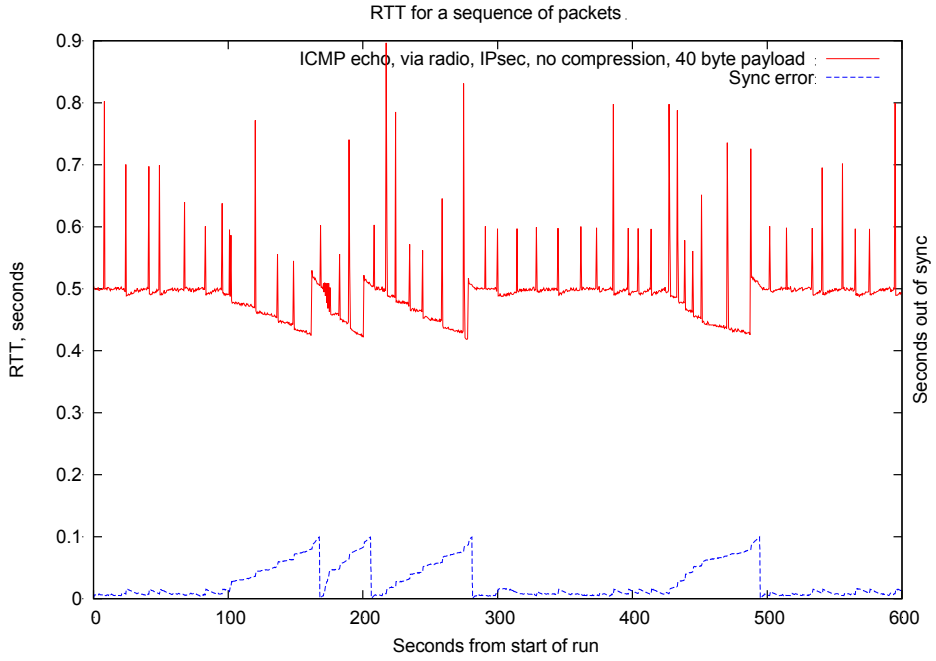

Figure 20.—RTT for a sequence of packets. ICMP echo, via radio, IPsec, no compression, 40 byte payload and sync error.

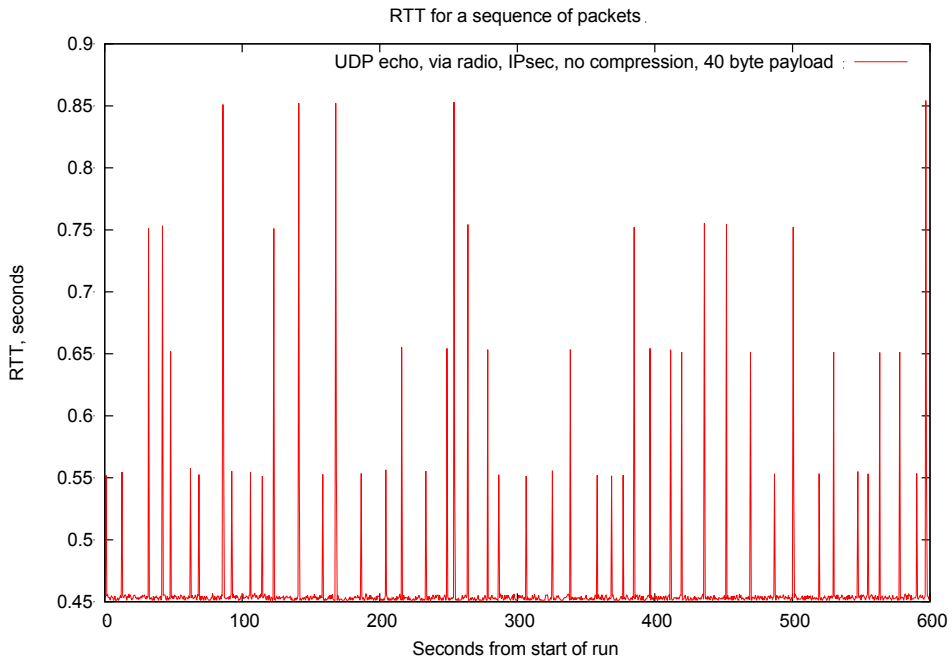

Figure 21.—RTT for a sequence of packets. UDP echo, via radio, IPsec, no compression, 40 byte payload.

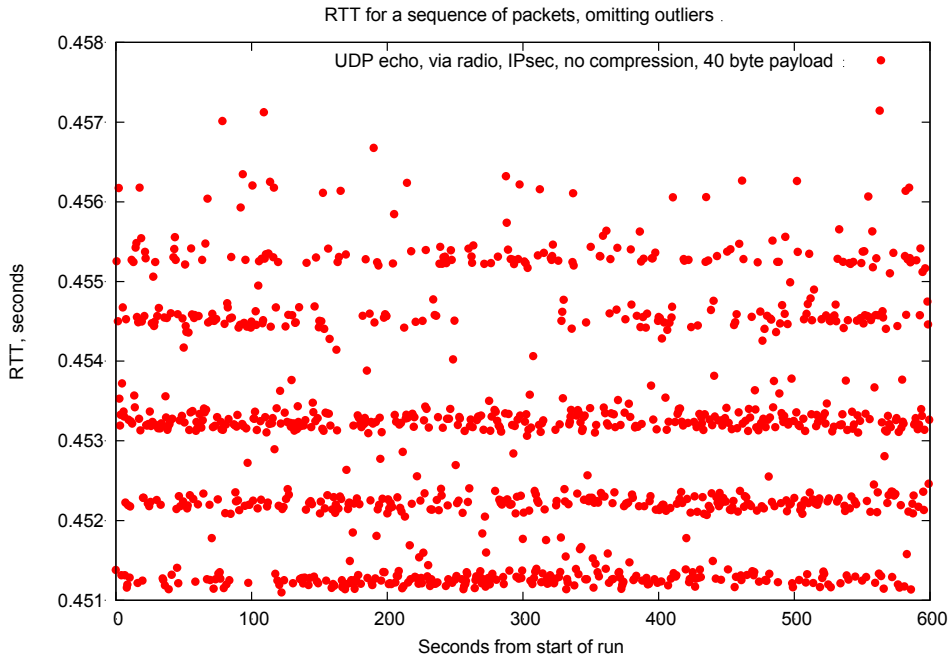

Figure 22.—RTT for a sequence of packets, omitting outliers. UDP echo, via radio, IPsec, no compression, 40 byte payload.

The ICMP frequency distribution of RTTs, Figure 23, is somewhat spread out. As noted, this is probably due to the impreciseness of timing in the ping6 traffic generating application.

As seen in previous tests, there is little jitter in the UDP traffic, excepting the packets that had to be enqueued and sent in a later slot.

Very little deviation, indicated in Figure 24.

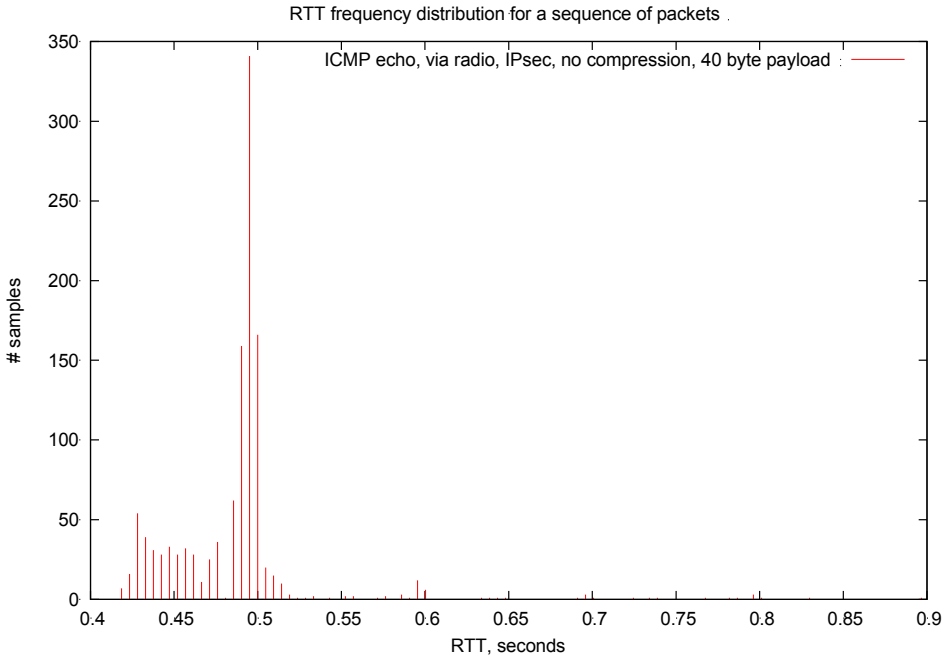

Figure 23.—RTT frequency distribution for a sequence of packets. ICMP echo, via radio, IPsec, no compression, 40 byte payload.

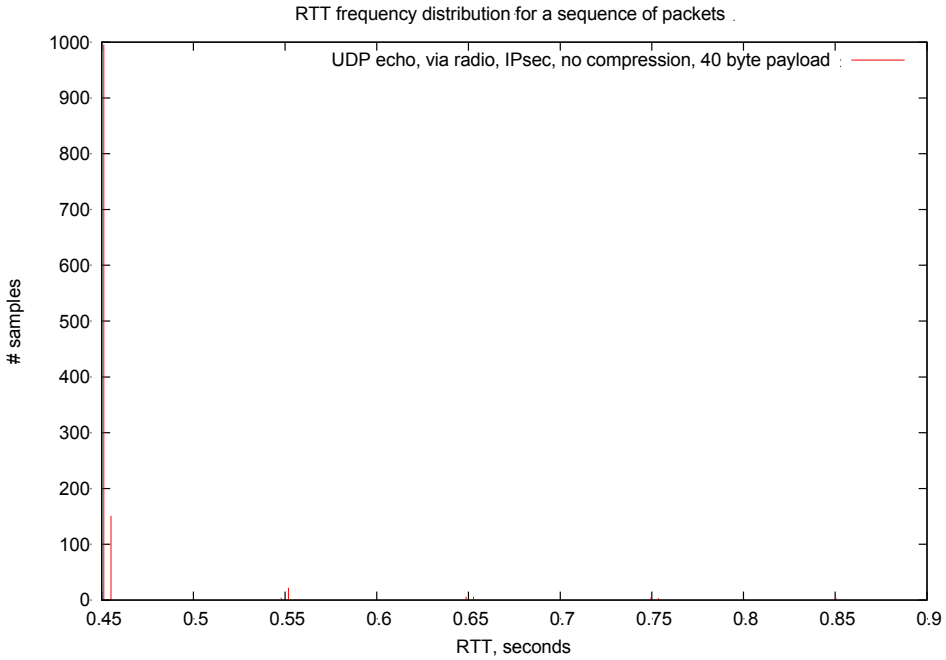

Figure 24.—RTT frequency distribution for a sequence of packets. UDP echo, via radio, IPsec, no compression, 40 byte payload.

4.5 Radio, ROHC—Packet Throughput and Return Time

This configuration tests ROHC by itself, exclusive of the IPsec overhead. The ROHC compressor should still compress the available non-IPsec IP header (as well as the UDP header or ICMP header (depending on the test). So, some gain over non-compressed tests will be seen.

Compared to the baseline test of a throughput calculation around 260 bytes/sec, this test indicates an increase of a throughput calculation to around 100 bytes/sec (Figure 25). This is a significant saving in link utilization. It is also interesting to note that the jaggedness of the throughput is virtually eliminated - indicating that there was always sufficient capacity to forward the occasional extraneous outside traffic, like routing updates.

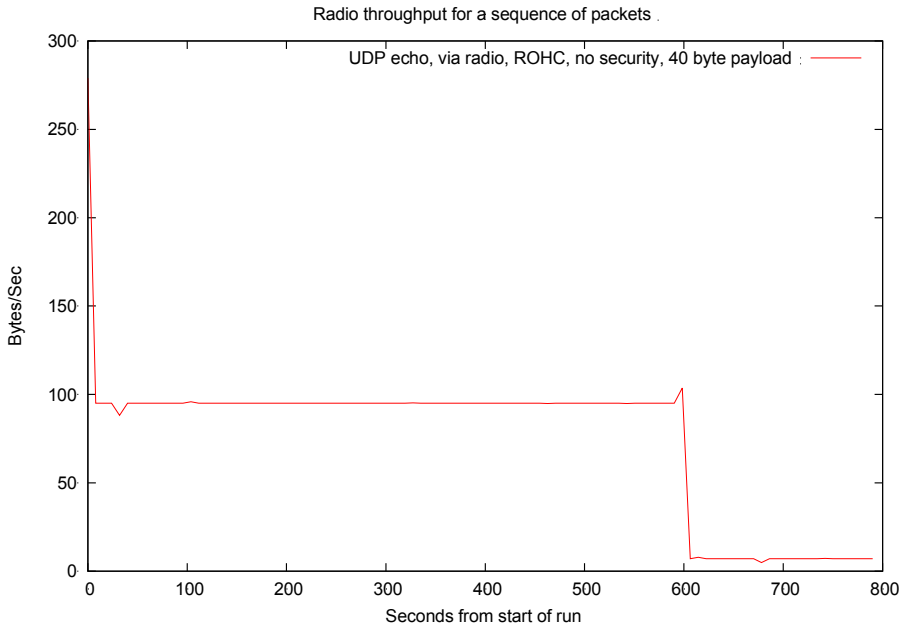

Figure 25.—Radio throughtput for a sequence of packets. UDP echo, via radio, ROHC, no security, 40 byte payload.

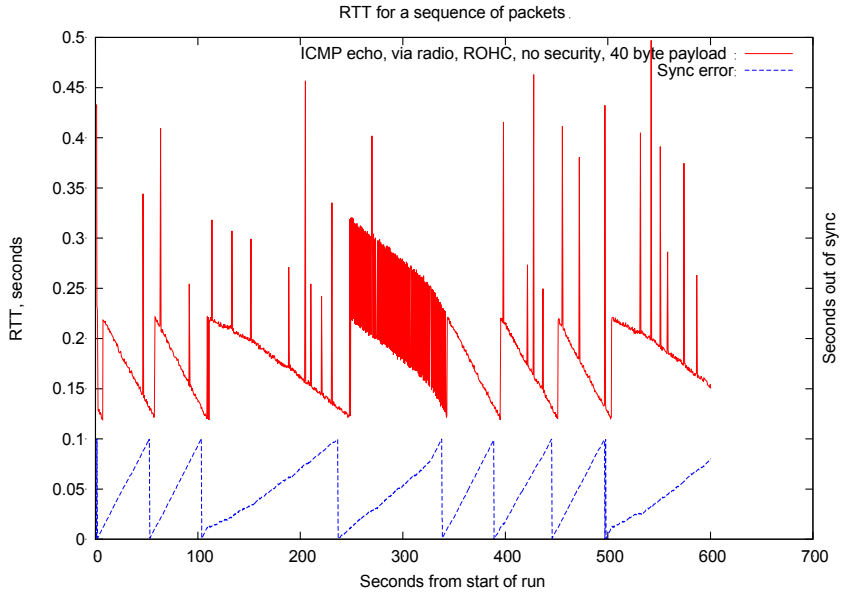

Figure 26.—RTT for a sequence of packets. ICMP echo, via radio, ROHC, no security, 40 byte payload and sync error.

As depicted in Figure 26, red trace, the staircase/ramp effect becomes more dramatic as the size of the packets (including all that additional overhead devoted to packet headers) that are sent between the radios increase.

Another peculiar effect (Figure 26, blue trace) is seen starting around 250 sec into the run, where the RTT delay oscillates between a two values of about 0.1 sec difference for every other packet transmitted. This could be an interaction with a system clock on the radios or the end points. For any of these nodes, the system clocks are likely to be running at multiples of the 10 Hz rate of the radio transmit slot timing, and might help "amplify" some tendency in the system to resonate under certain circumstances.

Figure 27 and Figure 28 depict the same measurements as performed in the previous test, as measured in this test.

The previously observed clustering of timing around 0.001 sec quanta is no longer so clearly defined and seems to be affected by some additional factor. Again, this would be interesting to investigate further, given time and resources.

Figure 29 is a subset of the first 30 sec of UDP traffic. It clearly demonstrates the relatively high startup cost of using ROHC. Note the first four packets have a much higher RTT. Then, after the ROHC compressor has achieved Second Order State, where headers are being optimally compressed, RTT assumes a much more desirable value. Figure 30 shows a RTT frequency distribution of a selected sequence of packets.

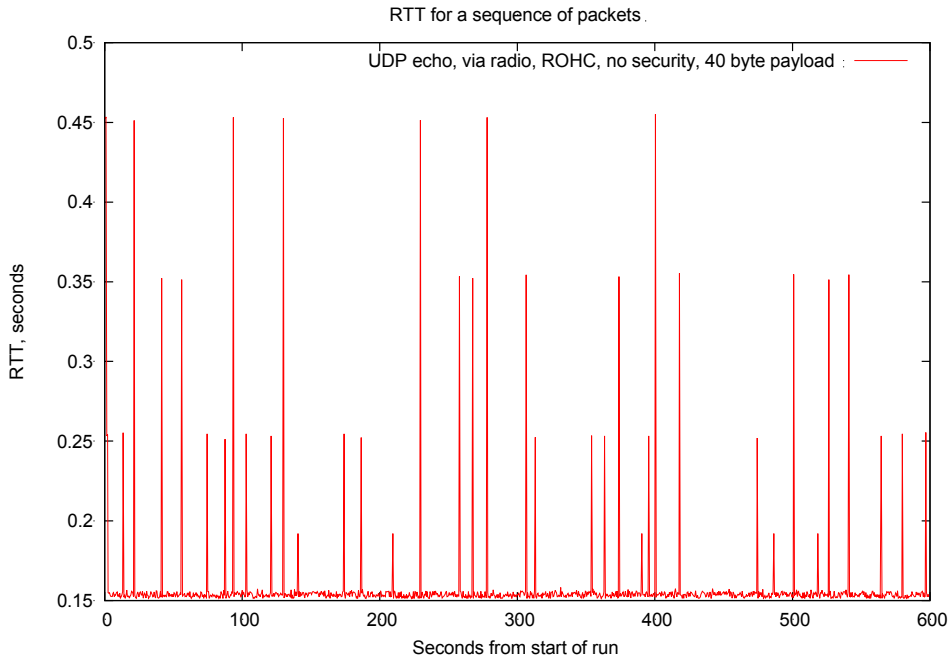

Figure 27.—RTT for a sequence of packets. UDP echo, via radio, ROHC, no security, 40 byte payload.

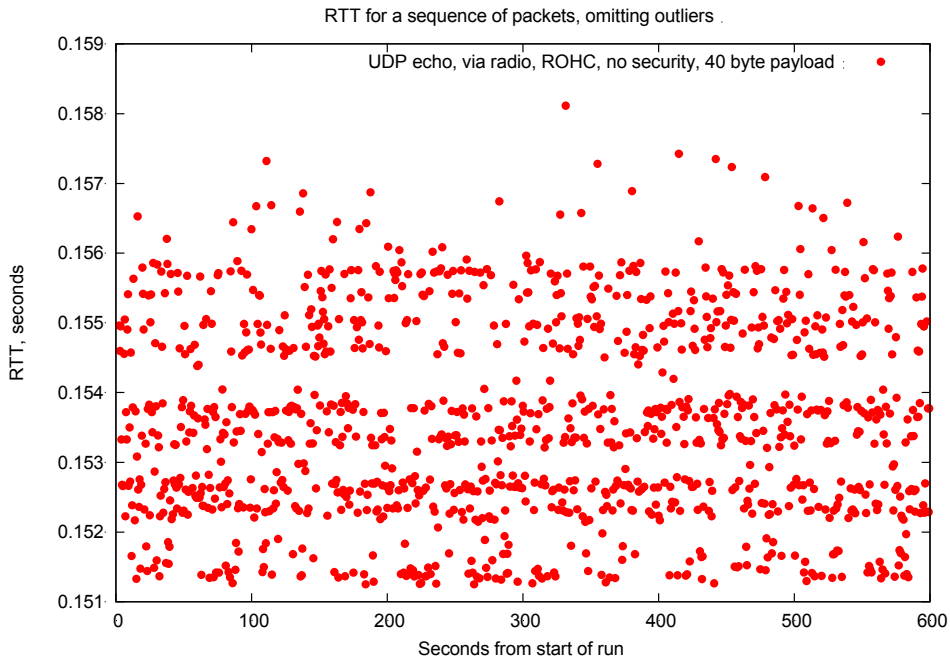

Figure 28.—RTT for a sequence of packets, omitting outliers. UDP echo, via radio, ROHC, no security, 40 byte payload.

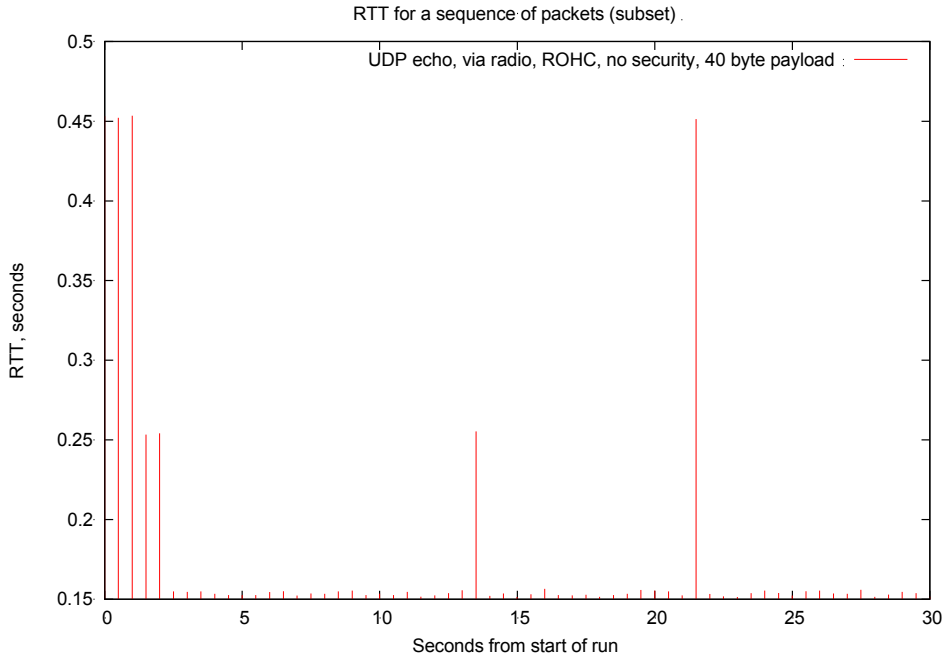

Figure 29.—RTT for a sequence of packets (subset). UDP echo, via radio, ROHC, no security, 40 bytre payload.

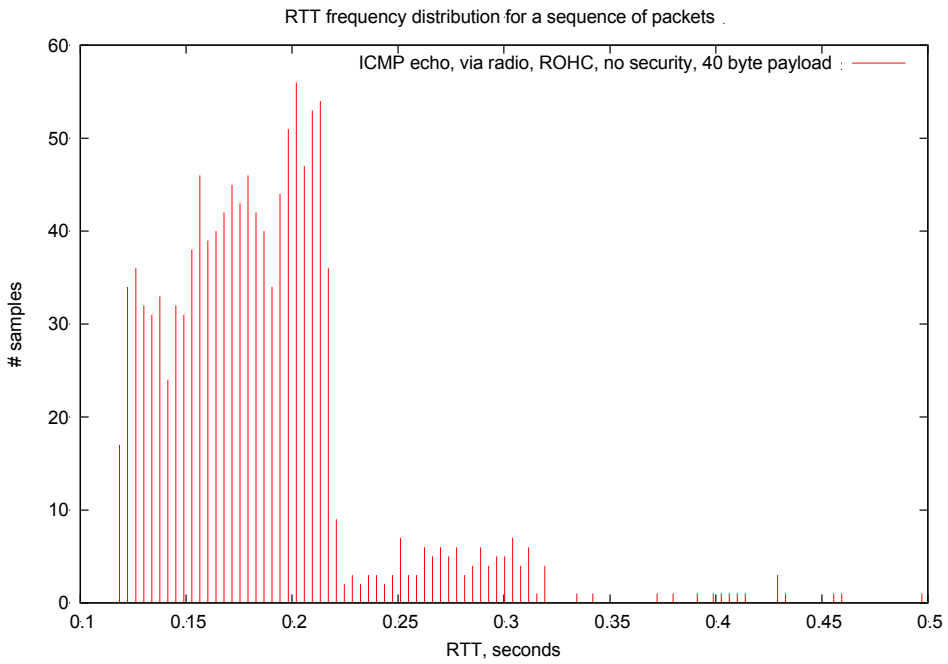

Figure 30.—RTT frequency distribution for a sequence of packets. ICMP echo, via radio, ROHC, no security, 40 byte payload.

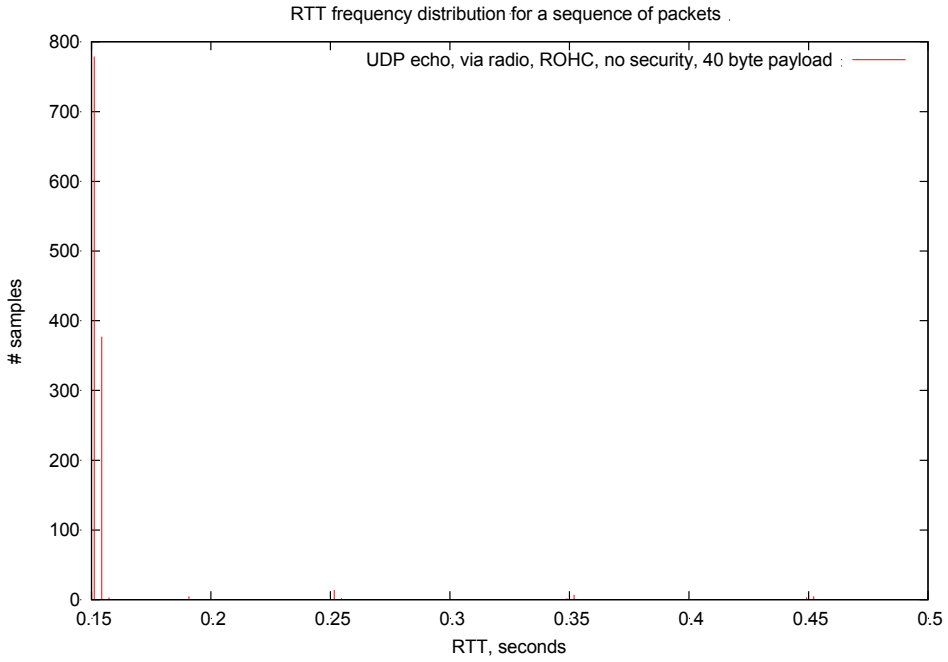

Figure 31.—RTT frequency distribution for a sequence of packets. UDP echo, via radio, ROHC, no security, 40 byte payload.

The standard deviation for ICMP RTTs (Figure 31) is broad and irregular. Ideally, the deviation would be small. This again may be an artifact of the imprecise timing of packet generation via *ping6*.

For UDP test traffic, the frequency distribution is concentrated almost entirely at 0.15 sec RTT, with a couple small outliers at 0.25, 0.35, and 0.45. This reiterates the above observation.

4.6 Radio, IPsec, ROHC—Packet Throughput and Return Time

In this test, all options are enabled. In addition, header compression, which should mediate that consumption, is also turned on. The results of this test are an indication that mobile routing and IPsec can be used to address UAS CNPC requirements and still have a usable link (i.e., non-saturated).

As depicted in Figure 32 the throughput is lower than worst case. RTT has moved from ~260 bytes/sec to around 220 bytes/sec, the lower number indicating more optimal use of the link.

Figure 33 and Figure 34 are the round trip times measured for ICMP and UDP echo traffic, respectively. The ICMP traffic has the usual ramp artifact as seen before. In both cases, occasional "outliers" that are an indication that the radio had to buffer and hold outgoing traffic pending the next transmission slot is shown.

Compared to similar charts for the other tests, there are notably fewer outliers, fewer instances of queued traffic being held up for the next available slot.

Time permitting, it would be instructive to focus on some instances of these outliers and closely examine the traffic prior to the outlier; see if there is unusual traffic (like extraneous broadcasts) being sent through the interface.

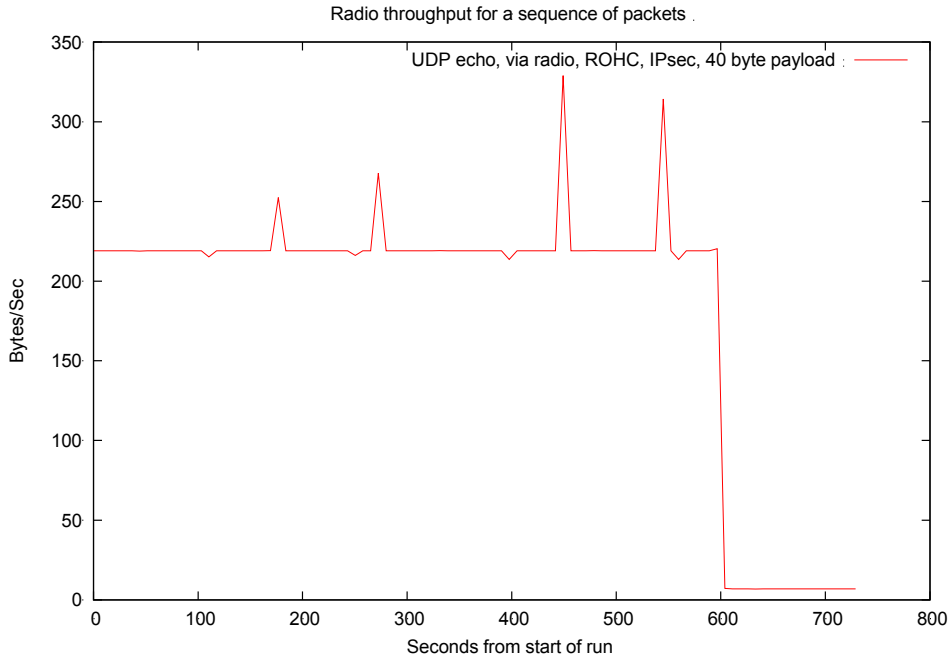

Figure 32.—Radio throughtput for a sequence of packets. UDP echo, via radio, ROHC, IPsec, 40 byte payload.

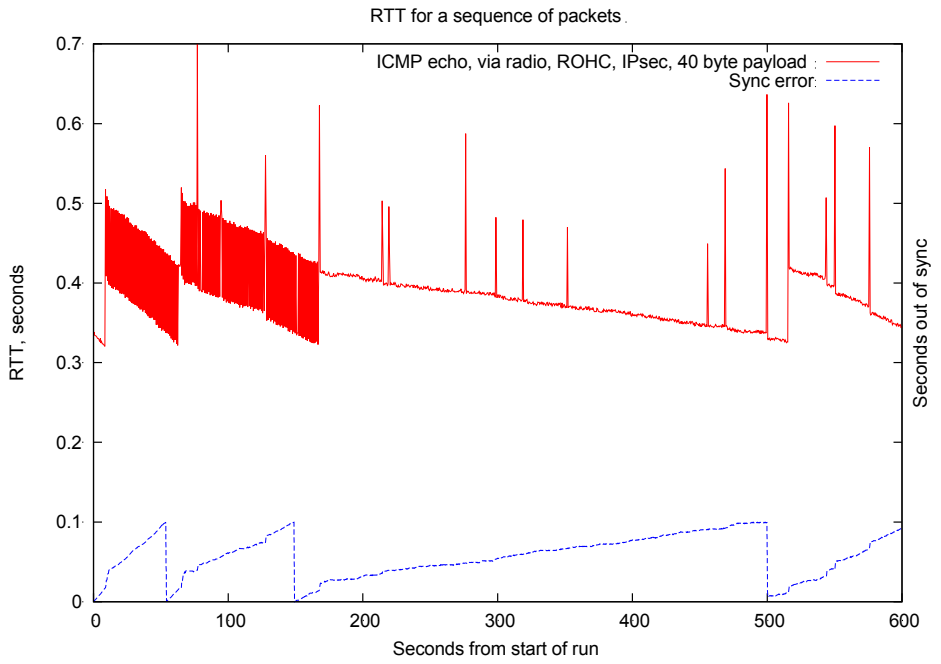

Figure 33.—RTT for a sequence of packets. IMP echo, via radio, ROHC, IPsec, 40 byte payload.

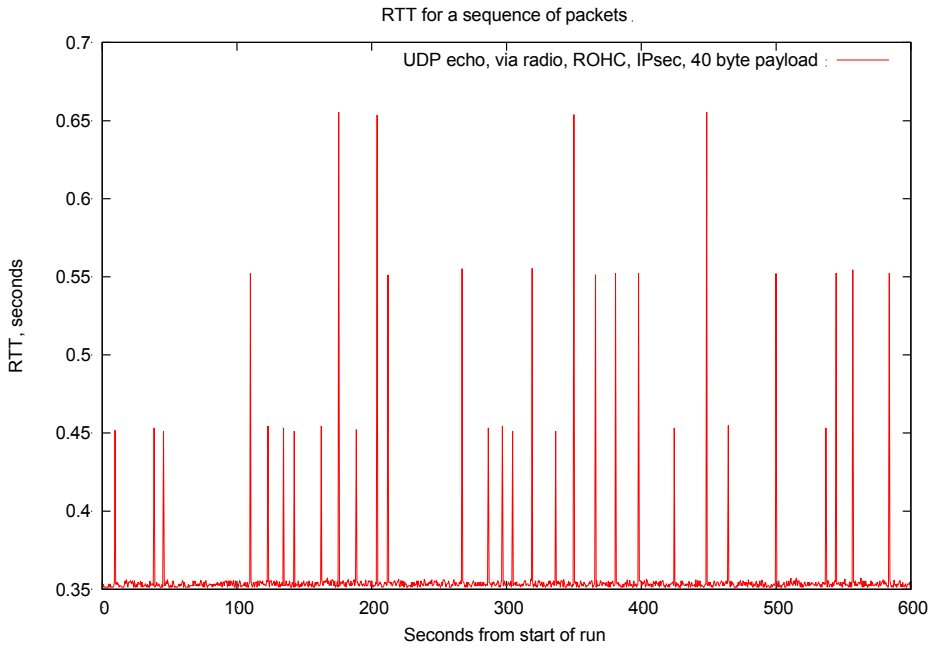

Figure 34.—RTT for a sequence of packets. UDP echo, via radio, ROHC, IPsec, 40 byte payload.

Figure 35 depicts another instance of that peculiar process scheduler quantization. In Figure 36, the expected startup cost are not seen, which would show as longer RTTs for the first few packets. More time would be required to investigate its absence.

Figure 37 and Figure 38 depict the frequency distribution for ICMP echo and UDP echo, respectively. There's significant deviation in the ICMP traffic and a significant lack of deviation in the UDP traffic. Regarding the tight deviation observed with the UDP traffic test, perhaps for links whose tendency to become congested is well controlled; exploiting time synchronization with that of the radio (N.B. for radios with the characteristics of those used in this test) is beneficial.

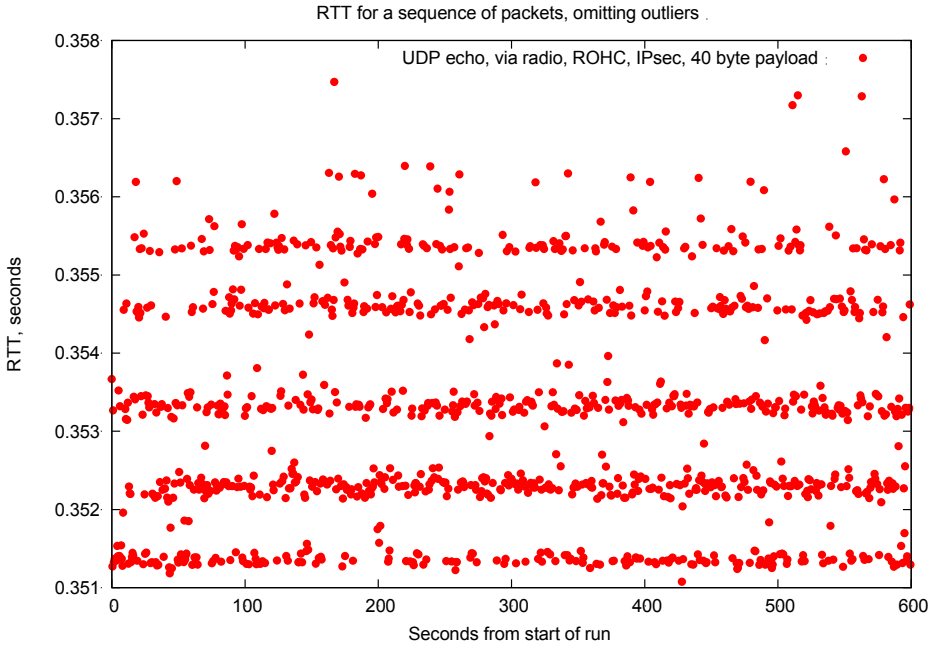

Figure 35.—RTT for a sequence of packets, omitting outliers. UDP echo, via radio, ROHC, IPsec, 40 byte payload.

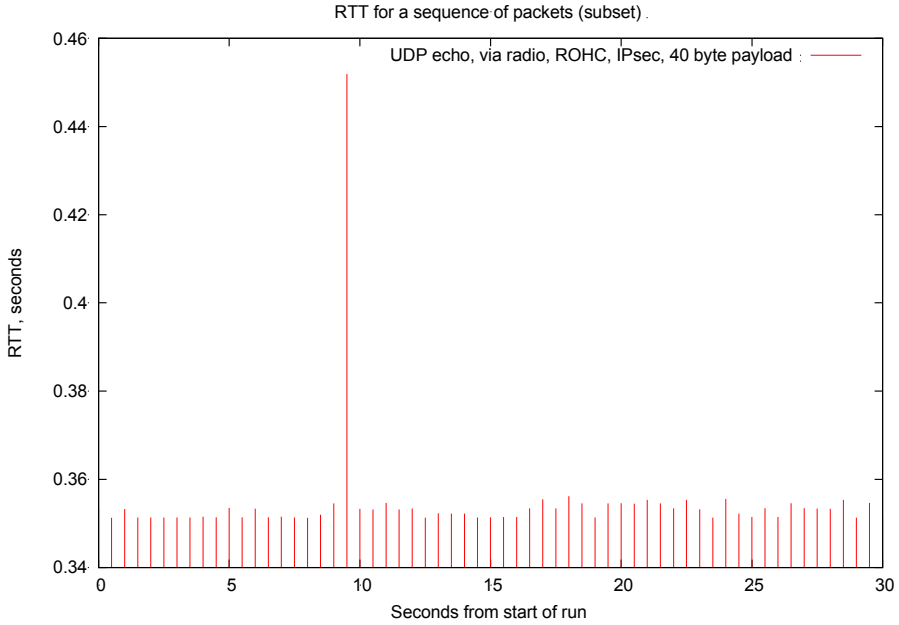

Figure 36.—RTT for a sequence of packets (subset). UDP echo, via radio, ROHC, IPsec, 40 byte payload.

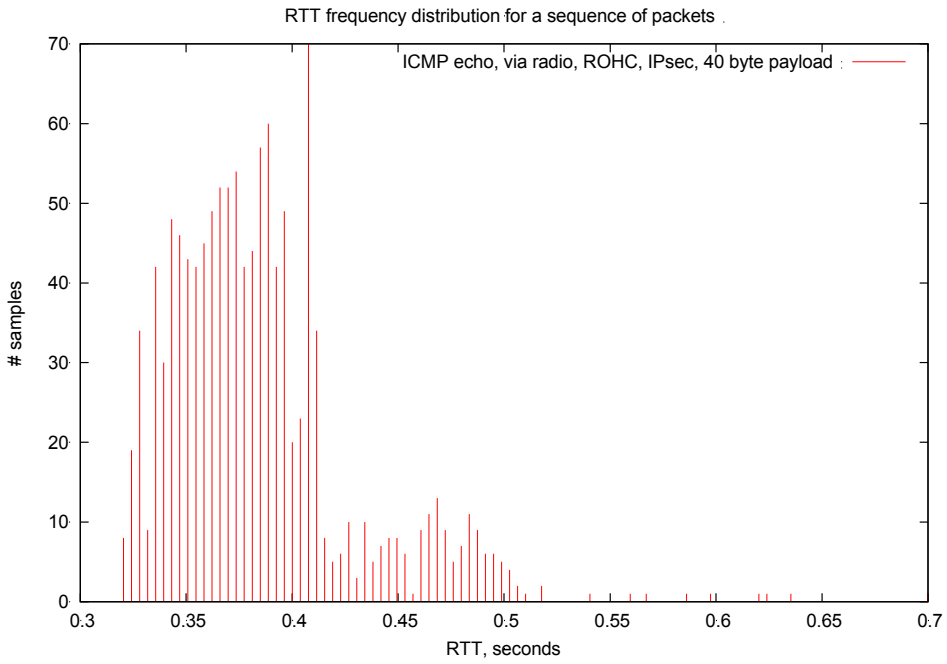

Figure 37.—RTT frequency distribution for a sequence of packets. ICMP echo, via radio, ROHC, IPsec, 40 byte payload.

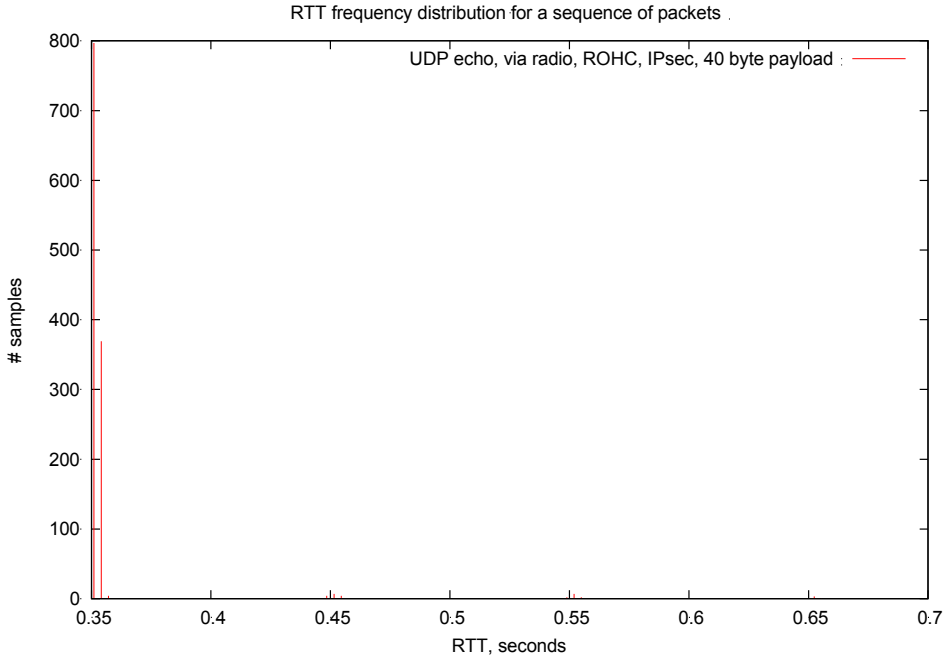

Figure 38.—RTT frequency distribution for a sequence of packets. UDP echo, via radio, ROHC, IPsec, 40 byte payload.

Less deviation is better, but if the cause is tight timing of the source of traffic, this might not be an accurate reflection of real-world use. Or it might be a caveat that in real world use, tight timing is desirable. We'll reiterate the point that to some extent, the observed effects are artifacts of the radios used in this test, and might not reflect experiences using other types of radio equipment.

5.0 Conclusion

Testing the transmission of a 40 byte UDP payload in our CNPC test network using the four different testing configurations of:

1. Baseline Payload
2. Baseline Payload + IPsec Only
3. Baseline + ROHC Only
4. Baseline Payload + ROHC + IPsec

The testing provided a characterization of how the CNPC test network and CNPC RF radio's performed in each test configuration. It was determined that ROHC is an excellent technique to reduce packet size. It is particularly useful when sending information over an RF link that has bandwidth constraints.
The analysis showed that due to the increase in packet size with the addition of IPsec, the baseline ping packet increased by 18.7 percent or 24 bytes and had a reduction in throughput of 18.3 percent.

With the addition of ROHC to the ESP encapsulated IP packet, the packet structure was reduced by 30.2 percent or 8 bytes and had an increase in throughput of 29.6 percent.

Adding confidentiality to our setup will have little impact on the results as the ROHC header compression was not able to successfully compress the inner IPv6 header as hoped for when using the null encryption cipher. So even if AES is enabled, the exact same overhead results should be observed i.e., the ROHC library will compress the first IPv6 header, the ESP header, but not the inner IPv6 header, the UDP header, the payload, or the ESP trailer including the ICV.

Table 2 depicts the measured parameters for all four test configurations.

The round trip times for ICMP and UDP traffic are close. Lower numbers (faster round trip times) are better. The UDP measurements tended to be slightly faster, likely because the generation of that traffic is always closely synchronized to the timing used by the radios in determining when transmit slots are available. The minimum and maximum times are a coarse way of understanding the jitter. The standard deviation is a better measure—the lower the number, the better, the more likely the link is being used efficiently.

Finally, in our test, average throughput was used as a direct indication of link efficiency. By keeping the payload size constant, when the amount of bytes were increased by the addition of IPsec overhead, the average throughput number increased indicating a less efficient link.

Regarding ROHC, a condition of how well it handles lossy channels was not tested. ROHC itself does nothing to handle errors or lost packets. It will, however, eventually detect when the ROHC decompressor is out of sync with the compressor and initiate a costly resynchronization (similar to the initial startup cost). This will be explored during flight test.

TABLE 2.—RESULTS TABLE

	Loopback	Radio, Baseline	Radio, IPsec	Radio, ROHC	Radio, IPsec, ROHC
UDP, avg. thruput, B/s	N/A	262.895	311.147	95.4017	219.03
ICMP RTT avg., sec	0.0015619	0.479911	0.490045	0.187938	0.387468
UDP RTT avg., sec	0.0016029	0.460836	0.461066	0.159071	0.357333
UDP RTT min., sec	0.001368	0.451063	0.451097	0.151249	0.351076
UDP RTT max., sec	0.002503	0.756798	0.85444	0.455316	0.655391
ICMP std. dev.	7.25e-05	0.0399578	0.0454718	0.0491613	0.0447224
UDP std. dev.	5.44e-05	0.0370915	0.0428625	0.0335293	0.0278838

Future work will involve integrating IPsec into the relevant flight environment and testing this system aboard the NASA S3 and NASA T34 aircraft in a realistic RF environment to see how the ROHC reacts and affects performance. Additionally, work efforts to get UMIP and Strongswan IKEv2 working together and evaluate the use of IPsec ESP transport mode from the PIC system to the mobile node rather than using tunnel mode, as was the case in this experiment will be performed. This will separate UMIP from Strongswan and UMIP will no longer manage the IPsec connection and revert to simply using IPv6-in-IPv6 tunnels (which are very compressible with ROHC) and IPsec ESP transport mode for end-to-end security. Using IPsec ESP transport mode encapsulation could show an additional reduction in packet size. This would increase the amount of compression that can be achieved from ROHC. Additional work with Rockwell Collins to add WiMAX-style datalink security to our CNPC system (PKMv2, EAP-TLS authentication, TEK encryption, etc.) is planned.

Appendix A.—Packet Analysis

Summary to send 40 bytes OTA (not counting radio headers):

Baseline: 128 bytes (88 bytes overhead)
Baseline+IPsec: 152 bytes (112 bytes overhead)
Baseline+ROHC: 44 bytes (4 bytes overhead)
Baseline+ROHC+IPsec: 106 bytes (66 bytes overhead)

Example Packet Flow without ROHC and without IPsec ESP

CN (eth1):

18:54:22.000118 IP6 (hlim 64, next-header UDP (17) payload length: 48) 2001:470:e23d:e012::12.44081 > 2001:470:e23d:f601::1.7: [udp sum ok] UDP, length 40

```
0x0000:  6000 0000 0030 1140 2001 0470 e23d e012  `....0.@...p.=..
0x0010:  0000 0000 0000 0012 2001 0470 e23d f601  ..........p.=..
0x0020:  0000 0000 0000 0001 ac31 0007 0030 6eef  .........1...0n.
0x0030:  0000 00e1 0000 0000 0000 0000 0000 0000  ................
0x0040:  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x0050:  0000 0000 0000 0000                      ........
```

18:54:22.452216 IP6 (hlim 63, next-header UDP (17) payload length: 48) 2001:470:e23d:e000::601.7 > 2001:470:e23d:e012::12.44081: [udp sum ok] UDP, length 40

```
0x0000:  6000 0000 0030 113f 2001 0470 e23d e000  `....0.?...p.=..
0x0010:  0000 0000 0000 0601 2001 0470 e23d e012  ..........p.=..
0x0020:  0000 0000 0000 0012 0007 ac31 0030 7ef0  ..........1.0~.
0x0030:  0000 00e1 0000 0000 0000 0000 0000 0000  ................
0x0040:  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x0050:  0000 0000 0000 0000                      ........
```

Blue = IPv6 Header (40 bytes)
Red = UDP Header (8 bytes)
Green = Payload (40 bytes)

HA (six0):

18:54:22.000138 IP6 (hlim 64, next-header IPv6 (41) payload length: 88) 2001:470:e23d:e000::1000 > 2001:470:e23d:190d:601:12c1:6744:52d3: IP6 (hlim 63, next-header UDP (17) payload length: 48) 2001:470:e23d:e012::12.44081 > 2001:470:e23d:f601::1.7: [udp sum ok] UDP, length 40

```
0x0000:  6000 0000 0058 2940 2001 0470 e23d e000  `....X)@...p.=..
0x0010:  0000 0000 0000 1000 2001 0470 e23d 190d  ..........p.=..
0x0020:  0601 12c1 6744 52d3 6000 0000 0030 113f  ....gDR.`....0.?
0x0030:  2001 0470 e23d e012 0000 0000 0000 0012  ...p.=..........
0x0040:  2001 0470 e23d f601 0000 0000 0000 0001  ...p.=..........
0x0050:  ac31 0007 0030 6eef 0000 00e1 0000 0000  .1...0n.........
0x0060:  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x0070:  0000 0000 0000 0000 0000 0000 0000 0000  ................
```

18:54:22.451977 IP6 (hlim 63, next-header IPv6 (41) payload length: 88)
2001:470:e23d:190d:601:12c1:6744:52d3 > 2001:470:e23d:e000::1000: IP6 (hlim 64, next-header UDP (17) payload length: 48) 2001:470:e23d:e000::601.7 > 2001:470:e23d:e012::12.44081: [udp sum ok] UDP, length 40

```
0x0000:  6000 0000 0058 293f 2001 0470 e23d 190d  `....X)?...p.=..
0x0010:  0601 12c1 6744 52d3 2001 0470 e23d e000  ....gDR....p.=..
0x0020:  0000 0000 0000 1000 6000 0000 0030 1140  ........`....0.@
0x0030:  2001 0470 e23d e000 0000 0000 0000 0601  ...p.=.........
0x0040:  2001 0470 e23d e012 0000 0000 0000 0012  ...p.=.........
0x0050:  0007 ac31 0030 7ef0 0000 00e1 0000 0000  ...1.0~.........
0x0060:  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x0070:  0000 0000 0000 0000 0000 0000 0000 0000  ................
```

Purple = Outer IPv6 Header (40 bytes)
Blue = Original IPv6 Header (40 bytes)
Red = UDP Header (8 bytes)
Green = Payload (40 bytes)

Access Router (rftun0):

18:54:22.000838 IP6 (hlim 63, next-header IPv6 (41) payload length: 88) 2001:470:e23d:e000::1000 > 2001:470:e23d:190d:601:12c1:6744:52d3: IP6 (hlim 63, next-header UDP (17) payload length: 48) 2001:470:e23d:e012::12.44081 > 2001:470:e23d:f601::1.7: [udp sum ok] UDP, length 40

```
0x0000:  6000 0000 0058 293f 2001 0470 e23d e000  `....X)?...p.=..
0x0010:  0000 0000 0000 1000 2001 0470 e23d 190d  ..........p.=..
0x0020:  0601 12c1 6744 52d3 6000 0000 0030 113f  ....gDR.`....0.?
0x0030:  2001 0470 e23d e012 0000 0000 0000 0012  ...p.=.........
0x0040:  2001 0470 e23d f601 0000 0000 0000 0001  ...p.=.........
0x0050:  ac31 0007 0030 6eef 0000 00e1 0000 0000  .1...0n.........
0x0060:  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x0070:  0000 0000 0000 0000 0000 0000 0000 0000  ................
```

18:54:22.451788 IP6 (hlim 64, next-header IPv6 (41) payload length: 88)
2001:470:e23d:190d:601:12c1:6744:52d3 > 2001:470:e23d:e000::1000: IP6 (hlim 64, next-header UDP (17) payload length: 48) 2001:470:e23d:e000::601.7 > 2001:470:e23d:e012::12.44081: [udp sum ok] UDP, length 40

```
0x0000:  6000 0000 0058 2940 2001 0470 e23d 190d  `....X)@...p.=..
0x0010:  0601 12c1 6744 52d3 2001 0470 e23d e000  ....gDR....p.=..
0x0020:  0000 0000 0000 1000 6000 0000 0030 1140  ........`....0.@
0x0030:  2001 0470 e23d e000 0000 0000 0000 0601  ...p.=.........
0x0040:  2001 0470 e23d e012 0000 0000 0000 0012  ...p.=.........
0x0050:  0007 ac31 0030 7ef0 0000 00e1 0000 0000  ...1.0~.........
0x0060:  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x0070:  0000 0000 0000 0000 0000 0000 0000 0000  ................
```

Purple = Outer IPv6 Header (40 bytes)
Blue = Original IPv6 Header (40 bytes)
Red = UDP Header (8 bytes)
Green = Payload (40 bytes)

Access Router (eth2):

18:54:22.001004 IP (tos 0x0, ttl 64, id 24576, offset 0, flags [DF], proto TCP (6), length 184)
10.0.6.5.43828 > 10.0.6.200.4507: Flags [P.], cksum 0x2177 (incorrect -> 0x3641), seq 231697:231841, ack 245427, win 895, length 144

```
0x0000:  4500 00b8 6000 4000 4006 b973 0a00 0605  E...`.@.@..s....
0x0010:  0a00 06c8 ab34 119b 6635 55cc 0e57 e56a  .....4..f5U..W.j
0x0020:  5018 037f 2177 0000 0300 0090 0000 0001  P...!w..........
0x0030:  0000 0000 0000 0080 6000 0000 0058 293f  ........`....X)?
0x0040:  2001 0470 e23d e000 0000 0000 0000 1000  ...p.=..........
0x0050:  2001 0470 e23d 190d 0601 12c1 6744 52d3  ...p.=......gDR.
0x0060:  6000 0000 0030 113f 2001 0470 e23d e012  `....0.?...p.=..
0x0070:  0000 0000 0000 0012 2001 0470 e23d f601  ...........p.=..
0x0080:  0000 0000 0000 0001 ac31 0007 0030 6eef  .........1...0n.
0x0090:  0000 00e1 0000 0000 0000 0000 0000 0000  ................
0x00a0:  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x00b0:  0000 0000 0000 0000                      ........
```

18:54:22.451679 IP (tos 0x0, ttl 64, id 9791, offset 0, flags [DF], proto TCP (6), length 184)
10.0.6.200.4507 > 10.0.6.5.43828: Flags [P.], cksum 0x2e3c (correct), seq 245427:245571, ack 231841, win 2800, length 144

```
0x0000:  4500 00b8 263f 4000 4006 f334 0a00 06c8  E...&?@.@..4....
0x0010:  0a00 0605 119b ab34 0e57 e56a 6635 565c  .......4.W.jf5V\
0x0020:  5018 0af0 2e3c 0000 0301 0090 0000 0002  P....<..........
0x0030:  0000 0000 0000 0080 6000 0000 0058 2940  ........`....X)@
0x0040:  2001 0470 e23d 190d 0601 12c1 6744 52d3  ...p.=......gDR.
0x0050:  2001 0470 e23d e000 0000 0000 0000 1000  ...p.=..........
0x0060:  6000 0000 0030 1140 2001 0470 e23d e000  `....0.@...p.=..
0x0070:  0000 0000 0000 0601 2001 0470 e23d e012  ...........p.=..
0x0080:  0000 0000 0000 0012 0007 ac31 0030 7ef0  ...........1.0~.
0x0090:  0000 00e1 0000 0000 0000 0000 0000 0000  ................
0x00a0:  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x00b0:  0000 0000 0000 0000                      ........
```

Yellow = IPv4 Header (20 bytes) – Not transmitted over RF
Light Green = TCP Header (20 bytes) – Not transmitted over RF
Dark Red = Rockwell Radio Interface Header (16 bytes) - Not transmitted over RF
Purple = Outer IPv6 Header (40 bytes)
Blue = Original IPv6 Header (40 bytes)
Red = UDP Header (8 bytes)
Green = Payload (40 bytes)

Mobile Router (eth2):

18:54:22.258921 IP (tos 0x0, ttl 64, id 11966, offset 0, flags [DF], proto TCP (6), length 184)
10.0.2.200.4507 > 10.0.2.6.43530: Flags [P.], cksum 0x98d3 (correct), seq 331109:331253, ack 205201, win 2800, length 144

```
0x0000:  4500 00b8 2ebe 4000 4006 f2b4 0a00 02c8  E.....@.@.......
0x0010:  0a00 0206 119b aa0a 80a7 7414 98b1 c17a  ..........t....z
0x0020:  5018 0af0 98d3 0000 0301 0090 0000 0001  P...............
0x0030:  0000 0000 0000 0080 6000 0000 0058 293f  ........`....X)?
```

0x0040: 2001 0470 e23d e000 0000 0000 0000 1000 ...p.=.....
0x0050: 2001 0470 e23d 190d 0601 12c1 6744 52d3 ...p.=.....gDR.
0x0060: 6000 0000 0030 113f 2001 0470 e23d e012 `....0.?...p.=..
0x0070: 0000 0000 0000 0012 2001 0470 e23d f601 .....p.=..
0x0080: 0000 0000 0000 0001 ac31 0007 0030 6eef .....1...0n.
0x0090: 0000 00e1 0000 0000 0000 0000 0000 0000 .....
0x00a0: 0000 0000 0000 0000 0000 0000 0000 0000 .....
0x00b0: 0000 0000 0000 0000 .....

18:54:22.259270 IP (tos 0x0, ttl 64, id 45524, offset 0, flags [DF], proto TCP (6), length 184)
10.0.2.6.43530 > 10.0.2.200.4507: Flags [P.], cksum 0x1978 (incorrect -> 0x9e6b), seq 205201:205345, ack 331253, win 1222, length 144
0x0000: 4500 00b8 b1d4 4000 4006 6f9e 0a00 0206 E.....@.@.o.....
0x0010: 0a00 02c8 aa0a 119b 98b1 c17a 80a7 74a4 .....z..t.
0x0020: 5018 04c6 1978 0000 0300 0090 0000 0002 P....x.....
0x0030: 0000 0000 0000 0080 6000 0000 0058 2940 .....`....X)@
0x0040: 2001 0470 e23d 190d 0601 12c1 6744 52d3 ...p.=.....gDR.
0x0050: 2001 0470 e23d e000 0000 0000 0000 1000 ...p.=.....
0x0060: 6000 0000 0030 1140 2001 0470 e23d e000 `....0.@...p.=..
0x0070: 0000 0000 0000 0601 2001 0470 e23d e012 .....p.=..
0x0080: 0000 0000 0000 0012 0007 ac31 0030 7ef0 .....1.0~.
0x0090: 0000 00e1 0000 0000 0000 0000 0000 0000 .....
0x00a0: 0000 0000 0000 0000 0000 0000 0000 0000 .....
0x00b0: 0000 0000 0000 0000 .....

Yellow = IPv4 Header (20 bytes) – Not transmitted over RF
Light Green = TCP Header (20 bytes) – Not transmitted over RF
Dark Red = Rockwell Radio Interface Header (16 bytes) - Not transmitted over RF
Purple = Outer IPv6 Header (40 bytes)
Blue = Original IPv6 Header (40 bytes)
Red = UDP Header (8 bytes)
Green = Payload (40 bytes)

Mobile Router (rftun0):

18:54:22.259020 IP6 (hlim 63, next-header IPv6 (41) payload length: 88) 2001:470:e23d:e000::1000 > 2001:470:e23d:190d:601:12c1:6744:52d3: IP6 (hlim 63, next-header UDP (17) payload length: 48) 2001:470:e23d:e012::12.44081 > 2001:470:e23d:f601::1.7: [udp sum ok] UDP, length 40
0x0000: 6000 0000 0058 293f 2001 0470 e23d e000 `....X)?...p.=..
0x0010: 0000 0000 0000 1000 2001 0470 e23d 190d .....p.=..
0x0020: 0601 12c1 6744 52d3 6000 0000 0030 113f ....gDR.`....0.?
0x0030: 2001 0470 e23d e012 0000 0000 0000 0012 ...p.=.....
0x0040: 2001 0470 e23d f601 0000 0000 0000 0001 ...p.=.....
0x0050: ac31 0007 0030 6eef 0000 00e1 0000 0000 .1...0n.....
0x0060: 0000 0000 0000 0000 0000 0000 0000 0000 .....
0x0070: 0000 0000 0000 0000 0000 0000 0000 0000 .....

18:54:22.259175 IP6 (hlim 64, next-header IPv6 (41) payload length: 88) 2001:470:e23d:190d:601:12c1:6744:52d3 > 2001:470:e23d:e000::1000: IP6 (hlim 64, next-header UDP (17) payload length: 48) 2001:470:e23d:e000::601.7 > 2001:470:e23d:e012::12.44081: [udp sum ok] UDP, length 40

0x0000: 6000 0000 0058 2940 2001 0470 e23d 190d `....X)@...p.=..
0x0010: 0601 12c1 6744 52d3 2001 0470 e23d e000 ....gDR....p.=..
0x0020: 0000 0000 0000 1000 6000 0000 0030 1140 ....`....0.@
0x0030: 2001 0470 e23d e000 0000 0000 0000 0601 ...p.=.........
0x0040: 2001 0470 e23d e012 0000 0000 0000 0012 ...p.=.........
0x0050: 0007 ac31 0030 7ef0 0000 00e1 0000 0000 ...1.0~.........
0x0060: 0000 0000 0000 0000 0000 0000 0000 0000 ....
0x0070: 0000 0000 0000 0000 0000 0000 0000 0000 ....

Purple = Outer IPv6 Header (40 bytes)
Blue = Original IPv6 Header (40 bytes)
Red = UDP Header (8 bytes)
Green = Payload (40 bytes)

Example Packet Flow with without ROHC and with IPsec ESP

CN (eth1):

23:58:37.000091 IP6 (hlim 64, next-header UDP (17) payload length: 48) 2001:470:e23d:e012::12.50431 > 2001:470:e23d:f601::1.7: [udp sum ok] UDP, length 40
0x0000: 6000 0000 0030 1140 2001 0470 e23d e012 `....0.@...p.=..
0x0010: 0000 0000 0000 0012 2001 0470 e23d f601 ....p.=..
0x0020: 0000 0000 0000 0001 c4ff 0007 0030 5621 ....0V!
0x0030: 0000 00e1 0000 0000 0000 0000 0000 0000 ....
0x0040: 0000 0000 0000 0000 0000 0000 0000 0000 ....
0x0050: 0000 0000 0000 0000 ....

23:58:37.455443 IP6 (hlim 63, next-header UDP (17) payload length: 48) 2001:470:e23d:e000::601.7 > 2001:470:e23d:e012::12.50431: [udp sum ok] UDP, length 40
0x0000: 6000 0000 0030 113f 2001 0470 e23d e000 `....0.?...p.=..
0x0010: 0000 0000 0000 0601 2001 0470 e23d e012 ....p.=..
0x0020: 0000 0000 0000 0012 0007 c4ff 0030 6622 ....0f"
0x0030: 0000 00e1 0000 0000 0000 0000 0000 0000 ....
0x0040: 0000 0000 0000 0000 0000 0000 0000 0000 ....
0x0050: 0000 0000 0000 0000 ....

Blue = Original IPv6 Header (40 bytes)
Red = UDP Header (8 bytes)
Green = Payload (40 bytes)

HA(six0):

23:58:37.000232 IP6 (hlim 64, next-header ESP (50) payload length: 112) 2001:470:e23d:e000::1000 > 2001:470:e23d:1c41:601:7acd:4a3e:3702: ESP(spi=0x00000014,seq=0x59d), length 112
0x0000: 6000 0000 0070 3240 2001 0470 e23d e000 `....p2@...p.=..
0x0010: 0000 0000 0000 1000 2001 0470 e23d 1c41 ....p.=.A
0x0020: 0601 7acd 4a3e 3702 0000 0014 0000 059d ..z.J>7.........
0x0030: 6000 0000 0030 113f 2001 0470 e23d e012 `....0.?...p.=..
0x0040: 0000 0000 0000 0012 2001 0470 e23d f601 ....p.=..
0x0050: 0000 0000 0000 0001 c4ff 0007 0030 5621 ....0V!
0x0060: 0000 00e1 0000 0000 0000 0000 0000 0000 ....

0x0070: 0000 0000 0000 0000 0000 0000 0000 0000 ....
0x0080: 0000 0000 0000 0000 0102 0229 e547 03ad ....).G..
0x0090: e21b 48eb 3ee7 9df6 ..H.>...

23:58:37.455318 IP6 (hlim 63, next-header ESP (50) payload length: 112)
2001:470:e23d:1c41:601:7acd:4a3e:3702 > 2001:470:e23d:e000::1000:
ESP(spi=0x00000013,seq=0x59e), length 112
0x0000: 6000 0000 0070 323f 2001 0470 e23d 1c41 `....p2?...p.=.A
0x0010: 0601 7acd 4a3e 3702 2001 0470 e23d e000 ..z.J>7....p.=..
0x0020: 0000 0000 0000 1000 0000 0013 0000 059e ....
0x0030: 6000 0000 0030 1140 2001 0470 e23d e000 `....0.@...p.=..
0x0040: 0000 0000 0000 0601 2001 0470 e23d e012 ....p.=..
0x0050: 0000 0000 0000 0012 0007 c4ff 0030 6622 ....0f"
0x0060: 0000 00e1 0000 0000 0000 0000 0000 0000 ....
0x0070: 0000 0000 0000 0000 0000 0000 0000 0000 ....
0x0080: 0000 0000 0000 0000 0102 0229 46e2 1170 ....)F..p
0x0090: d67e 6248 9ac6 1e95 .~bH....

Purple = Outer IPv6 Header (40 bytes)
Orange = IPsec ESP Header (8 bytes)
Blue = Original IPv6 Header (40 bytes)
Red = UDP Header (8 bytes)
Green = Payload (40 bytes)
Orange = IPsec ESP Trailer (16 bytes)

Access Router (rftun0):

23:58:37.000706 IP6 (hlim 63, next-header ESP (50) payload length: 112) 2001:470:e23d:e000::1000 >
2001:470:e23d:1c41:601:7acd:4a3e:3702: ESP(spi=0x00000014,seq=0x59d), length 112
0x0000: 6000 0000 0070 323f 2001 0470 e23d e000 `....p2?...p.=..
0x0010: 0000 0000 0000 1000 2001 0470 e23d 1c41 ....p.=.A
0x0020: 0601 7acd 4a3e 3702 0000 0014 0000 059d ..z.J>7.........
0x0030: 6000 0000 0030 113f 2001 0470 e23d e012 `....0.?...p.=..
0x0040: 0000 0000 0000 0012 2001 0470 e23d f601 ....p.=..
0x0050: 0000 0000 0000 0001 c4ff 0007 0030 5621 ....0V!
0x0060: 0000 00e1 0000 0000 0000 0000 0000 0000 ....
0x0070: 0000 0000 0000 0000 0000 0000 0000 0000 ....
0x0080: 0000 0000 0000 0000 0102 0229 e547 03ad ....).G..
0x0090: e21b 48eb 3ee7 9df6 ..H.>...

23:58:37.454818 IP6 (hlim 64, next-header ESP (50) payload length: 112)
2001:470:e23d:1c41:601:7acd:4a3e:3702 > 2001:470:e23d:e000::1000:
ESP(spi=0x00000013,seq=0x59e), length 112
0x0000: 6000 0000 0070 3240 2001 0470 e23d 1c41 `....p2@...p.=.A
0x0010: 0601 7acd 4a3e 3702 2001 0470 e23d e000 ..z.J>7....p.=..
0x0020: 0000 0000 0000 1000 0000 0013 0000 059e ....
0x0030: 6000 0000 0030 1140 2001 0470 e23d e000 `....0.@...p.=..
0x0040: 0000 0000 0000 0601 2001 0470 e23d e012 ....p.=..
0x0050: 0000 0000 0000 0012 0007 c4ff 0030 6622 ....0f"
0x0060: 0000 00e1 0000 0000 0000 0000 0000 0000 ....

0x0070: 0000 0000 0000 0000 0000 0000 0000 0000 ....
0x0080: 0000 0000 0000 0000 0102 0229 46e2 1170 ....)F..p
0x0090: d67e 6248 9ac6 1e95 .~bH....

Purple = Outer IPv6 Header (40 bytes)
Orange = IPsec ESP Header (8 bytes)
Blue = Original IPv6 Header (40 bytes)
Red = UDP Header (8 bytes)
Green = Payload (40 bytes)
Orange = IPsec ESP Trailer (16 bytes)

Access Router (eth2):

23:58:37.000860 IP (tos 0x0, ttl 64, id 29965, offset 0, flags [DF], proto TCP (6), length 208)
10.0.6.5.42515 > 10.0.6.200.4507: Flags [P.], cksum 0x218f (incorrect -> 0xd091), seq 254913:255081, ack 262658, win 584, length 168
0x0000: 4500 00d0 750d 4000 4006 a44e 0a00 0605 E...u.@.@..N....
0x0010: 0a00 06c8 a613 119b 7769 fea3 6583 0abb ....wi..e...
0x0020: 5018 0248 218f 0000 0300 00a8 0000 0001 P..H!...........
0x0030: 0000 0000 0000 0098 6000 0000 0070 323f ....`....p2?
0x0040: 2001 0470 e23d e000 0000 0000 0000 1000 ...p.=..........
0x0050: 2001 0470 e23d 1c41 0601 7acd 4a3e 3702 ...p.=.A..z.J>7.
0x0060: 0000 0014 0000 059d 6000 0000 0030 113f ....`....0.?
0x0070: 2001 0470 e23d e012 0000 0000 0000 0012 ...p.=..........
0x0080: 2001 0470 e23d f601 0000 0000 0000 0001 ...p.=..........
0x0090: c4ff 0007 0030 5621 0000 00e1 0000 0000 .....0V!........
0x00a0: 0000 0000 0000 0000 0000 0000 0000 0000 ....
0x00b0: 0000 0000 0000 0000 0000 0000 0000 0000 ....
0x00c0: 0102 0229 e547 03ad e21b 48eb 3ee7 9df6 ...).G....H.>...

23:58:37.454710 IP (tos 0x0, ttl 64, id 6556, offset 0, flags [DF], proto TCP (6), length 208)
10.0.6.200.4507 > 10.0.6.5.42515: Flags [P.], cksum 0x6da2 (correct), seq 262658:262826, ack 255081, win 2800, length 168
0x0000: 4500 00d0 199c 4000 4006 ffbf 0a00 06c8 E.....@.@.......
0x0010: 0a00 0605 119b a613 6583 0abb 7769 ff4b ....e...wi.K
0x0020: 5018 0af0 6da2 0000 0301 00a8 0000 0002 P...m...........
0x0030: 0000 0000 0000 0098 6000 0000 0070 3240 ....`....p2@
0x0040: 2001 0470 e23d 1c41 0601 7acd 4a3e 3702 ...p.=.A..z.J>7.
0x0050: 2001 0470 e23d e000 0000 0000 0000 1000 ...p.=..........
0x0060: 0000 0013 0000 059e 6000 0000 0030 1140 ....`....0.@
0x0070: 2001 0470 e23d e000 0000 0000 0000 0601 ...p.=..........
0x0080: 2001 0470 e23d e012 0000 0000 0000 0012 ...p.=..........
0x0090: 0007 c4ff 0030 6622 0000 00e1 0000 0000 .....0f"........
0x00a0: 0000 0000 0000 0000 0000 0000 0000 0000 ....
0x00b0: 0000 0000 0000 0000 0000 0000 0000 0000 ....
0x00c0: 0102 0229 46e2 1170 d67e 6248 9ac6 1e95 ...)F..p.~bH....

Yellow = IPv4 Header (20 bytes) – Not transmitted over RF
Light Green = TCP Header (20 bytes) – Not transmitted over RF
Dark Red = Rockwell Radio Interface Header (16 bytes) - Not transmitted over RF
Purple = Outer IPv6 Header (40 bytes)
Blue = Original IPv6 Header (40 bytes)
Orange = IPsec ESP Header (8 bytes)
Red = UDP Header (8 bytes)
Green = Payload (40 bytes)
Orange = IPsec ESP Trailer (16 bytes)

Mobile Router (eth2):

23:58:37.260286 IP (tos 0x0, ttl 64, id 64922, offset 0, flags [DF], proto TCP (6), length 208)
10.0.2.200.4507 > 10.0.2.6.42904: Flags [P.], cksum 0x6e97 (correct), seq 304899:305067, ack 239889, win 2800, length 168

```
0x0000:  4500 00d0 fd9a 4000 4006 23c0 0a00 02c8  E.....@.@.#.....
0x0010:  0a00 0206 119b a798 ca3d 4915 db60 5763  .........=I..`Wc
0x0020:  5018 0af0 6e97 0000 0301 00a8 0000 0001  P...n...........
0x0030:  0000 0000 0000 0098 6000 0000 0070 323f  ........`....p2?
0x0040:  2001 0470 e23d e000 0000 0000 0000 1000  ...p.=..........
0x0050:  2001 0470 e23d 1c41 0601 7acd 4a3e 3702  ...p.=.A..z.J>7.
0x0060:  0000 0014 0000 059d 6000 0000 0030 113f  ........`....0.?
0x0070:  2001 0470 e23d e012 0000 0000 0000 0012  ...p.=..........
0x0080:  2001 0470 e23d f601 0000 0000 0000 0001  ...p.=..........
0x0090:  c4ff 0007 0030 5621 0000 00e1 0000 0000  .....0V!........
0x00a0:  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x00b0:  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x00c0:  0102 0229 e547 03ad e21b 48eb 3ee7 9df6  ...).G....H.>...
```

23:58:37.260664 IP (tos 0x0, ttl 64, id 44384, offset 0, flags [DF], proto TCP (6), length 208)
10.0.2.6.42904 > 10.0.2.200.4507: Flags [P.], cksum 0x1990 (incorrect -> 0x1a99), seq 239889:240057, ack 305067, win 1193, length 168

```
0x0000:  4500 00d0 ad60 4000 4006 73fa 0a00 0206  E....`@.@.s.....
0x0010:  0a00 02c8 a798 119b db60 5763 ca3d 49bd  .........`Wc.=I.
0x0020:  5018 04a9 1990 0000 0300 00a8 0000 0002  P...............
0x0030:  0000 0000 0000 0098 6000 0000 0070 3240  ........`....p2@
0x0040:  2001 0470 e23d 1c41 0601 7acd 4a3e 3702  ...p.=.A..z.J>7.
0x0050:  2001 0470 e23d e000 0000 0000 0000 1000  ...p.=..........
0x0060:  0000 0013 0000 059e 6000 0000 0030 1140  ........`....0.@
0x0070:  2001 0470 e23d e000 0000 0000 0000 0601  ...p.=..........
0x0080:  2001 0470 e23d e012 0000 0000 0000 0012  ...p.=..........
0x0090:  0007 c4ff 0030 6622 0000 00e1 0000 0000  .....0f"........
0x00a0:  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x00b0:  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x00c0:  0102 0229 46e2 1170 d67e 6248 9ac6 1e95  ...)F..p.~bH....
```

Yellow = IPv4 Header (20 bytes) – Not transmitted over RF
Light Green = TCP Header (20 bytes) – Not transmitted over RF
Dark Red = Rockwell Radio Interface Header (16 bytes) - Not transmitted over RF
Purple = Outer IPv6 Header (40 bytes)
Blue = Original IPv6 Header (40 bytes)
Orange = IPsec ESP Header (8 bytes)
Red = UDP Header (8 bytes)
Green = Payload (40 bytes)
Orange = IPsec ESP Trailer (16 bytes)

Mobile Router (rftun0):

23:58:37.260400 IP6 (hlim 63, next-header ESP (50) payload length: 112) 2001:470:e23d:e000::1000 > 2001:470:e23d:1c41:601:7acd:4a3e:3702: ESP(spi=0x00000014,seq=0x59d), length 112

```
        0x0000:  6000 0000 0070 323f 2001 0470 e23d e000  `....p2?...p.=..
        0x0010:  0000 0000 0000 1000 2001 0470 e23d 1c41  ..........p.=.A
        0x0020:  0601 7acd 4a3e 3702 0000 0014 0000 059d  ..z.J>7.........
        0x0030:  6000 0000 0030 113f 2001 0470 e23d e012  `....0.?...p.=..
        0x0040:  0000 0000 0000 0012 2001 0470 e23d f601  ..........p.=..
        0x0050:  0000 0000 0000 0001 c4ff 0007 0030 5621  .............0V!
        0x0060:  0000 00e1 0000 0000 0000 0000 0000 0000  ................
        0x0070:  0000 0000 0000 0000 0000 0000 0000 0000  ................
        0x0080:  0000 0000 0000 0000 0102 0229 e547 03ad  ...........).G..
        0x0090:  e21b 48eb 3ee7 9df6                      ..H.>...
```

23:58:37.260575 IP6 (hlim 64, next-header ESP (50) payload length: 112) 2001:470:e23d:1c41:601:7acd:4a3e:3702 > 2001:470:e23d:e000::1000: ESP(spi=0x00000013,seq=0x59e), length 112

```
        0x0000:  6000 0000 0070 3240 2001 0470 e23d 1c41  `....p2@...p.=.A
        0x0010:  0601 7acd 4a3e 3702 2001 0470 e23d e000  ..z.J>7....p.=..
        0x0020:  0000 0000 0000 1000 0000 0013 0000 059e  ................
        0x0030:  6000 0000 0030 1140 2001 0470 e23d e000  `....0.@...p.=..
        0x0040:  0000 0000 0000 0601 2001 0470 e23d e012  ..........p.=..
        0x0050:  0000 0000 0000 0012 0007 c4ff 0030 6622  .............0f"
        0x0060:  0000 00e1 0000 0000 0000 0000 0000 0000  ................
        0x0070:  0000 0000 0000 0000 0000 0000 0000 0000  ................
        0x0080:  0000 0000 0000 0000 0102 0229 46e2 1170  ...........)F..p
        0x0090:  d67e 6248 9ac6 1e95                      .~bH....
```

Purple = Outer IPv6 Header (40 bytes)
Blue = Original IPv6 Header (40 bytes)
Orange = IPsec ESP Header (8 bytes)
Red = UDP Header (8 bytes)
Green = Payload (40 bytes)
Orange = IPsec ESP Trailer (16 bytes)

Example Packet Flow with with ROHC and without IPsec ESP

CN (eth1):

20:45:49.000102 IP6 (hlim 64, next-header UDP (17) payload length: 48) 2001:470:e23d:e012::12.37640 > 2001:470:e23d:f601::1.7: [udp sum ok] UDP, length 40

```
0x0000:  6000 0000 0030 1140 2001 0470 e23d e012  `....0.@...p.=..
0x0010:  0000 0000 0000 0012 2001 0470 e23d f601  ..........p.=..
0x0020:  0000 0000 0000 0001 9308 0007 0030 8818  .............0..
0x0030:  0000 00e1 0000 0000 0000 0000 0000 0000  ................
0x0040:  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x0050:  0000 0000 0000 0000                      ........
```

20:45:49.153822 IP6 (hlim 63, next-header UDP (17) payload length: 48) 2001:470:e23d:e000::601.7 > 2001:470:e23d:e012::12.37640: [udp sum ok] UDP, length 40

```
0x0000:  6000 0000 0030 113f 2001 0470 e23d e000  `....0.?...p.=..
0x0010:  0000 0000 0000 0601 2001 0470 e23d e012  ..........p.=..
0x0020:  0000 0000 0000 0012 0007 9308 0030 9819  .............0..
0x0030:  0000 00e1 0000 0000 0000 0000 0000 0000  ................
0x0040:  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x0050:  0000 0000 0000 0000                      ........
```

Blue = IPv6 Header (40 bytes)
Red = UDP Header (8 bytes)
Green = Payload (40 bytes)

HA (six0):

20:45:49.000027 IP6 (hlim 64, next-header IPv6 (41) payload length: 88) 2001:470:e23d:e000::1000 > 2001:470:e23d:1f8d:601:7da2:55cd:33f6: IP6 (hlim 63, next-header UDP (17) payload length: 48) 2001:470:e23d:e012::12.37640 > 2001:470:e23d:f601::1.7: [udp sum ok] UDP, length 40

```
0x0000:  6000 0000 0058 2940 2001 0470 e23d e000  `....X)@...p.=..
0x0010:  0000 0000 0000 1000 2001 0470 e23d 1f8d  ..........p.=..
0x0020:  0601 7da2 55cd 33f6 6000 0000 0030 113f  ..}.U.3.`....0.?
0x0030:  2001 0470 e23d e012 0000 0000 0000 0012  ...p.=..........
0x0040:  2001 0470 e23d f601 0000 0000 0000 0001  ...p.=..........
0x0050:  9308 0007 0030 8818 0000 00e1 0000 0000  .....0..........
0x0060:  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x0070:  0000 0000 0000 0000 0000 0000 0000 0000  ................
```

20:45:49.153487 IP6 (hlim 63, next-header IPv6 (41) payload length: 88) 2001:470:e23d:1f8d:601:7da2:55cd:33f6 > 2001:470:e23d:e000::1000: IP6 (hlim 64, next-header UDP (17) payload length: 48) 2001:470:e23d:e000::601.7 > 2001:470:e23d:e012::12.37640: [udp sum ok] UDP, length 40

```
0x0000:  6000 0000 0058 293f 2001 0470 e23d 1f8d  `....X)?...p.=..
0x0010:  0601 7da2 55cd 33f6 2001 0470 e23d e000  ..}.U.3....p.=..
0x0020:  0000 0000 0000 1000 6000 0000 0030 1140  ........`....0.@
0x0030:  2001 0470 e23d e000 0000 0000 0000 0601  ...p.=..........
0x0040:  2001 0470 e23d e012 0000 0000 0000 0012  ...p.=..........
0x0050:  0007 9308 0030 9819 0000 00e1 0000 0000  .....0..........
0x0060:  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x0070:  0000 0000 0000 0000 0000 0000 0000 0000  ................
```

Purple = Outer IPv6 Header (40 bytes)
Blue = Original IPv6 Header (40 bytes)
Red = UDP Header (8 bytes)
Green = Payload (40 bytes)

Access Router (rftun0):

20:45:49.000529 IP6 (hlim 63, next-header IPv6 (41) payload length: 88) 2001:470:e23d:e000::1000 > 2001:470:e23d:1f8d:601:7da2:55cd:33f6: IP6 (hlim 63, next-header UDP (17) payload length: 48) 2001:470:e23d:e012::12.37640 > 2001:470:e23d:f601::1.7: [udp sum ok] UDP, length 40

```
        0x0000:  6000 0000 0058 293f 2001 0470 e23d e000  `....X)?...p.=..
        0x0010:  0000 0000 0000 1000 2001 0470 e23d 1f8d  ..........p.=..
        0x0020:  0601 7da2 55cd 33f6 6000 0000 0030 113f  ..}.U.3.`....0.?
        0x0030:  2001 0470 e23d e012 0000 0000 0000 0012  ...p.=.........
        0x0040:  2001 0470 e23d f601 0000 0000 0000 0001  ...p.=.........
        0x0050:  9308 0007 0030 8818 0000 00e1 0000 0000  .....0..........
        0x0060:  0000 0000 0000 0000 0000 0000 0000 0000  ................
        0x0070:  0000 0000 0000 0000 0000 0000 0000 0000  ................
```

20:45:49.153140 IP6 (hlim 64, next-header IPv6 (41) payload length: 88) 2001:470:e23d:1f8d:601:7da2:55cd:33f6 > 2001:470:e23d:e000::1000: IP6 (hlim 64, next-header UDP (17) payload length: 48) 2001:470:e23d:e000::601.7 > 2001:470:e23d:e012::12.37640: [udp sum ok] UDP, length 40

```
        0x0000:  6000 0000 0058 2940 2001 0470 e23d 1f8d  `....X)@...p.=..
        0x0010:  0601 7da2 55cd 33f6 2001 0470 e23d e000  ..}.U.3....p.=..
        0x0020:  0000 0000 0000 1000 6000 0000 0030 1140  ........`....0.@
        0x0030:  2001 0470 e23d e000 0000 0000 0000 0601  ...p.=.........
        0x0040:  2001 0470 e23d e012 0000 0000 0000 0012  ...p.=.........
        0x0050:  0007 9308 0030 9819 0000 00e1 0000 0000  .....0..........
        0x0060:  0000 0000 0000 0000 0000 0000 0000 0000  ................
        0x0070:  0000 0000 0000 0000 0000 0000 0000 0000  ................
```

Purple = Outer IPv6 Header (40 bytes)
Blue = Original IPv6 Header (40 bytes)
Red = UDP Header (8 bytes)
Green = Payload (40 bytes)

Access Router (eth2):

20:45:49.001698 IP (tos 0x0, ttl 64, id 53847, offset 0, flags [DF], proto TCP (6), length 100)
10.0.6.5.40520 > 10.0.6.200.4507: Flags [P.], cksum 0x2123 (incorrect -> 0x6f4f), seq 115323:115383, ack 137005, win 207, length 60

```
        0x0000:  4500 0064 d257 4000 4006 4770 0a00 0605  E..d.W@.@.Gp....
        0x0010:  0a00 06c8 9e48 119b d685 2a98 f3b3 0840  .....H....*....@
        0x0020:  5018 00cf 2123 0000 0300 003c 0000 0001  P...!#.....<....
        0x0030:  0000 0000 0000 002c e54d 8818 0000 00e1  .......,.M......
        0x0040:  0000 0000 0000 0000 0000 0000 0000 0000  ................
        0x0050:  0000 0000 0000 0000 0000 0000 0000 0000  ................
        0x0060:  0000 0000                                ....
```

20:45:49.152932 IP (tos 0x0, ttl 64, id 59668, offset 0, flags [DF], proto TCP (6), length 100)
10.0.6.200.4507 > 10.0.6.5.40520: Flags [P.], cksum 0x551a (correct), seq 137028:137088, ack 115383, win 2800, length 60

```
0x0000:  4500 0064 e914 4000 4006 30b3 0a00 06c8  E..d..@.@.0.....
0x0010:  0a00 0605 119b 9e48 f3b3 0857 d685 2ad4  .......H...W..*.
0x0020:  5018 0af0 551a 0000 0301 003c 0000 0002  P...U......<....
0x0030:  0000 0000 0000 002c e50b 9819 0000 00e1  .......,........
0x0040:  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x0050:  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x0060:  0000 0000                                ....
```

Yellow = IPv4 Header (20 bytes) – Not transmitted over RF
Light Green = TCP Header (20 bytes) – Not transmitted over RF
Dark Red = Rockwell Radio Interface Header (16 bytes) - Not transmitted over RF
Dark Blue = ROHC Header (4 bytes)
Green = Payload (40 bytes)

Mobile Router (eth2):

20:45:49.059840 IP (tos 0x0, ttl 64, id 34787, offset 0, flags [DF], proto TCP (6), length 100)
10.0.2.200.4507 > 10.0.2.6.52111: Flags [P.], cksum 0x431b (correct), seq 218895:218955, ack 93283, win 2800, length 60

```
0x0000:  4500 0064 87e3 4000 4006 99e3 0a00 02c8  E..d..@.@.......
0x0010:  0a00 0206 119b cb8f e21f 6035 9ddb 19ab  ..........`5....
0x0020:  5018 0af0 431b 0000 0301 003c 0000 0001  P...C......<....
0x0030:  0000 0000 0000 002c e54d 8818 0000 00e1  .......,.M......
0x0040:  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x0050:  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x0060:  0000 0000                                ....
```

20:45:49.061246 IP (tos 0x0, ttl 64, id 22634, offset 0, flags [DF], proto TCP (6), length 100)
10.0.2.6.52111 > 10.0.2.200.4507: Flags [P.], cksum 0x1924 (incorrect -> 0x3a75), seq 93283:93343, ack 218955, win 923, length 60

```
0x0000:  4500 0064 586a 4000 4006 c95c 0a00 0206  E..dXj@.@..\....
0x0010:  0a00 02c8 cb8f 119b 9ddb 19ab e21f 6071  ..............`q
0x0020:  5018 039b 1924 0000 0300 003c 0000 0002  P....$.....<....
0x0030:  0000 0000 0000 002c e50b 9819 0000 00e1  .......,........
0x0040:  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x0050:  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x0060:  0000 0000                                ....
```

Yellow = IPv4 Header (20 bytes) – Not transmitted over RF
Light Green = TCP Header (20 bytes) – Not transmitted over RF
Dark Red = Rockwell Radio Interface Header (16 bytes) - Not transmitted over RF
Dark Blue = ROHC Header (4 bytes)
Green = Payload (40 bytes)

Mobile Router (rftun0):

20:45:49.060080 IP6 (hlim 63, next-header IPv6 (41) payload length: 88) 2001:470
:e23d:e000::1000 > 2001:470:e23d:1f8d:601:7da2:55cd:33f6: IP6 (hlim 63, next-hea
der UDP (17) payload length: 48) 2001:470:e23d:e012::12.37640 > 2001:470:e23d:f6
01::1.7: [udp sum ok] UDP, length 40

```
0x0000:  6000 0000 0058 293f 2001 0470 e23d e000  `....X)?...p.=..
0x0010:  0000 0000 0000 1000 2001 0470 e23d 1f8d  ..........p.=..
0x0020:  0601 7da2 55cd 33f6 6000 0000 0030 113f  ..}.U.3.`....0.?
0x0030:  2001 0470 e23d e012 0000 0000 0000 0012  ...p.=.........
0x0040:  2001 0470 e23d f601 0000 0000 0000 0001  ...p.=.........
0x0050:  9308 0007 0030 8818 0000 00e1 0000 0000  .....0.........
0x0060:  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x0070:  0000 0000 0000 0000 0000 0000 0000 0000  ................
```

20:45:49.060277 IP6 (hlim 64, next-header IPv6 (41) payload length: 88)
2001:470:e23d:1f8d:601:7da2:55cd:33f6 > 2001:470:e23d:e000::1000: IP6 (hlim 64, next-header UDP
(17) payload length: 48) 2001:470:e23d:e000::601.7 > 2001:470:e23d:e012::12.37640: [udp sum ok]
UDP, length 40

```
0x0000:  6000 0000 0058 2940 2001 0470 e23d 1f8d  `....X)@...p.=..
0x0010:  0601 7da2 55cd 33f6 2001 0470 e23d e000  ..}.U.3....p.=..
0x0020:  0000 0000 0000 1000 6000 0000 0030 1140  ........`....0.@
0x0030:  2001 0470 e23d e000 0000 0000 0000 0601  ...p.=.........
0x0040:  2001 0470 e23d e012 0000 0000 0000 0012  ...p.=.........
0x0050:  0007 9308 0030 9819 0000 00e1 0000 0000  .....0.........
0x0060:  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x0070:  0000 0000 0000 0000 0000 0000 0000 0000  ................
```

Purple = Outer IPv6 Header (40 bytes)
Blue = Original IPv6 Header (40 bytes)
Red = UDP Header (8 bytes)
Green = Payload (40 bytes)

Example Packet Flow with ROHC and IPsec ESP

CN (eth1):

21:41:00.000089 IP6 (hlim 64, next-header UDP (17) payload length: 48) 2001:470:e23d:e012::12.47286
> 2001:470:e23d:f601::1.7: [udp sum ok] UDP, length 40

```
0x0000:  6000 0000 0030 1140 2001 0470 e23d e012  `....0.@...p.=..
0x0010:  0000 0000 0000 0012 2001 0470 e23d f601  ..........p.=..
0x0020:  0000 0000 0000 0001 b8b6 0007 0030 626a  .............0bj
0x0030:  0000 00e1 0000 0000 0000 0000 0000 0000  ................
0x0040:  0000 0000 0000 0000 0000 0000 0000 0000  ................
0x0050:  0000 0000 0000 0000                      ........
```

21:41:00.354686 IP6 (hlim 63, next-header UDP (17) payload length: 48) 2001:470:e23d:e000::601.7 >
2001:470:e23d:e012::12.47286: [udp sum ok] UDP, length 40

```
0x0000:  6000 0000 0030 113f 2001 0470 e23d e000  `....0.?...p.=..
0x0010:  0000 0000 0000 0601 2001 0470 e23d e012  ..........p.=..
0x0020:  0000 0000 0000 0012 0007 b8b6 0030 726b  .............0rk
```

0x0030: 0000 00e1 0000 0000 0000 0000 0000 0000 ....
0x0040: 0000 0000 0000 0000 0000 0000 0000 0000 ....
0x0050: 0000 0000 0000 0000 ....

Blue = IPv6 Header (40 bytes)
Red = UDP Header (8 bytes)
Green = Payload (40 bytes)

HA (six0):

21:41:00.000052 IP6 (hlim 64, next-header ESP (50) payload length: 112) 2001:470:e23d:e000::1000 > 2001:470:e23d:1dfa:601:7504:5cc7:37e8: ESP(spi=0x00000014,seq=0x599), length 112
0x0000: 6000 0000 0070 3240 2001 0470 e23d e000 `....p2@...p.=..
0x0010: 0000 0000 0000 1000 2001 0470 e23d 1dfa ....p.=..
0x0020: 0601 7504 5cc7 37e8 0000 0014 0000 0599 ..u.\.7.........
0x0030: 6000 0000 0030 113f 2001 0470 e23d e012 `....0.?...p.=..
0x0040: 0000 0000 0000 0012 2001 0470 e23d f601 ....p.=..
0x0050: 0000 0000 0000 0001 b8b6 0007 0030 626a 0bj
0x0060: 0000 00e1 0000 0000 0000 0000 0000 0000 ....
0x0070: 0000 0000 0000 0000 0000 0000 0000 0000 ....
0x0080: 0000 0000 0000 0000 0102 0229 c220 3174 ....)..1t
0x0090: a5f2 d242 f6ac ef8a ...B....

21:41:00.354370 IP6 (hlim 63, next-header ESP (50) payload length: 112) 2001:470:e23d:1dfa:601:7504:5cc7:37e8 > 2001:470:e23d:e000::1000: ESP(spi=0x00000013,seq=0x599), length 112
0x0000: 6000 0000 0070 323f 2001 0470 e23d 1dfa `....p2?...p.=..
0x0010: 0601 7504 5cc7 37e8 2001 0470 e23d e000 ..u.\.7....p.=..
0x0020: 0000 0000 0000 1000 0000 0013 0000 0599 ....
0x0030: 6000 0000 0030 1140 2001 0470 e23d e000 `....0.@...p.=..
0x0040: 0000 0000 0000 0601 2001 0470 e23d e012 ....p.=..
0x0050: 0000 0000 0000 0012 0007 b8b6 0030 726b 0rk
0x0060: 0000 00e1 0000 0000 0000 0000 0000 0000 ....
0x0070: 0000 0000 0000 0000 0000 0000 0000 0000 ....
0x0080: 0000 0000 0000 0000 0102 0229 b4f5 708e ....)..p.
0x0090: 2c72 3146 77cf bba8 ,r1Fw...

Purple = Outer IPv6 Header (40 bytes)
Blue = Original IPv6 Header (40 bytes)
Orange = IPsec ESP Header (8 bytes)
Red = UDP Header (8 bytes)
Green = Payload (40 bytes)
Orange = IPsec ESP Trailer (16 bytes)

Access Router (rftun0):

21:41:00.000598 IP6 (hlim 63, next-header ESP (50) payload length: 112) 2001:470:e23d:e000::1000 > 2001:470:e23d:1dfa:601:7504:5cc7:37e8: ESP(spi=0x00000014,seq=0x599), length 112
0x0000: 6000 0000 0070 323f 2001 0470 e23d e000 `....p2?...p.=..
0x0010: 0000 0000 0000 1000 2001 0470 e23d 1dfa ....p.=..
0x0020: 0601 7504 5cc7 37e8 0000 0014 0000 0599 ..u.\.7.........

0x0030: 6000 0000 0030 113f 2001 0470 e23d e012 `....0.?...p.=..
0x0040: 0000 0000 0000 0012 2001 0470 e23d f601 ....p.=..
0x0050: 0000 0000 0000 0001 b8b6 0007 0030 626a 0bj
0x0060: 0000 00e1 0000 0000 0000 0000 0000 0000 ....
0x0070: 0000 0000 0000 0000 0000 0000 0000 0000 ....
0x0080: 0000 0000 0000 0000 0102 0229 c220 3174 ....)..1t
0x0090: a5f2 d242 f6ac ef8a ...B....

21:41:00.354032 IP6 (hlim 64, next-header ESP (50) payload length: 112)
2001:470:e23d:1dfa:601:7504:5cc7:37e8 > 2001:470:e23d:e000::1000:
ESP(spi=0x00000013,seq=0x599), length 112
0x0000: 6000 0000 0070 3240 2001 0470 e23d 1dfa `....p2@...p.=..
0x0010: 0601 7504 5cc7 37e8 2001 0470 e23d e000 ..u.\.7....p.=..
0x0020: 0000 0000 0000 1000 0000 0013 0000 0599 ....
0x0030: 6000 0000 0030 1140 2001 0470 e23d e000 `....0.@...p.=..
0x0040: 0000 0000 0000 0601 2001 0470 e23d e012 ....p.=..
0x0050: 0000 0000 0000 0012 0007 b8b6 0030 726b 0rk
0x0060: 0000 00e1 0000 0000 0000 0000 0000 0000 ....
0x0070: 0000 0000 0000 0000 0000 0000 0000 0000 ....
0x0080: 0000 0000 0000 0000 0102 0229 b4f5 708e ....)..p.
0x0090: 2c72 3146 77cf bba8 ,r1Fw...

Purple = Outer IPv6 Header (40 bytes)
Blue = Original IPv6 Header (40 bytes)
Orange = IPsec ESP Header (8 bytes)
Red = UDP Header (8 bytes)
Green = Payload (40 bytes)
Orange = IPsec ESP Trailer (16 bytes)

Access Router (eth2):

21:41:00.002439 IP (tos 0x0, ttl 64, id 10123, offset 0, flags [DF], proto TCP (6), length 162)
10.0.6.5.40522 > 10.0.6.200.4507: Flags [P.], cksum 0x2161 (incorrect -> 0x6a81), seq
188601:188723, ack 201653, win 198, length 122
0x0000: 4500 00a2 278b 4000 4006 f1fe 0a00 0605 E...'.@.@.....
0x0010: 0a00 06c8 9e4a 119b 4637 ff06 5ecf 22bf .....J..F7..^."
0x0020: 5018 00c6 2161 0000 0300 007a 0000 0001 P...!a.....z....
0x0030: 0000 0000 0000 006a e24b 6000 0000 0030 ....j.K`....0
0x0040: 113f 2001 0470 e23d e012 0000 0000 0000 .?...p.=........
0x0050: 0012 2001 0470 e23d f601 0000 0000 0000 .....p.=........
0x0060: 0001 b8b6 0007 0030 626a 0000 00e1 0000 ....0bj......
0x0070: 0000 0000 0000 0000 0000 0000 0000 0000 ....
0x0080: 0000 0000 0000 0000 0000 0000 0000 0000 ....
0x0090: 0000 0102 0229 c220 3174 a5f2 d242 f6ac .....)..1t...B..
0x00a0: ef8a ..

21:41:00.353811 IP (tos 0x0, ttl 64, id 3823, offset 0, flags [DF], proto TCP (6), length 162)
10.0.6.200.4507 > 10.0.6.5.40522: Flags [P.], cksum 0xfb0e (correct), seq 201676:201798, ack
188723, win 2800, length 122
0x0000: 4500 00a2 0eef 4000 4006 0a9b 0a00 06c8 E.....@.@.....
0x0010: 0a00 0605 119b 9e4a 5ecf 22d6 4637 ff80 ....J^."F7..

0x0020: 5018 0af0 fb0e 0000 0301 007a 0000 0002 P.....z....
0x0030: 0000 0000 0000 006a e24d 6000 0000 0030 .....j.M`....0
0x0040: 1140 2001 0470 e23d e000 0000 0000 0000 .@...p.=.....
0x0050: 0601 2001 0470 e23d e012 0000 0000 0000 .....p.=.....
0x0060: 0012 0007 b8b6 0030 726b 0000 00e1 0000 .....0rk.....
0x0070: 0000 0000 0000 0000 0000 0000 0000 0000 .....
0x0080: 0000 0000 0000 0000 0000 0000 0000 0000 .....
0x0090: 0000 0102 0229 b4f5 708e 2c72 3146 77cf .....)..p.,r1Fw.
0x00a0: bba8 ..

Yellow = IPv4 Header (20 bytes) – Not transmitted over RF
Light Green = TCP Header (20 bytes) – Not transmitted over RF
Dark Red = Rockwell Radio Interface Header (16 bytes) - Not transmitted over RF
Dark Blue = ROHC Header (2 bytes)
Blue = Original IPv6 Header (40 bytes)
Red = UDP Header (8 bytes)
Green = Payload (40 bytes)
Orange = IPsec ESP Trailer (16 bytes)

Mobile Router (eth2):

21:41:00.160844 IP (tos 0x0, ttl 64, id 54545, offset 0, flags [DF], proto TCP (6), length 162)
10.0.2.200.4507 > 10.0.2.6.52115: Flags [P.], cksum 0xa3c2 (correct), seq 247567:247689, ack 174031, win 2800, length 122
0x0000: 4500 00a2 d511 4000 4006 4c77 0a00 02c8 E.....@.@.Lw....
0x0010: 0a00 0206 119b cb93 177b 54c8 7487 7d4b .....{T.t.}K
0x0020: 5018 0af0 a3c2 0000 0301 007a 0000 0001 P.....z....
0x0030: 0000 0000 0000 006a e24b 6000 0000 0030 .....j.K`....0
0x0040: 113f 2001 0470 e23d e012 0000 0000 0000 .?...p.=.....
0x0050: 0012 2001 0470 e23d f601 0000 0000 0000 .....p.=.....
0x0060: 0001 b8b6 0007 0030 626a 0000 00e1 0000 .....0bj.....
0x0070: 0000 0000 0000 0000 0000 0000 0000 0000 .....
0x0080: 0000 0000 0000 0000 0000 0000 0000 0000 .....
0x0090: 0000 0102 0229 c220 3174 a5f2 d242 f6ac .....)..1t...B..
0x00a0: ef8a ..

21:41:00.163083 IP (tos 0x0, ttl 64, id 51019, offset 0, flags [DF], proto TCP (6), length 162)
10.0.2.6.52115 > 10.0.2.200.4507: Flags [P.], cksum 0x1962 (incorrect -> 0x45eb), seq 174031:174153, ack 247689, win 920, length 122
0x0000: 4500 00a2 c74b 4000 4006 5a3d 0a00 0206 E....K@.@.Z=....
0x0010: 0a00 02c8 cb93 119b 7487 7d4b 177b 5542 .....t.}K.{UB
0x0020: 5018 0398 1962 0000 0300 007a 0000 0002 P....b.....z....
0x0030: 0000 0000 0000 006a e24d 6000 0000 0030 .....j.M`....0
0x0040: 1140 2001 0470 e23d e000 0000 0000 0000 .@...p.=.....
0x0050: 0601 2001 0470 e23d e012 0000 0000 0000 .....p.=.....
0x0060: 0012 0007 b8b6 0030 726b 0000 00e1 0000 .....0rk.....
0x0070: 0000 0000 0000 0000 0000 0000 0000 0000 .....
0x0080: 0000 0000 0000 0000 0000 0000 0000 0000 .....
0x0090: 0000 0102 0229 b4f5 708e 2c72 3146 77cf .....)..p.,r1Fw.
0x00a0: bba8 ..

Yellow = IPv4 Header (20 bytes) – Not transmitted over RF
Light Green = TCP Header (20 bytes) – Not transmitted over RF
Dark Red = Rockwell Radio Interface Header (16 bytes) - Not transmitted over RF
Dark Blue = ROHC Header (2 bytes)
Blue = Original IPv6 Header (40 bytes)
Red = UDP Header (8 bytes)
Green = Payload (40 bytes)
Orange = IPsec ESP Trailer (16 bytes)

Mobile Router (rftun0):

21:41:00.161078 IP6 (hlim 63, next-header ESP (50) payload length: 112) 2001:470:e23d:e000::1000 > 2001:470:e23d:1dfa:601:7504:5cc7:37e8: ESP(spi=0x00000014,seq=0x599), length 112

```
        0x0000:  6000 0000 0070 323f 2001 0470 e23d e000  `....p2?...p.=..
        0x0010:  0000 0000 0000 1000 2001 0470 e23d 1dfa  ..........p.=..
        0x0020:  0601 7504 5cc7 37e8 0000 0014 0000 0599  ..u.\.7.........
        0x0030:  6000 0000 0030 113f 2001 0470 e23d e012  `....0.?...p.=..
        0x0040:  0000 0000 0000 0012 2001 0470 e23d f601  ..........p.=..
        0x0050:  0000 0000 0000 0001 b8b6 0007 0030 626a  .............0bj
        0x0060:  0000 00e1 0000 0000 0000 0000 0000 0000  ................
        0x0070:  0000 0000 0000 0000 0000 0000 0000 0000  ................
        0x0080:  0000 0000 0000 0000 0102 0229 c220 3174  ...........)..1t
        0x0090:  a5f2 d242 f6ac ef8a                      ...B....
```

21:41:00.161296 IP6 (hlim 64, next-header ESP (50) payload length: 112) 2001:470:e23d:1dfa:601:7504:5cc7:37e8 > 2001:470:e23d:e000::1000: ESP(spi=0x00000013,seq=0x599), length 112

```
        0x0000:  6000 0000 0070 3240 2001 0470 e23d 1dfa  `....p2@...p.=..
        0x0010:  0601 7504 5cc7 37e8 2001 0470 e23d e000  ..u.\.7....p.=..
        0x0020:  0000 0000 0000 1000 0000 0013 0000 0599  ................
        0x0030:  6000 0000 0030 1140 2001 0470 e23d e000  `....0.@...p.=..
        0x0040:  0000 0000 0000 0601 2001 0470 e23d e012  ..........p.=..
        0x0050:  0000 0000 0000 0012 0007 b8b6 0030 726b  .............0rk
        0x0060:  0000 00e1 0000 0000 0000 0000 0000 0000  ................
        0x0070:  0000 0000 0000 0000 0000 0000 0000 0000  ................
        0x0080:  0000 0000 0000 0000 0102 0229 b4f5 708e  ...........)..p.
        0x0090:  2c72 3146 77cf bba8                      ,r1Fw...
```

Purple = Outer IPv6 Header (40 bytes)
Blue = Original IPv6 Header (40 bytes)
Orange = IPsec ESP Header (8 bytes)
Red = UDP Header (8 bytes)
Green = Payload (40 bytes)
Orange = IPsec ESP Trailer (16 bytes)

References

1. C. Perkins, Ed., Johnson, D., and J. Arkko, “Mobility Support in IPv6”, RFC 6275, July 2011. [Online]. Available: http://www.rfceditor.org/rfc/rfc6275.txt
2. Devarapalli, V., Wakikawa, R., Petrescu, A., and P. Thubert, “Network Mobility (NEMO) Basic Support Protocol”, RFC 3963, January 2005. [Online]. Available: http://www.rfceditor.org/rfc/rfc3963.txt
3. UMIP Development Team (2014), “UMIP”, Computer Software, http://www.umip.org
4. Steffen, A., “strongSwan”, Computer Software, http://www.strongswan.org
5. ROHC Library Development Team (2014), “ROHC Library”, Computer Software, http://rohc-lib.org