# Deployment, Management, & Operations of Internet Routers for Space-Based Communication

A thesis presented to
the faculty of
the Scripps College of Communication of Ohio University

In partial fulfillment
of the requirements for the degree
Master of Information and Telecommunication Systems

Zack A. Sims
May 2015

This thesis titled

Deployment, Management, & Operations of Internet Routers for Space-Based Communication

by

ZACK A. SIMS

has been approved for
the J. Warren McClure School of Information and Telecommunication Systems
and the Scripps College of Communication by

Hans Kruse
Professor of Information and Telecommunication Systems

Scott Titsworth
Dean, Scripps College of Communication

# ABSTRACT

SIMS, ZACK A., M.I.T.S., May 2015, Information and Telecommunication Systems

Deployment, Management, & Operations of Internet Routers for Space-Based Communication

Director of Thesis: Hans Kruse

This thesis addresses certain technical and financial challenges associated with the deployment and operation of relay spacecraft using the Internet Protocol as the primary routing protocol. Though IP in space has been a hot topic for nearly a decade, few studies address the capabilities of management protocols being used to operate a geostationary fleet. Likewise, few have addressed the real-world cost structure of replacing a traditional bent-pipe fleet with an IP-enabled fleet. Within our research, we investigate whether SNMP, TFTP, and SCP are capable of meeting the Tracking, Telemetry, and Command requirements set by a real-world geostationary relay service provider. We also investigate the driving forces of relay deployment and operational costs, identify Rough Order of Magnitude costs for a geostationary IP-enabled relay, and define a financial profile categorizing the costs of replacing a bent-pipe fleet with an IP-enabled fleet.

# ACKNOWLEDGEMENTS

When I first started this study, I had little idea of the challenges that lay ahead. The project scope was too large, my research questions were weak, and the industry was highly sensitive. However, my greatest obstacle was filling a knowledge gap larger than I thought was possible. Thankfully, with the guidance and support of great minds, wonderful mentors, and life-long friends; I was able to carve out achievable goals, frame them with solid questions, and gain access to the knowledge required.

First and most of all, Id like to thank my mentors: Wes Eddy and Mike Coriell, who have continued working with me over the years without expecting anything in return. Despite their busy schedules, neither has ever turned me away and both have always provided valuable insight. I dont believe Ive met anyone else so willing to add value to others as they have been to me.

Second, the faculty of my Graduate and Undergraduate studies have my thanks for the discipline and knowledge they helped me obtain. If not for their hard work, Id be half the person I am today. Specifically, Id like to acknowledge a few key individuals: Dovel Meyers, Janice Johnson, Hans Kruse, Jim Reneau, Shawn Ostermann, and Philip Campbell; in many ways, each of them have gone the extra-mile in my personal development. I truly appreciate the effort!

Lastly, I want to thank my family who tolerated my late nights of study and who cheered me on throughout every success: Lydia, my loving wife, who never gave up supporting me; Ron and Cheryl, my parents, who have always been there without hesitation; Zane, my brother, who has been a strong sounding board and supporter; and all of my close friends who I would not hesitate to call brothers and sister. Their love and friendship has been the foundation to all of my successes.

# Table of Contents

# LIST OF TABLES

# LIST OF FIGURES

# LIST OF ACRONYMS

| | |
|---|---|
| AQM | Active Queue Management |
| ASIC | Application Specific Integrated Circuit |
| BER | Bit Error Rate |
| BP | Bundle Protocol |
| CCSDS | Consultative Committee for Space Data Systems |
| CoS | Class of Service |
| COTS | Commercial Off The Shelf |
| CPU | Central Processing Unit |
| dB | Decibels |
| DTN | Delay (or Disruption) Tolerant Networking |
| Eb/No | Energy-per-bit to Noise ratio |
| GRGT | Guam Remote Ground Terminal |
| GS | General Schedule |
| GSFC | Goddard Space Flight Center |
| HDLC | High-Level Data Link Control |
| IETF | Internet Engineering Task Force |
| IP | Internet Protocol |
| IPv4 | Internet Protocol version 4 |
| IPv6 | Internet Protocol version 6 |
| IPSec | Internet Protocol Security |

| | |
|---|---|
| IRTF | Internet Research Task Force |
| ISO | International Organization for Standardization |
| ISP | Internet Service Provider |
| LAN | Local Area Network |
| LEO | Low Earth Orbit |
| LTP | Lichlider Transport Protocol |
| MA | Multiple Access |
| MER | Message Error Rate |
| MIB | Management Information Base |
| MIC | Modem Interface Card |
| NASA | National Aeronautics and Space Administration |
| NCTS | Naval Computer and Telecommunications Station |
| NISN | NASA Integrated Services Network |
| OS | Operating System |
| OSI | Open Systems Interconnection |
| PDU | Protocol Data Units |
| PER | Packet Error Rate |
| PPBE | Planning, Programming, Budgeting, and Execution |
| QoS | Quality of Service |
| RCP | Remote Copy |
| RF | Radio Frequency |

| | |
|---|---|
| RRQ | Read Request |
| RTO | Retransmission Timeout |
| RTOS | Real-Time Operating System |
| RTT | Round Trip Time |
| SA | Single Access |
| SCaN | Space Communication and Navigation |
| SCP | Secure Copy |
| SDR | Software Defined Radio |
| SI | Space Internetworking |
| SLE | Space Link Extension |
| SN | Space Network |
| SNMP | Simple Network Management Protocol |
| SNMPv3 | Simple Network Management Protocol version 3 |
| SNUG | Space Network User's Guide |
| SSI | Solar System Internetworking |
| STGT | Second TDRS Ground Terminal |
| TCP | Transmission Control Protocol |
| TDRS | Tracking and Data Relay Satellite |
| TDRSS | Tracking and Data Relay Satellite System |
| TFTP | Trivial File Transfer Protocol |
| TRL | Technology Readiness Level |

| | |
|---|---|
| TT&C | Tracking, Telemetry, and Command |
| UDP | User Datagram Protocol |
| USN | Universal Space Network |
| WAN | Wide Area Network |
| WRQ | Write Request |
| WSGT | White Sands Ground Terminal |

# 1 INTRODUCTION

Previously, communication to, from, and across space links has been characterized by limited automation, planned circuit-switched connections, inefficient bandwidth utilization, and unintelligent relay mechanisms. The limitations that accompany these characteristics have been viewed as unavoidable for a long time. However, there are researchers, such as National Aeronautics and Space Administration (NASA) engineers Scott Burleigh, Dave Isreal, Keith Hogie, and more (Isreal & Spinolo, 2004) (Jackson, 2005) (Jones, 2009) (Wyatt, Burleigh, Jones, Torgerson, & Wissler, 2009), who believe the use of internetworking technologies can eliminate certain technical barriers in the field of space communication. In recent years, experiments, demonstrations, and operational deployments of internetworking technologies aboard spacecraft have provided evidence to support Space Internetworking's (SI) usefulness (Hogie, Criscuolo, & Parise, 2005) (Isreal & Zillig, 2002) (Welch, Brooks, Beering, Hoder, & Zernic, 1997). However, operational and experimental studies of SI have failed to demonstrate certain technical and business aspects regarding the usability, manageability, and financial constraints involved with deploying SI in operational environments. Within this thesis, we hope to facilitate the advancement of SI by investigating the challenges associated with the deployment and operation of IP-enabled geostationary spacecraft. More specifically, our research (1) models and tests whether current spacecraft Tracking, Telemetry, and Command (TT&C) technologies can be replaced by commonly used IP management protocols within a geostationary relay service provider's network and (2) defines how much it would cost a geostationary relay service provider to replace their bent-pipe fleet with an IP-enabled fleet using phased integration.

## 1.1 History of Space Comm.

You can trace Space Communication's roots back to the days when Radio Frequency (RF) signals were reflected off the Earth's ionosphere. By doing this, early communication devices were able to transmit signals to remote locations using the atmosphere's natural density as a signal relay mechanism. However, use of the ionosphere is naturally constrained by transmission distance, signal strength, weather conditions, and frequency band ranges (Mead, 1999). This practice eventually led to the use of artificial orbital satellites during the 1960's. Early satellites are known as passive satellites, or satellites that were designed specifically to bounce signals between two remote locations. Passive satellites work better than the ionosphere, but still have similar limitations. The next generation of relay evolution came with the advent of active satellites, or satellites that receive incoming signals and 'act' on them before relaying. Though early satellites were passive, the need for more reliable signals led to the proliferation of active satellites. Today, all relay spacecraft in use are active satellites (Mead, 1999).

As history has shown time and again, authors and dreamers often emerge as visionaries for many fields of study. Space communication is no exception to this trend. In 1945, twelve years before the Russians launched Sputnik (the first satellite to reach orbital heights), Arthur C. Clarke wrote fiction discussing the concept of geostationary relays. Clarke later referred to these satellites as EXTRA-TERRESTRIAL RELAYS (Clark, 1961). Though Clarke's vision laid the foundation for geo telecommunications supporting global coverage, the earliest satellites were launched in Low Earth Orbit (LEO) (Mead, 1999). LEO configurations require relatively large fleets to support global coverage, whereas geo configurations require a minimum of three spacecraft. Since launch costs and ground station complexity increase with fleet size, only a limited number of LEO deployments were ever made. A prime example is AT&T's 1962 launch of Telstar, the first active communication satellite. Though it helped further satellite research, Telstar didn't make it very far as a practical tool (Mead, 1999).

Different orbits began to emerge throughout the 1960's, including geosynchronous in 1963, and later geostationary. Few believed that Clarke's vision would emerge as early as it did (Mead, 1999). SYNCOM 3 became the first satellite to reach geostationary orbit (preceded by SYNCOM 2 which was the first geosynchronous satellite). In April 1965, INTELSAT 1 was launched as the first commercial communications satellite. INTELSAT became the first organization to support global coverage using a geostationary fleet. This commercial satellite paradigm is comparable to NASA's Space Network (SN) which is the model used in this thesis. NASA's first generation Tracking and Data Relay Satellite (TDRS) was launched in 1983. Eight additional TDRS have been released since then and allow the SN to provide global relay coverage.

### 1.2 Satellite Relay as a Service

Since our research models a relay service provider, we feel it's appropriate to define exactly what one does. A relay service provider is any organization that provides relay services through spacecraft infrastructure. Relay services allow a customer to transmit communication signals to remote locations that would otherwise be unreachable. In the past, some relays and their ground stations were owned and operated by the users themselves (Mead, 1999). However, in recent years, the trend has been for a stand-alone entity to lease its infrastructure to end-users. This trend is largely due to the cost and risk associated with managing and deploying spacecraft (Mead, 1999). Similar to other Wide Area Networks (WAN), the equipment and resources (i.e. ground stations and relay constellations) are very expensive. Since few users require full and continuous access to a relay network, it's impractical for those users to own the infrastructure.

Relay service providers are akin to Internet Service Providers (ISPs). Similar to the way ISPs offer infrastructure access as a service, relay service providers offer spacecraft services to their customers. Whether a service provider is in the private or public sector hardly affects the way it functions; they even abide the same rules set by standards bodies (*Radiocommunication Sector (ITU-R)*, 2011). NASA's Space Communication and Navigation (SCaN) program (*Space Communication and Navigation Home Page*, n.d.) is a perfect example. The SN, one of SCaN's network elements, provides relay services to customers across the globe (*Space Segment*, n.d.). These customers include: NASA programs, US agencies, foreign agencies, and commercial entities (NASA, Exploration and Space Communications Projects Division 450, 2007b). SN services include: "Telecommunications, Tracking, Testing, Analysis, and Data Distribution/Processing Services" (NASA, Exploration and Space Communications Projects Division 450, 2007b). Commercial relay providers include entities like the Universal Space Network (USN) (*Satellite Communication*, n.d.) and INMARSAT (*Services - INMARSAT*, n.d.). With these providers, service catalogs are similar to SCaN's. Typically, customers transfer signals between their Operations Center and remote platform through the provider's network. Customer platform types may include: LEO spacecraft, maritime vessels, aerial vehicles, military devices, and more.

Relay provision depends heavily on a spacecraft's line of sight, frequency bands, link schedules, and protocol stacks (Fortescue, Swinerd, & Stark, 2011). Line of sight and frequency band support are mostly static properties while scheduling and protocol stacks depend on the provider's platform and customer requirements. Static properties are directly influenced by spacecraft engineering decisions and by the physical properties of the universe. Frequency band support is integral to the circuitry aboard the spacecraft and is typically decided based on cost and/or regulatory policies (Fortescue et al., 2011). Once a spacecraft has entered orbit, it's impossible to rewire it. However the use of Software Defined Radios (SDRs) have allowed for some flexibility in upgrading certain 'wired' features. Line of sight, on the other hand, is a limitation of RF communication and requires that the customer platform and provider ground terminal are both within view of the relay. Link scheduling is entirely a policy-based limitation. It's the process of allocating time slots where communication channels are reserved for specific customers. This task can be simple or complex, depending on the predictability and urgency of customer needs. For some providers, link scheduling can vary over time and is an ongoing operations task.

Protocol stacks refer to the set of techniques supported by a platform that allow it to communicate with other platforms. Protocols are often categorized according to a layered network model to support abstraction. In international telecommunications, the International Organization for Standardization's (ISO) Open Systems Interconnection (OSI) model (*International Organization for Standardization*, n.d.) has become a de facto approach to protocol abstraction. The first layer, the physical layer, is primarily responsible for the relationship between a transmitter or receiver and the transmission medium (e.g. radios, amplifiers, antennas, etc. become associated with RF channels). When relays operate at this layer they are referred to as bent-pipe relays. The sole responsibility of a bent-pipe relay is the interception and redirection of customer signals via up-conversion and down-conversion. SCaN's TDRS System (TDRSS) currently offers bent-pipe services, however articles and design documents suggest that future generations may implement SDRs and enhanced protocol stacks (Isreal & Spinolo, 2004) (IOAG, Space Internetworking Strategy Group (SISG), 2010a) (NASA, Space Communications and Navicagion (SCAN), TBD 2011) (NASA, Space Communications and Navicagion (SCAN), 2010).

# 2 SPACE INTERNETWORKING

For a relay to support SI services, it must implement a minimum of OSI layers 1 through 3. Layer 2, the data link layer, builds on the properties set by the physical layer. Its purpose is to govern interactions between two or more devices on a communication channel; establishing the concept of a network. Layer 2 protocols have the capability to add or enhance certain end-to-end requirements, such as: time-synchronization, spectrum utilization, signal multiplexing, Quality/Class of Service (QoS, CoS), and security (Bhasin & Hayden, 2004) (Eddy, n.d.). In order for today's TDRS models to support onboard Layer 2 services, they first require SDRs (which bring the benefit of cross-link radio flexibility) and modems (which support modulation/demodulation of raw RF signals into coded data). SCaN is currently planning for SDRs and other next generation equipment for enhanced services (NASA, Space Communications and Navicagion (SCAN), 2010) (Schier, Rush, Williams, & Vrotsos, 2005).

Protocols become less tangible as you move up the stack. Internetworking emerges at layer 3 (the network layer) where the primary goal is to support end-to-end communication between devices in distinctly different networks. The Internet Protocol (IP), for instance, establishes a logical, hierarchical addressing scheme which allows networks to exchange data at central hubs known as routers. Routers have at least one interface to each of their networks through which they ”route” incoming data according to the rules in their routing table. Routing table rules are divided into two categories: static rules and dynamic rules. Static rules are manually configured by the network operator while dynamic rules are automatically generated according to information collected using routing protocols. There are advantages and disadvantages to each type, however their role in space communication is beyond the scope of our study.

Information and telecommunication systems come with considerations beyond the equipment, channels, and addresses used. Security, for instance, is especially useful where data is sensitive or mission critical. However, integrity, availability, and authentication can be implemented in many ways and at multiple layers of the protocol stack. For example, IP encryption is done two different ways, depending on the protocol version. IP version 4 (IPv4) and IP version 6 (IPv6) both support encryption. However, IPv6 was designed with encryption in its specification whereas IPv4 requires the Internet Protocol Security (IPSec) extension. Encryption is sometimes done at other layers as well; even at the physical layer! In fact, NASA has used layer 1 encryption on spacecraft signals for many years (Eddy & Fuentes, n.d.).

Data integrity and reliability features are sometimes addressed at layer 4 (the transport layer). This layer handles the delivery of data to upper-layer services after it reaches its destination. Transport layer reliability often handles issues like: flow control, segmentation, and error detection. These features are intended to improve network performance and the overall experience of its users. However, since protocols are designed to be used in specific conditions, they're prone to complications when introduced to new environments. Since the IP stack was designed for terrestrial communication, some protocols experience undesirable limitations in space communication. The Transmission Control Protocol (TCP), for instance, is a time-sensitive protocol designed to guarantee end-to-end message delivery. Unfortunately, its time sensitivity results in a performance decay as delivery times increase (Wood, Peoples, Parr, Scotney, & Moore, 2007).

Layers 4 through 7 deal with application-specific features; meaning IP relay services end at layer 3. However, if the provider's platform is operated using IP (like in our research model), all layers of the OSI model must be implemented. For this study, the next relevant layer is layer 7 (the application layer) because this is where operational data is processed (Hogie et al., 2005). The application layer defines the specifics of how applications interpret incoming data (whether it treats it as operational instructions, remote system diagnostics, confirmation of delivery, etc.) and what the application does after receiving it.

Operational data can be defined as any coded signal delivered to or generated by the platform in order to assess or control its state or the state of its subsystems. Operations are divided into three categories: Tracking, Telemetry, and Command. Tracking is a service performed by the provider whereby return signals are used to determine the location and attitude of a spacecraft. This operational task requires complex mathematical operations and is not actually part of a protocol. Telemetry is diagnostic data generated by payloads or subsystems that update the provider with the state of onboard devices. This is an onboard application that acts as a continuous solicitation service to the provider. In contrast, Command data is sent from the provider's Operations Center when one or more aspects of the platform need controlled. Once the Command request is received, an acknowledgment is sent back to the Operations Center.

## 2.1 Ground Station IP Support

A provider's infrastructure is split into two parts: space segment and ground segment. Though our study focuses on the space segment, it's important to discuss how the ground segment works for end-to-end service provision. Figure 2.1 shows a diagram depicting how the infrastructure works for customer services and relay operations. At the core of this diagram, we find the provider's ground station. This is where all customer forward and return traffic is processed and rendered for service. Processing can include Tracking services, protocol transformations, and return data queuing (NASA, Exploration and Space Communications Projects Division 450, 2007b) (Eddy, n.d.). The ground station is also home to the provider's Operations Center where management and operations take place for the relay platforms.

An IP backbone acts as the interface between the customer Operations Centers and the ground station. For the SN, this backbone is known as the NASA Integrated Services Network (NISN) (NASA, Exploration and Space Communications Projects Division 450, 2007b). It's possible to offer services without using IP in the backbone (e.g. using point-to-point leased lines) and doing so has no impact on whether the ground station supports IP. As long as it supports IP routing from the ground terminals to the point of customer interconnection, the ground station qualifies as IP-enabled. Currently, the SN ground station is not IP-enabled, however modifications have been taking place to provide

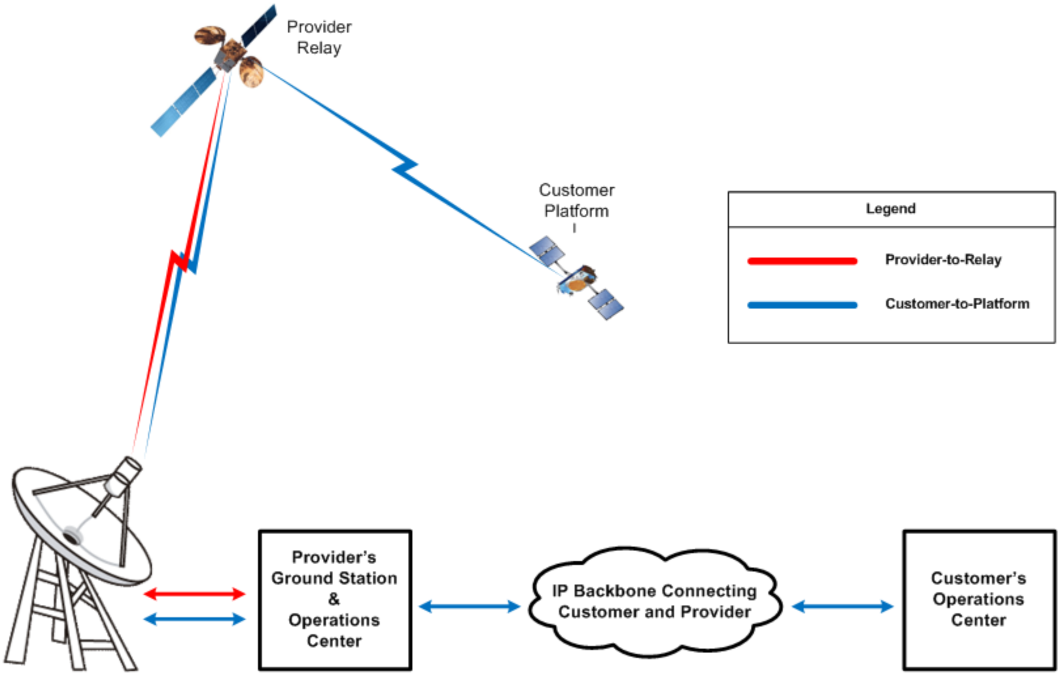

Figure 2.1: *End-to-End Relay Services and Operations*

Illustration depicting the end-to-end paradigm used by relay service providers and their customers. The customer communicates with its remote platform (in this case a spacecraft) by leasing access to the provider's infrastructure.

IP ground support (*Tracking and Data Relay Satellite: Continuing the Critical Lifeline*, n.d.) (NASA, Space Communications and Navicagion (SCAN), 2010) (Isreal & Spinolo, 2004).

A customer's layer 2 protocols can be handled at one of two locations: the customer's Operations Center or the provider's ground station. In order for ground stations to be fully IP-enabled however, the ground station must be solely responsible for managing layer 2 metadata (frames); otherwise the customer's IP metadata (headers) becomes hidden. Having access to IP headers allows the ground station to make intelligent routing decisions. However, there are additional benefits to framing at the ground station. Currently, SN customers form frames at their Operations Center and transmit them to the provider using a technique known as tunneling. In networking, tunneling is the process of piggy-backing one protocol on top of another from the same, or higher layer. In other words, SCaN customers transmit layer 2 data on top of layer 3 across the NISN. Sometimes, for security reasons, tunneling is desirable. However, tunneling comes with performance sacrifices and should therefore be avoided if possible.

Tunneling frames over IP can cause more problems than performance decay. However, this depends heavily on the mechanisms used in the IP backbone. If the Protocol Data Units (PDU) (defined chunks of data prefixed or postfixed by protocol metadata; e.g. layer 2 frames and IP packets) of these mechanisms are smaller than the customer's frames, a process known as fragmentation is required. Fragmentation is the process of splitting PDUs into smaller chunks that are combined at a later point in the network. One complication comes from a reliance on time-synchronization. Many layer 2 protocols, such as High-Level Data Linc Control (HDLC) (Rash, Hogie, Criscuolo, & Parise, 2003), require continuous or predictable signaling to maintain their channels. To minimize tunneling errors, protocols like the Consultative Committee for Space Data Systems'

(CCSDS) Space Link Extension (SLE) (*Space Link Extension (SLE) Data Transfer Services Protocol and Application Program Interface (API) Specifications*, n.d.) have been crated. SCaN currently uses SLE to manage partial layer 2 responsibilities of tunneled customer frames.

## 2.2 IP-Enabled Relays

Though spacecraft with onboard SDRs and modems have advantages over traditional bent-pipe relays, further benefits emerge with the addition of routers and IP stacks. Within this thesis, we define an IP-enabled relay as any spacecraft that implements these features and is capable of providing IP relay and telecommunications services. Routers can be implemented as hardware or software and each method comes with strengths and weaknesses. The IP stack, however, should be implemented as part of the Operating System (OS) software that drives spacecraft subsystems.

Spacecraft hardware is split into two categories: buses and payloads. The bus is a combination of the spacecraft structure and the subsystems that support operations. These subsystems typically include: power; attitude and orbit control; Telemetry & Command; thermal; data handling; and propulsion (Fortescue et al., 2011). Many bus architectures have been developed; mostly to compensate for the unique needs of individual missions (Fortescue et al., 2011). Our TDRS model, for instance, is built from the Boeing 601 bus (*Tracking and Data Relay Satellite: Continuing the Critical Lifeline*, n.d.).

Buses act as a platform for payloads. Payloads are special-purpose hardware, usually designed to satisfy mission-specific goals. Since a relay's primary mission is communication, we feel IP-enabled relays should use a router payload instead of a software router. Experiments and demonstrations of router payloads include: Cisco IRIS (aboard Intelsat-14) (*Cisco 18400 Space Router*, 2010), CLEO (aboard UK-DMC) (Wood et al., 2008), CANDOS (aboard Columbia) (Isreal & Zillig, 2002), and OCA (aboard ISS)

(NASA, International Space Station (ISS), 2000). In relation to our research model, Cisco IRIS rises above the rest, literally. It's the only hardware router that has flown in geostationary orbit and is the only Commercial Off The Shelf (COTS) router proven to be capable of doing so.

Non-relay spacecraft may benefit from software routers, therefore they shouldn't be discounted from SI. Software routers have nominal impacts on the mass, power, and thermal load of a spacecraft (which contrasts the impact of adding payloads). For a relay, however, the benefits of dedicated communication hardware outweigh the costs. Subsystems share computational resources and typically perform less reliably and efficiently than payloads (Fortescue et al., 2011). This is largely because payloads process

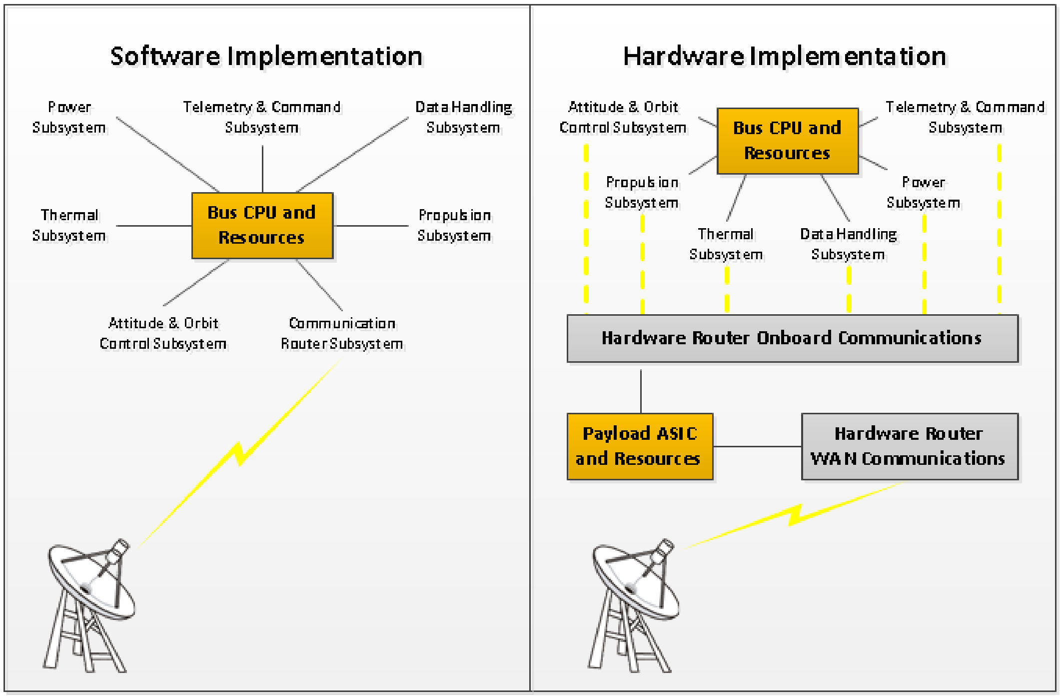

Figure 2.2: *Space Router Implementation Types*

Illustration comparing the two primary router implementations: software routers vs hardware routers. Software routers are installed as bus sub-systems on shared hardware while hardware routers are physical payloads with dedicated hardware.

data using Application Specific Integrated Circuits (ASIC) whereas subsystems use a general-purpose Central Processing Unit (CPU). Figure 2.2 illustrates the difference between software router subsystems and physical router payloads.

Researchers often describe SI spacecraft as a node on a network (Jackson, 2005) (Joseph & Lazbin, 2004), however onboard stacks cause subsystems and payloads to become addressable; making the spacecraft a network of nodes (Eddy, n.d.). Figure 2.3 illustrates how the IP-enabled relay uses its router to internetwork with the ground station and the customer platform. In an ideal, end-to-end, IP relay network, forward data IP headers would be used to determine: how packets arrive at the ground station; which ground terminal the packets pass through; which relay antenna is the best route to the destination platform and; which destination subsystem or payload receives the data. Likewise, in an ideal system, return data should arrive at the customer's Operations Center using a single destination IP address.

Though the base requirement for an IP-enabled relay is simple, an ideal system like the one described above would require the design of new and enhanced technology (NASA, Space Communications and Navicagion (SCAN), 2010). For example, before opportunistic, cross-link communication can be achieved, improved phased array antennas need to emerge (Eddy, n.d.). Also, prior to being used in mission critical operations, new technology must reach a certain Technology Readiness Levels (TRL) (Fortescue et al., 2011) through simulations, demonstrations, experiments, and other benchmarking facilities.

As mentioned earlier, communication through traditional spacecraft architectures requires operational instructions to be sent in advance. Addresses for these architectures are flat (not hierarchical) and are used only to denote a single spacecraft. When this architecture receives operational data, it strips the spacecraft address and uses a secondary

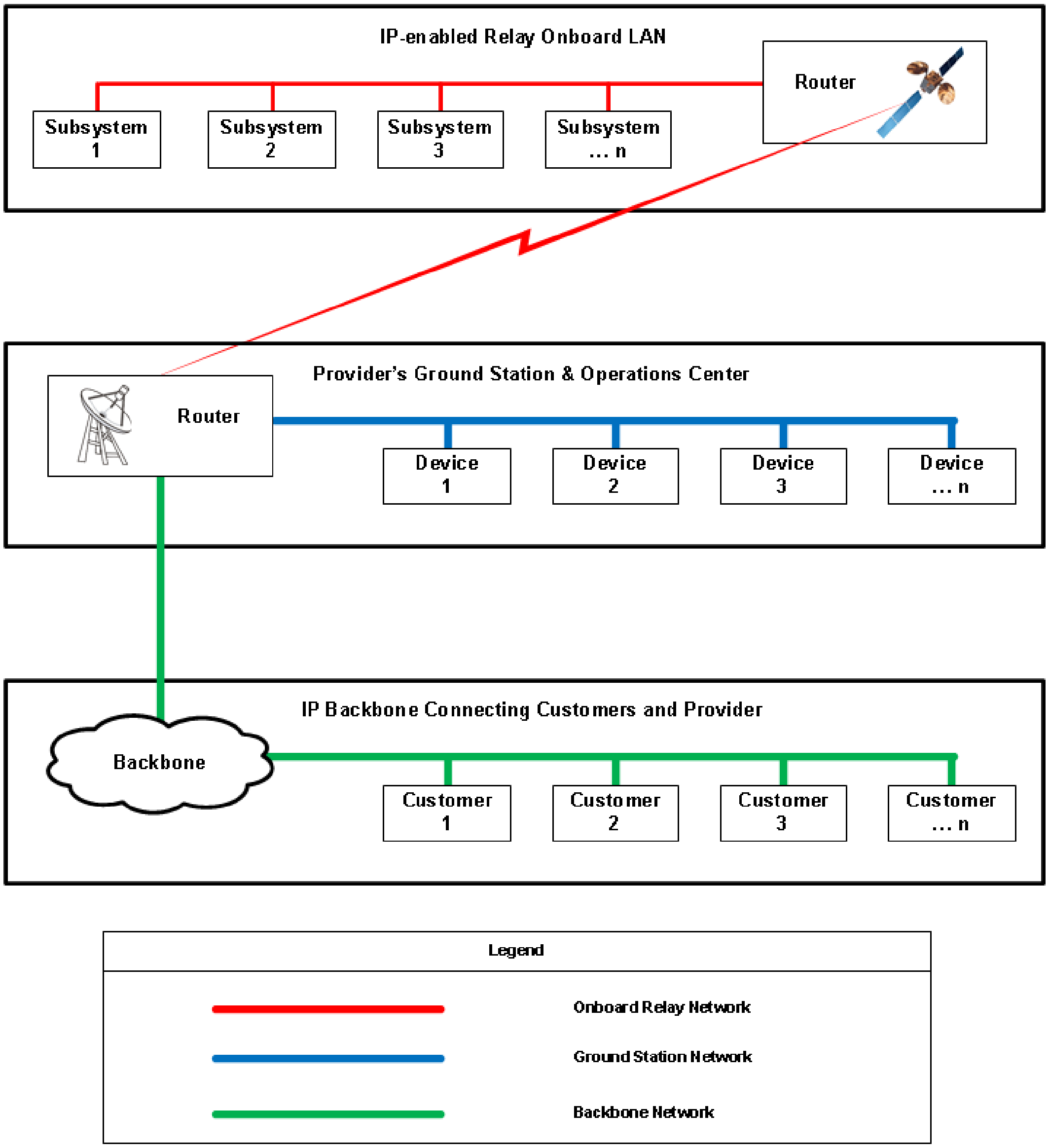

Figure 2.3: *IP-Enabled Relay as a Network*

Graphically shows how bringing IP aboard the spacecraft allows routing at all levels, including at a sub-system and payload level. Traditional spacecraft use a client-server paradigm in sub-system communication whereas IP allows for onboard LANs of bus, ring, or even mesh configurations.

address (the Service ID) to deliver data to the correct subsystem or payload. Onboard data delivery is handled in a master-slave paradigm where every system communicates with a single master system, essentially creating a bottleneck for inter-process communication. In contrast, the global addressability and hierarchical nature of IP allows customers and providers to apply networking, subnetworking, and internetworking concepts to their Operations Centers, ground stations, backbones, fleets, platforms, platform subsystems, and platform payloads for end-to-end communication. Onboard networks also remove the master-slave bottleneck by establishing a concurrent Local Area Network (LAN).

## 2.3 A Brief Look at Solar System Internetworking

As mentioned in the introduction, researchers believe that SI can overcome many limitations of contact scheduling. NASA has even reported that opportunistic, unscheduled contact capabilities are required for future human and robotic exploration (Bhasin & Hayden, 2004) (Spearing et al., 2005). The main challenge has been the creation of infrastructure architectures and designs that support multi-hop transmissions capable of high data rates along delayed and disrupted network links. Disrupted, in this context, refers to a link which has periods where data transfer is impossible. When referring to SI beyond Earth's orbit, the term Solar System Internetworking (SSI) (IOAG, Space Internetworking Strategy Group (SISG), 2010a) is sometimes adopted. SSI is essentially a more complex version of SI; having greater distances, more network nodes, and more complicated end-to-end routing conditions.

Space communication is quite different from its terrestrial counterpart. This holds true for space-, ground-, and cross-link communications. The risk and cost of spacecraft deployment has resulted in slower development of solution which overcome constraints. These constraints include: ”intermittent communication links; highly asymmetric or unidirectional communication links; bit error rates higher than most hardwired links; [spacecraft equipment incapable of supporting] multiple mobile nodes forming a dynamic network topology; [and limited techniques for] maintaining a single address for a spacecraft as it uses different ground stations” (Hogie et al., 2005). Other challenges include: meeting requirements for higher data rates; programmable, adaptable onboard devices; autonomous positioning and navigation; and more (Spearing et al., 2005).

All identified problems require solutions with both hardware and protocol components. Opportunistic, ad-hod networking, for instance, require hardware that supports undirected signal reception and onboard beam-forming; including enhanced phased array antennas, SDRs, routers, etc... Likewise, the same problem requires layer 2 and 3 protocols capable of node detection, route determination, store-and-forward capacity, and appropriate encryption and authentication. Data rates are affected by the transmission hardware (such as today's RF antennas or tomorrow's optical transmitters) as well as protocol specifications, such as: PDU size, multiplexing, fragmentation, flow-control, timers, etc...

Some engineers are looking to design new protocols to replace IP for space use (Jones, 2009) (Wyatt et al., 2009). Both CCSDS and The Internet Research Task Force (IRTF) (*Delay-Tolerant Networking Research Group (DTNRG)*, n.d.) use the term Delay (or Disruption) Tolerant Networking (DTN) to describe an umbrella of research to solve protocol-specific issues with SI/SSI. Two leading protocols have emerged to address single-hop and multi-hop issues. The Lichlider Transport Protocol (LTP) is a point-to-point protocol designed to reliably transmit data from one spacecraft to another while the Bundle Protocol (BP) was designed to handle the end-to-end addressability and store-and-forward components required by a disruption-prone mobile network. While DTN has made some headway, it still has reasons for concern (Wood, Eddy, & Holliday, 2009) (Wood, Holliday, Floreani, & Eddy, 2009). Likewise, IP has not proven to be unusable; though the transport layers driving the applications do have limits (Wood et al., 2007).

# 3 METHOD

Up to this point we've covered the groundwork for our research topic. However, in this section we concretely define: our research questions; our research model; the parameters of our analysis; the methods of data collection and; the methods by which conclusions are drawn. Our research questions are asfollows:

1. Can current spacecraft TT&C requirements be satisfied by commonly used IP management protocols within a geostationary relay service provider's space segment?
2. What are the integration, deployment, and operational costs associated with replacing a geostationary, bent-pipe relay space segment with a fully IP-enabled relay space segment?

Our research is broken into two areas of focus: technical capabilities and financial cost. Consequently we've broken our model into two components. Both components represent an aspect of SCaN's Space Network infrastructure; providing us the same conditions faced by a policy-driven, real-world, geostationary relay provider capable of global coverage. Much of the information used to design our model comes from the Space Network Users' Guide (SNUG) (NASA, Exploration and Space Communications Projects Division 450, 2007b), however additional information has been collected from published articles, government websites and reports, standards documents, and conversations with industry professionals.

## 3.1 The Technical Component

The technical component of our model is used to determine whether management protocols meet NASA TT&C requirements and is used to address our first research question. Physically, the SN ground segment is composed of three ground terminals, each supporting 120 degrees of space segment coverage. The three supporting ground terminals include: White Sands Ground Terminal (WSGT), located near Las Cruces, New Mexico; Second TDRS Ground Terminal (STGT), co-located with WSGT; and Guam Remote Ground Terminal (GRGT), located at the Naval Computer and Telecommunications Station (NCTS) in Guam. Each terminal communicates with more than one spacecraft and all spacecraft are managed by the provider's Operations Center.

We can simplify this model in multiple ways. First, since all spacecraft supported by a single terminal are nearly the same distance from that terminal and operate under the same policies (Eddy & Fuentes, n.d.), we can assume the communication channels of those spacecraft have comparably similar channel properties. Because of this, we include only one spacecraft per ground terminal within the model. Next, through conversations with NASA engineers, we know that all three terminals have comparably similar space-to-ground and ground-to-space channel properties (Eddy & Fuentes, n.d.). This means the only difference between each terminal is in the ground configuration. Since WSGT and STGT are co-located, we assume the two terminals have nominal differences in their ground configurations; thus we remove STGT from the model. Lastly, we consider the difference between WSGT and GRGT. In this case, the key difference is the distance between the terminal and the Operations Center. Since ground segment implementation specifics are beyond our scope, we abstract the added distance as latency. Given that our model is based on the SN, we estimate this latency using the maximum amount of time allowed to tranfer messages in the NISN backbone. We calculate this estimate as one half the maximum Round Trip Time (RTT) for Mission Critical NISN services (NASA, Space Communications and Navicagion (SCAN), 2007). With latency now being the only difference between WSGT and GRGT (and with GRGT having higher constraints than

WSGT), we assume that any protocol that succeeds using GRGT will also succeed using WSGT; thus we remove WSGT from our model. This compresses our technical component to a 2-node, point-to-point network.

For our model to work, we must define the criteria for success. We do this using NASA's SN TT&C requirements. The first part is easy. As mentioned earlier, Tracking techniques are not expected to change, so we assume it passes all requirements. Telemetry and Command, however, must meet the following requirements provided by NASA Systems Engineers (Eddy & Fuentes, n.d.):

1. Message delivery is real-time
2. Order of message delivery is predictable
3. Messages arrive without error (95% success)
4. Message delivery is acknowledged (Command only)
5. Protocol overhead is minimal

Requirement 5 is highly subjective and only applies to the implementation and design of new protocols; making it an unnecessary criteria for our study. Likewise, requirement 4 is assumed to pass because our management protocols either support notifications or can easily be supplemented to do so. Requirements 1 through 3, however, are relevant for our protocols.

Since spacecraft communication is asymmetric, we model two channels: a forward-link and return-link. Both channels are assigned the following channel affects: bandwidth, Bit Error Rate (BER), and latency. The first, bandwidth, is a measure of the number of information bits transmitted from one end of a channel to the other within a given timeframe. In our case, we use Kbps which measures the number of Kilobits (1024 bits) sent every second. The bandwidth for TT&C forward- and return- links in our model represent those planned by SCaN (Schier et al., 2005).

The next affect, BER, is derived by using a conversion formula against a known Energy-per-bit to Noise ratio (Eb/No) (Ippolito, 2008) which is a product of spacecraft engineering (Eddy, n.d.). In our study, we use the following formula to convert Eb/No in decibels (dB) to BER values.

**Eb/No to BER Conversion:**

$$BER = \frac{ERFC\left(\sqrt{10^{Eb/No\ dB \div 10}}\right)}{2}$$

Where

- $BER$ is the Bit Error Rate
- $ERFC()$ is the Complementary Error Function

Latency, the last of our channel affects, is driven by the technology and environment of the network. It measures the amount of time required for data to travel from its origin to its destination. We estimate the latency between the spacecraft and ground terminal by multiplying the speed of light in a vacuum by the distance required for geostationary orbit (rounded up to the nearest millisecond). This is added to the abstracted ground segment latency to get the total channel latency.

**Latency between a GRGT Supported TDRS & the Ops Center:**

$$Latency_{Channel} = \frac{SP_{Light}\ m/s}{Dist_{Geo}\ m} + \frac{RTT_{Critical}}{2}$$

Where

- $Latency_{Channel}$ is the total latency between a GRGT supported Spacecraft and the Provider's Operations Center

- $SP_{Light}$ is the speed of light in meters per second within a vacuum.
- $Dist_{Geo}$ is the distance in meters above the equator at which geostationary orbit is possible.
- $RTT_{Critical}$ the maximum Round Trip Time in seconds as defined for NISN Mission Critical services.

Latency and error rates are both also impacted by the hardware being used. More specifically, network device queues have historically caused latency and errors as a result of naturally occurring traffic bursts (Baker & fairhurst, 2014). Queue delays occur in most data networks due to the time it takes to move packets from a source interface to their destination interface. Assuming no errors occur along the network path and given a constant queue size and bandwidth, queue delays are predictable. However, bursty networks are prone to unpredictable queue sizes, resulting in variable delays. Likewise, errors can occur when a queue receives more packets than it can send. This case leads to congestion until the queue is full; at which point packets must be dropped. We consider the IRIS as our model for queue-related concerns aboard the spacecraft. Since the ground segment is beyond our scope, we assume the abstracted latency already represents queue delays.

### 3.1.1 SNMP

Simple Network Management Protocol (SNMP) has been used in network device management for years and has undergone 3 major version revisions. Our study focuses on SNMPv3 (*SNMP Version 3 (SNMPv3) Message Format*, 2005) which is the latest version defined by the Internet Engineering Task Force (IETF). SNMP works over a User Datagram Protocol (UDP) stack and is composed of two node types: Management Stations and Agents. Agents are managed network devices that have Management

Information Bases (MIB) which SNMP uses to collect data. MIBs are data definitions that describe how information about a device or system should be stored. Management Stations can get information from Agents using one or both of two communication methods: poll-driven and interrupt-driven (*SNMP Version 3 (SNMPv3) Message Format*, 2005). Poll-driven communication requires the Management Station to request data from the Agent before it is sent. Interrupt-driven communication, on the other hand, occurs when the Agent enters a predefined state or one or more of its managed data crossed a predefined threshold; at which point the data is sent to the Management Station. A Management Station may also set management information by sending a request to which the Agent responds. This operation is similar to poll-driven communication, except instead of getting information, the Management Station is setting it.

In terms of Telemetry, SNMP has two options to satisfy the provider's needs. First, the provider can use a poll-driven approach. SNMP can do this using its GetRequest and/or GetBulkRequest PDUs. GetRequests work by sending a comprehensive list of managed objects to the Agent. Agents respond to GetRequest PDUs with Response PDUs (formerly GetResponse) which contain the requested data for each object. GetBulkRequests work similarly, except they allow the Management Station to request certain object collections (tabular data in the MIB definition) without explicitly requesting each object in the collection. Assuming the provider has designed MIBs in a tabular form, GetBulkRequest has the advantage of using less forward-link bandwidth than its GetRequest counterpart.

The second Telemetry option is using an interrupt-driven approach. Though interrupts aren't commonly used to collect routine information (*Understanding Simple Network Management Protocol (SNMP) Traps*, 2006) (*SNMP Version 3 (SNMPv3) Message Format*, 2005), it might be best to implement Telemetry in this way due to the asymmetric nature of space communication. SNMP does this through a mechanism known as a Trap. During engineering of the spacecraft MIBs, Traps could be set to catch specific state changes of the spacecraft, at which point the Trap is sent to the Management Station. Assuming Traps are set on a timer to send Telemetry data at a known interval, Telemetry can be achieved without the use of any forward-link bandwidth.

There's only one option for Command using SNMP. The poll-like SetRequest can be used by Management Stations to set certain managed data objects at the Agent. Once a SetRequest is received, Agents set the MIB values and send a Response to the Management Station identifying the new values of the configured managed objects (which acts as the Command response in requirement 4 supplied by NASA Systems Engineers). Any spacecraft subsystem software that is monitoring these MIB values can quickly and easily identify the necessary Command-related actions based on the new values set by the Management Station. Responses are easily associated with their request because they share a Request ID in the protocol header.

Since SNMP runs over UDP, its underlying protocol stack does not require the establishment of a connection nor does it come with any retransmission timers. The design of SNMP is equally simple in that its communication methods do not require a connection and retransmissions are not part of the protocol specification. In the case of errors, the Management Station's SNMP implementation may incorporate algorithms for configurable retransmissions, which means timeouts for this protocol can exist. Traps, however, are not capable of retransmission since they are not acknowledged.

### 3.1.2 TFTP

Trivial File Transfer Protocol (TFTP) (Sollins, 1992) is a simple, lightweight protocol used to transfer files between a Client and a Server over UDP. TFTP Clients have two basic actions: reading files from a Server and writing files to a Server. Each action has a similar request-response method of communication. When a Client reads from the Server, it begins by sending a Read Request (RRQ) packet. If the Server accepts the request, it responds with a DATA packet containing the first 512 Bytes of the file. For every DATA packet the Client receives, it responds with an ACK packet containing the DATA packet's Block Number which lets the Server know the packet was received. This DATA-ACK process continues until the last DATA packet of the file is received. The Client knows this has happened when it receives a DATA packet containing less than 512 Bytes. If the file is evenly divisible by 512 Bytes, the Server send an additional DATA packet that's 0 bytes long. Writing files to the Server is a very similar process. Clients initiate the transfer with a Write Request (WRQ) packet to which the Server responds with an ACK packet. Once the Client receives this ACK, it sends the first DATA packet to the Server; at which point the DATA-ACK process begins until the terminating DATA packet is received by the Server.

TFTP Telemetry and Command both require an assumption that processes exist at the Operations Center and spacecraft that look for, parse, and operate on files as they are received from their counterpart. Telemetry must assume that subsystem management data is collected, stored, and forwarded as files aboard the spacecraft. Within the ground segment we must also assume a process exists to intercept Telemetry files and present the data to operators in a suitable manner. In this configuration, either the Operations Center or the spacecraft can be the Client. If Operations Center is the Client, it will need to send RRQs to the spacecraft requesting known Telemetry filenames. On the other hand, if the spacecraft acts as the Client, it would send WRQs to the Operations Center. Neither method is significantly more efficient than the other because protocol overhead is nearly the same.

Command requires similar assumptions to Telemetry. The Operations Center requires processes that generate Command files and transfers them over TFTP to the spacecraft. It also requires processes aboard the spacecraft that intercepts and operates on incoming Command files. Unlike Telemetry, Command requires that the Operations Center assume the role of Client. This is because the Client initiates all file transfers and there is no way the spacecraft would know in advance that the Operations Center needs to Command it. Command messages are implicitly acknowledged given that each packet of the message is acknowledged (which satisfies requirement 4 supplied by NASA Systems Engineers).

Similar to SNMP, TFTP runs over UDP and therefore the underlying protocols are connectionless and timer-less. TFTP however, operates under a pseudo-connection, lock-step paradigm and uses retransmission timers to compensate for lost packets. Lock-step connections require that each node have only one packet in the network at a given time. The simplicity of TFTP enables the Client and Server both to predict and expect the next packet in the exchange (whether it's a DATA or ACK packet). Because of this, timers are set after a packet is sent and, if the next expected packet is not received before the timer runs out, the packet is retransmitted. Fortunately, since the timer length is undefined by the protocol, we can configure it to an appropriate value for our latency. The lock-step connection can break in one of two ways: the file has finished transferring or a connection timer expires. The connection timer is different than the retransmission timer, but is also configurable to suit our latency.

### 3.1.3 SCP

Secure Copy (SCP) (Rinne & Ylonen, 2013) is not a standard protocol (Pechanec, 2007), which makes it difficult to find solid documentation. Despite this, there are two things we know about SCP that helps us define how it works. First, we know that SCP runs over Secure Shell (SSH) (Pechanec, 2007) which runs over TCP (Ylonen & Lonvick, 2006). Next, we know that SCP evolved out of Remote Copy (RCP) on the BSD Operating System and SCP differs little from one implementation to the next (because they were all forked from the same source) (Pechanec, 2007) (Rinne & Ylonen, 2013). SCP is used to transfer files between a Client and Server over a secure transport. Before file transfer can begin, a TCP and SSH connection must first be established. Once this is done, the transfer begins. Like TFTP, either node can initiate the transfer, whether it's reading or writing files. Whenever a node is the sender it goes into SCP Source mode and when it's receiving files it's in SCP Sink mode (Pechanec, 2007). When the SSH connection is made, the Source knows which file(s) it should send and begins sending immediately. Assuming we send only one file at a time, SCP Sources begin by sending a Cmmmm message (Pechanec, 2007) which defines the mode of transfer, file size, and file name. After sending this message, the Source begins sending packets filled with file data until the file has been completely sent. The Sink responds to every file with an ACK that gives either an Ok, a Warning, or an Error. If an Error occurs, the file is considered corrupt and the connection is broken. This is ok, however, since TCP is capable of correcting many errors as long as the network path isn't too erroneous.

Telemetry and Command for SCP follows the same assumptions as TFTP above. Processes for file interception and generation must exist for this form of management to work. Likewise, it's possible for either the Operations Center or the spacecraft to initiate Telemetry transfers, however Command initiation must still remain on the ground. A big weakness of SCP over TFTP and SNMP is the amount of time required to set up a connection and the fact that each file transfer needs to establish a new connection. This requires a lot of protocol overhead and adds constraints to available bandwidth and increases the amount of time it takes for a message to fully reach its destination. The advantage gained over the other protocols is the end-to-end upper-layer encryption.

Security is certainly important at all layers, but the pros and cons of having encryption at the application layer is beyond our scope.

Since SCP runs over TCP and SSH, we have both timers and connections in the protocol stack. SCP itself, however, doesn't have timers nor does it establish a connection. When analyzing this protocol, we need to observe the impacts latency has on TCP communication. SSH connections are merely security-based and should have no impact on our criteria.

### 3.2 Management Protocol Analysis for an IP-Enabled Fleet

We began this study under the impression that our technical component would exhibit stochastic properties requiring experimental analysis. After further review, however, we were surprised to find that all stochastic properties in our model can be overcome and therefore we assume they have no impact on the success criteria of our protocols. It was also surprising to find that certain static properties that we thought might cause problems can be ignored as well. We elaborate in the following paragraphs.

First we look at bandwidth. TDRS TT&C forward- and return-links use 2Kbps Ku-band and 4Kbps Ku-band, respectively (Bhasin & Hayden, 2004). These are static values based on SCaN mission plans for 2015, but can be adjusted for models of future studies if needed. The only impact bandwidth might have on our study is the implicit criteria: can enough Telemetry and Command be transmitted. However, NASA's technique has been to reserve enough bandwidth on the return-link to support any and all Command responses and use the remaining return-link bandwidth to dump as much Telemetry as possible. Since the return-link is larger than the forward-link (which means response traffic should never fill the return-link) and since the amount of Telemetry is flexible, we assume that bandwidth has no impact on the success of our protocols.

Next we look at BER. As stated earlier, BER is calculated using the Eb/No of the communication channel. Most NASA missions implement a BER of $1e^{-5}$ or better (Eddy & Fuentes, n.d.). However, given that Eb/No is adjustable during engineering (Eddy & Fuentes, n.d.), BER is implicitly adjustable and therefore we assume that BER can be set significantly low enough to have nearly no impact on our success criteria. We say ”nearly no impact” because it is generally accepted in the networking community that errors will always exist (however frequent or infrequent they may occur). For this reason, we include an analysis on the impacts of errors occuring in the conclusions section. Figure 3.1 illustrates how higher Eb/No values result in lower Bit Error Rates.

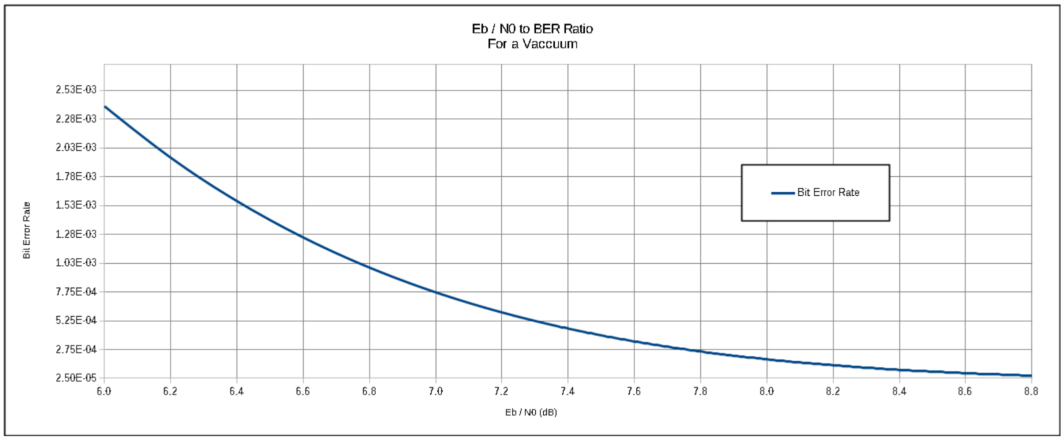

Figure 3.1: *Relationship Between Eb/No and BER*

Graph illustrating the inverse relationship between Eb/N0 and BER. As Eb/No continues toward infinity, BER experiences an asymptotic tendency toward 0.

Latency is the next parameter we address. Since both channels have the same latency, we only need to calculate the value once. As defined earlier, we calculate latency using the speed of light within a vacuum, the distance from the equator to geostationary orbit, and half the maximum RTT allowed for Mission Critical NISN services. Given that the speed of light is 299,792,458m/s and geostationary orbit is roughly 35,786,000m above the equator, we say the ground-to-space latency is 119.369ms (which rounds up to 120ms).

Next, given a max RTT of 120ms (NASA, Space Communications and Navicagion (SCAN), 2007), the latency between the Ground Terminal and Operations Center is estimated at 60ms. This gives us a total channel latency of 180ms.

Recall that latency and error rates are affected by queues in bursty networks. Fortunately, the relay's Telemetry and Command operates over an out-of-band router interface; meaning the interface queue is not shared with other traffic in the network. The interface's queue size can be adjusted to accommodate the maximum data size capable of being sent by our management protocols (given the known bandwidth), which eliminates packet drops. This kind of adjustment is possible using the Cisco IRIS which boasts the same OS as terrestrial Cisco devices (*Cisco 18400 Space Router*, 2010). The OS also comes stock with Active Queue Management (AQM) (Baker & fairhurst, 2014) implementations that work to minimize network latency and further help us predict queue delays (Eddy, n.d.). With this we can set a min and max queue delay based on the queue size and traffic loads. Since interface traffic is directly sent over the backplane of a router when possible (rather than being queued) (Eddy, n.d.), it's unlikely that a queue delay large enough to significantly impact our research will exist; especially when considering the IRIS has a throughput of 250Mbps (*Cisco 18400 Space Router*, 2010) (which is much larger than the return-link bandwidth). Though ground segment implementation is beyond our scope, we assume a router at the operations center with an equally flexible queus (perhaps another COTS Cisco device).

Since our model assumes that queue-related errors do not exist and since BER can be ignored, the only remaining source of errors (and uncertainty) comes from the sensitivity of protocol timers. If any protocol in the stack is time-sensitive, we compare the length of timers to our latency. In some cases, these timers are adjustable and we can assume no uncertainty exists. In our model, the only stack where time-sensitivity causes concern is

SCP which has a TCP connection. Here we use TCP's protocol radius (Wood et al., 2007) to determine whether latency has a significant impact on our success criteria. For the sake of our study, if our channel latency has no impact on TCP retransmissions, we calculate the largest queue delay acceptable within our model and assume retransmissions are minimal.

## 3.3 The Financial Component

The financial component represents a shift from the SN's current bent-pipe space segment to a fully IP-enabled space segment using phased integration. This component is used to address our second research question. We begin by defining the costs associated with the endeavor which we then use to build a financial profile. The profile categorizes costs according to: 1) the way NASA functionally distributes funding and 2) the time periods impacting the budget. Costs associated with this study include: salaries of bent-pipe operators, salaries of IP network operators, spacecraft hardware costs, launch costs, and engineering costs. Estimates are represented as Rough Orders of Magnitude (ROM) due to the limited availability of requisite product and service information. This is a common practice in assessing high-cost projects (Eddy, n.d.), so we feel the outputs of our study will be sufficiently accurate.

Salaries are based on the cost of operators and network administrators at the senior administrative level in New Mexico (where the SN Operations Center is located). Relay hardware, baseline engineering, and launch estimates are based on the second generation TDRS (H, I, and J) (*Tracking and Data Relay Satellite (TDRS) - H, I, J*, n.d.) built using the boeing 601 bus (*High-Power Spacecraft for the 21st Century*, n.d.). IP routing and digitization payloads are modeled using the Cisco IRIS and its Modem Interface Card (MIC) (*Cisco 18400 Space Router*, 2010). Finally, additional engineering costs are assessed by determining whether high-impact, physical spacecraft parameters are within allowable limits.

### 3.3.1 Financial Profiling by Function

As a public entity, NASA is bound by budgetary constraints set by the federal government. NASA uses the Planning, Programming, Budgeting, and Execution (PPBE) process (NASA, 2008) which requires a detailed analysis during budget formulation. This budget is decided based on strategic goals and objectives of the agency according to the needs of each program and its associated projects. In our study, there are two NASA projects that are impacted: the Space Network Project at Goddard Space Flight Center (GSFC) which pays for the operations of spacecraft once they reach orbit and; the TDRS Project at GSFC which pays for the development and launch of spacecraft (from vendors like Boeing) on procurement contracts (Eddy, n.d.). For this reason, we divide our financial profile into two functional categories: operational costs (operator annual salaries) and deployment costs (hardware, launch, and engineering costs).

First, we look at operational costs. Since the SN requires 1 bent-pipe operator per 3 spacecraft (Kunath, Reinert, & Barnes, n.d.), we assume that IP-enabled relays require the same fleet-to-operator ratio. In our model, this means that a fully IP-enabled fleet of a given size requires the same number of operators as a fully bent-pipe fleet of the same size. Bent-pipe operator salaries are based on estimates used in similar SCaN studies. IP-enabled relay operator salaries, however, are determined using the commercial salary of a senior-level network administrator in New Mexico (where the SN Operations Center is located). We compare this value to the US General Schedule (GS) pay grades, determine the highest grade it fits in, and set it at the highest step in that grade (rounding to the nearest thousand). We choose the highest grade and step due to the added value of working in a mission-critical, high-clearance, position.

Next, we look at deployment costs. Since engineering impacts the hardware used aboard the spacecraft, we first determine whether an IP-enabled spacecraft requires modifications to the bus before we can determine a cost. Since IP stacks are an integral part of most Real-Time Operating System (RTOS) kernels (like those used aboard spacecraft), we assume that engineering onboard LANs requires nominal implementation changes. However, the addition of payloads to the bus may require further engineering if the thermal load, power, and/or mass of the spacecraft exceed specified thresholds. To determine whether these values exceed their thresholds, we first determine how many MICs are required by each spacecraft. Each MIC supports 3 transponders (*Cisco 18400 Space Router*, 2010), which for the purpose of this research are defined as the collection of physical equipment between the SDR and the antenna used to establish a communication channel. The second generation TDRS has the ability to establish 9 simultaneous channels (NASA, Exploration and Space Communications Projects Division 450, 2007b) (Eddy, n.d.) through: 2 Single Access (SA) antennas; 1 Multiple Access

Table 3.1: *Relay Hardware Specifications*

| **Hardware Property** | **Value** |
|---|---|
| TDRS Power - Allowance | $2758_W$ |
| TDRS Dry Mass - Allowance | $1490_{kg}$ |
| TDRS Power | $2042_W$ |
| TDRS Dry Mass | $1319_{kg}$ |
| TDRS Transponder Count | 18 |
| IRIS Power | $32_W$ |
| IRIS Mass | $10_{kg}$ |
| MIC Power | $177_W$ |
| MIC Mass | $25_{kg}$ |
| MIC Transponder - Allowance | 3 |

A collection of TDRS-, IRIS-, and MIC-related specifications which outline the current and allowable values for thermal load, power, mass, and interfaces. Values collected from: (*Tracking and Data Relay Satellite (TDRS) - H, I, J*, n.d.); (*Tracking and Data Relay Satellite (TDRS) Characteristics*, n.d.); (Eddy, n.d.); (*Cisco 18400 Space Router*, 2010); (**?**)

(MA) antenna capable of 1 forward and 5 return channels and; 1 out-of-band TT&C antenna. If we assume each channel requires a redundant pair of all transponder equipment, we end up with 18 transponders supported by 6 MICs.

Managing thermal load means that the dissipation of thermal energy from electronics is properly removed from the bus. Boeing 601 models have ventilation pipelines that extend out from the honeycomb cavities where payloads are mounted (Fortescue et al., 2011). However, these pipelines must account for a threshold of temperature and may require additional engineering if the components generate high levels of heat. Assuming an engineer is given the total power of the equipment ($P_{tot}$) and the amount of power radiated by the antenna ($P_{rad}$), they can determine the amount of power dissipated as thermal energy ($P_{dis}$) by calculating: $P_{dis} = P_{tot} - P_{rad}$ (Eddy, n.d.). These kinds of measurements and adjustments are required before any spacecraft can be flown. Since thermal load analysis and modifications are already required, and since we don't have complete thermal data from a TDRS, we assume that nominal changes in workflow (and cost) are required. We also assume that the spacecraft bus has been designed for continuous, uninterrupted operation under other conditions where thermal load increases.

The impacts of power and mass are easier for us to estimate without being spacecraft engineers. According to table 3.1, second generation TDRS use 2042 watts and have a dry mass of 1319.5 kg. The TDRS allowance for these values are: 4800 watts and 1490 kg. Since we know the power and mass of an IRIS (32 watts, 10 kg), the power and mass of a single MIC (177 watts, 25 kg), and the number of MICs required (6), we can support our model by comparing the maximum allowed power and mass with the newly calculated power and mass.

**Power Allowance:**

$$Power_{max} = 4800_W$$

$$Power_{new} = 2042_W + 32_W + (177_W \times 6) = 3136_W$$

$$Power_{max} >= Power_{new}$$

**Mass Allowance:**

$$Mass_{max} = 1490_{kg}$$

$$Mass_{new} = 1319.5_{kg} + 10_{kg} + (25_{kg} \times 6) = 1479.5_{kg}$$

$$Mass_{max} >= Mass_{new}$$

As shown above, the addition of payloads do not cause the bus to exceed its thresholds. This means engineering and launch costs should remain roughly unchanged. With this, we determine the deployment cost of an IP-enabled relay by calculating the sum of: the second generation TDRS launch and hardware costs, the cost of an IRIS payload, and the cost of 6 IRIS MICs.

### 3.3.2 Financial Profiling by Time Period

In addition to functional categories, there are two time periods where costs differ: the period during integration and the period after integration. We use these periods to further divide our financial profile into four categories:

1. Deployment costs during integration.
2. Operational costs during integration.
3. Deployment costs after integration.
4. Operational costs after integration.

In order to determine the difference between an ongoing bent-pipe space segment and a space segment integrating IP-enabled relays, we must compare the costs of each deployment plan side-by-side. Unfortunately, the SN doesn't provide an ongoing deployment plan and, as a result, we've designed an adaptable plan that represents a relatively simple fleet. Timeline contingency shifts (such as from failed, delayed, and/or cancelled launches) are considered part of normal risk assessment and are beyond our scope. Each of our plans operates in 5-year cycles beginning with 2 off years (no launches) and ending with 3 launch years (1 launch per year). We refer to these cycles as procurement cycles, which closely represent the way NASA purchases TDRS (such as the second generation TDRS which were all bought on a single contract). However, since NASA doesn't have a set way it disburses payments in a procurement contract (Eddy, n.d.), we evenly divide the cost of the 3 spacecraft over the 5-year procurement cycle.

## Adaptable Deployment Plan

- $T$ is an integer value representing the current year. $T = 0$ is 2015.
- The ground segment is fully IP-enabled at $T = 0$.
- The expected lifetime of each relay is 15 years.
- 3 relays are launched every 5 years; 2 off years followed by 3 consecutive launch years.
- There are 9 bent-pipe relays (3 per ground terminal) at $T = 0$.
- The first bent-pipe relay was launched at $T = -12$ and retires at $T = 3$.

The key difference between each plan is the integration offset. In a systems environment, when one system is replacing another, it's common to slowly phase the new technology in; running the new and legacy systems at the same time. This can result in additional overhead, but offers an opportunity to: work out issues without jeopardizing the system; migrate customers to the new platform in a reasonable timeframe; limit up-front costs of a complete system overhaul and; much more. In our study, the next scheduled

5-year cycle of the bent-pipe plan begins at $T = 1$ with the next launch being $T = 3$. Conversely, in the integration plan, we use a 1-year phasing offset to ease the change from one system to another. Here, the next 5-year cycle begins at $T = 0$ with the first IP-enabled launch at $T = 2$.

During integration, operational costs are impacted by the increased number of spacecraft in orbit. The 1-year offset results in periods where 10 relays (rather than 9) are in the fleet. This requires a fourth (IP-enabled) relay operator. As integration moves toward a fully IP-enabled fleet, bent-pipe operators should slowly be phased out (or cross-trained) and replaced by IP-enabled operators. Though there are periods during integration where the fleet moves back to 9 spacecraft, it's realistic to assume the fourth operator will be present throughout the duration of integration and will only be phased out once the fleet is fully IP-enabled. Also, though the first IP-enabled spacecraft is launched at $T = 2$, it's more realistic to assume the fourth operator will be brought on 1 year early (at $T = 1$) for planning, training, and on-boarding.

Deployment costs during integration are affected in two ways. The first impact is obvious; given the added hardware, we estimate the cost increase caused by the payloads. The second impact is less obvious. Since costs are divided across 5-year periods, and since there is a 1-year overlap during the first year of integration, we know that this year will have higher procurement costs than any other year. For the sake of annual budget justifications, NASA would likely not agree to pay equal disbursements during this first cycle because the procurement costs at $T = 0$ would be more than double those at $T = -1$. Instead, it's more likely that the contract during this first cycle would allow NASA to pay less the first year and pay more each of the subsequent 4 years. To account for this, we calculate the sum of all money owed during the first procurement cycle (the cost of 3

IP-enabled relays and the remaining cost from the previous cycle) and evenly divide this value over the 5 years.

Cost comparisons after integration are simpler than those during integration. Since this period should have the same number of spacecraft and providers as it would in the ongoing plan, we can do one-to-one comparisons. I.e., we can directly compare the cost of launching 3 bent-pipe relays within 5 years to the cost of launching 3 IP-enabled relays within 5 years. Likewise, we can easily compare the annual salaries of 3 bent-pipe relay operators to 3 IP-enabled relay operators.

### 3.4 ROM Cost Analysis for an IP-Enabled Fleet

All data in this analysis is collected through static sources. Ultimately, we use this data to build the financial profile defined by our model. Unfortunately, the results of this analysis rely on the parameters of our deployment plan. More specifically, deployment and operational costs during integration are impacted by: Spacecraft Lifetime, Phasing Offset, Procurement Cycle Length, and Deployment Frequency. To compensate for this weakness, we follow-up our study with a sensitivity analysis of these input variables. It should be noted that raw numeric outputs from our model will vary given different deployment plan inputs. To reconcile this, we normalize our findings by representing them as percent changes between the two plans.

For ROM cost estimates, we first determine operator costs. Similar NASA studies have estimated bent-pipe operator salaries at \$131,000 (Kunath et al., n.d.). IP-enabled relay operators, however, require different skills and may vary in cost. Since such spacecraft would be managed through technologies comparable to terrestrial networking, we cross-reference Senior-level network administrator salaries in New Mexico (*Senior Network Administrator Salary in Las Cruces, NM*, n.d.) against the United States GS pay scale (US Office of Personnel Management, n.d.). Using the process described in the last section and given a commercial estimate of $103,000 annually, we land in the GS-15 range. The highest step in the GS system is step 10, which brings us to $130,810. After rounding to the nearest thousand, we find that IP-enabled operators and bent-pipe operators are likely to be paid roughly the same amount, thus we estimate both at $131,000.

Next, we determine the total cost of each relay. Since NASA's second generation TDRS are bent-pipe, we use their total deployment cost as a baseline price. We derive this value by taking the total cost of $800,000,000 (*National Aeronautics and Space Administration Media Kit; TDRS-J*, 2006) (which includes hardware, engineering, and launch costs) and dividing it among the 3 spacecraft (H, I, and J). Rounding to the nearest thousand, this means that NASA paid a ROM cost of $266,667,000 per TDRS. The IP-enabled relay, however, must include the cost of adding 1 Cisco IRIS and 6 MICs to the baseline price. Unfortunately for our study, IRIS was withdrawn from the market sometime after 2012, so no current estimates are available. However, since ROM estimates allow for a relatively large margin of error, we assume that each payload (IRIS and MIC) costs $1,500,000 (Eddy, n.d.). Using the baseline and payload prices, we estimate each IP-enabled spacecraft costs $277,167,000.

# 4 CONCLUSIONS

Here we provide a summary of our conclusions with more details in the subsequent sections. We begin with our protocol analysis. When we assume a BER of zero, each of the three NASA TT&C requirements pass for SNMP, TFTP, and SCP. Bandwidth and BER have no influence on our success criteria while latency raises concern for SCP's TCP layer due to its timers. After further investigation, however, we find that TCP is not affected by the channel property's relatively small values.

The cost analysis yielded a financial profile with four categories. The first, deploy costs during integration, produces two sub-categories: cost increase during the first procurement cycle and cost increase during the remaining cycles. These increases were 23.94% and 3.94% respectively. Operational costs during integration increased by 33.33%, deployment costs after integration increased by 3.94%, and operational costs after integration had no increase.

Since both analysis are based on stringent assumptions, we extend the research for both. For the protocol analysis, we determine the worst-case impact to latency when a first-order packet error occurs. SNMP Telemetry is the only message type that cannot implement retransmissions; giving an impact of 0ms when a packet is erroneous. Assuming the provider implements SNMP retransmissions for Command through the Management Station, all remaining message types experience an impact of 1 RTT (360ms).

Given that our financial profile is actually the output of a single permutation of 4 inputs, we do a series of variance-based sensitivity analysis to better define the impact of of a provider using different inputs. There are only two areas of the profile that exhibit variance among varied inputs: deployment costs during integration for the first procurement cycle and operational costs during integration. For each, we identify the top two worst-case scenarios. See Tables 4.5 and 4.7 in this chapter for a quick view of the outcomes.

## 4.1 Protocol Analysis Results

Having eliminated the need for stochastic analysis, we've statically analyzed each protocol's ability to satisfy NASA TT&C requirements. Here, we draw conclusions under the assumption that BER is zero, however we follow up by analyzing the impacts of non-zero BERs. Since bandwidth, BER, and queue errors have insignificant impacts on the provider's network, we find that the only remaining source of errors is the affect latency has on protocol timers.

Given that SCP is the only unadjustable, time-sensitive protocol in our study (due to TCP retransmission timers), we know that SNMP and TFTP can operate in our model without error. For SCP, we use TCP's protocol radius to determine that our model's RTT of 360ms (which is the sum of latencies for the forward- and return-links) is well below the TCP initial Retransmission Time Outs (RTO) of 3s (3,000ms) given in (Wood et al., 2007) and is likewise within the radius of normal TCP operations. This leaves a relatively large margin of 2,640ms for queue delays (if necessary) before SCP would experience errors in the network.

As described in the method, we assume the following regarding NASA TT&C requirements: Tracking passes all requirements; Command passes requirement 4 and; requirement 5 is not applicable. This leaves us with requirements 1 through 3 for Telemetry and Command. For our first requirement, ”Message delivery is real-time”, we know that real-time message delivery is dependent on the predictability of latency in the network and on whether the platform is using an RTOS. For forward- and return-link traffic, we find that latency is predictable for each protocol. Since we know that spacecraft use RTOSs, we can state that each protocol passes requirement 1 for both Telemetry and Command.

Next, we look at requirement 2, ”Order of message delivery is predictable”. We know that in a point-to-point network, if no one message can be delayed longer than another (due to packet retransmissions), then data delivery is serialized and message order is predictable. Since errors (through BER, congestion, and/or timeouts) are assumed to have insignificant impacts, we know that retransmission-related delivery delays are nonexistent. For this reason, we state that each protocol passes requirement 2 for both Telemetry and Command.

Lastly, we look at requirement 3, ”Messages arrive without error (95% success)” which requires messages to be delivered without errors ≥ 95% of the time. Given that BER-, congestion-, and latency-related errors are assumed to have insignificant impacts in our model, we state that each protocol passes requirement 3 for both Telemetry and Command.

In summary, based on the outcomes of our static analysis, we conclude that IP-enabled relays are capable of being managed using TT&C operations over SNMP, TFTP, and/or SCP. This conclusion, however, is dependent on highly constrained error-mitigating facilities and does not define the boundaries at which requirements begin to fail. Though these boundaries are beyond our scope, they should be revisited in future research.

### 4.1.1 Non-Zero BER Analysis

Though we’ve concluded that our protocols are capable of meeting NASA TT&C requirements, we know its impossible to have a BER of zero. This begs the age old question ”what if?” For this reason, we include an analysis of the impacts packet errors have on our model. Specifically, we look at the worst-case impacts to latency that providers need to be aware of and adjust for. We begin by illustrating how the error rates of packets and messages have the same inverse relationship to Eb/No that BER does. To do this, we need to make assumptions regarding the data being transmitted in the channel. First, we estimate an average packet size for each protocol and the expected number of packets for each message type. These values allow us to calculate the expectation values for packet loss probability and for message loss probability; which we represent as Packet Error Rate (PER) and Message Error Rate (MER), respectively.

Since NASA doesn't implement packets in its space links, we base our estimates on layer 2 traffic. Each protocol's payload sizes are based on the size of SN TT&C frames and the number of packets per message are based on the number of frames per SN TT&C message. Currently, Command messages consist of a single, 8 byte frame while Telemetry messages consist of 32 separate, 512 Byte frames (Eddy, n.d.). We also estimate header sizes for each protocol stack. For simplicity, we ignoring connection-related packets for layers 1 through 4 (meaning that SCP's layer 7 SSH connection packets are included). Also, since our study ignores implementation specifics of layers 1 and 2, we only estimate the headers of layers 3 through 7.

For SNMP, we include IP, UDP, and SNMP headers for each packet. Assuming we use IPv6, we have an IP header of 40 Bytes (Deering & Hinden, 1998), a UDP header of 8 Bytes (Postel, 1980), and an SNMP header of > 17 Bytes (*SNMP Version 3 (SNMPv3) Message Format*, 2005). The SNMP header is variable-length due to two fields: a UID for the destination application and a block for security information. If we assume 8 Byte UIDs and 64 Byte security blocks, our SNMP headers are 137 Bytes long; giving us 145 Byte Command packets and 649 Byte Telemetry packets. We use the Trap technique described in the method for Telemetry and the SNMP SetRequest technique for Command.

The next protocol, TFTP includes IP, UDP, and TFTP headers for each packet. TFTP headers (for DATA and ACK packets) are only 4 Bytes long (Sollins, 1992). With the 48 Bytes from IPv6 and UDP, the total header length is 52 Bytes; giving us 564 Byte Telemetry DATA packets and 60 Byte Command DATA packets. ACK packets have a payload of 0 Bytes, so ACKs for both Telemetry and Command are 52 Bytes long.

SCP is the last protocol and includes IP, TCP, and SSH headers. The minimum TCP header size is 20 Bytes (assuming no option flags are included) (*TCP/IP Guide*, 2005). SSH, however, has variable-length headers that depend on multiple factors: the cipher suite, the MAC size, and the payload size (Ylonen & Lonvick, 2006). Each SSH PDU consists of: a 4 Byte ”packet length” field, a 1 Byte ”padding length” field, a variable-length ”payload” field, a variable-length ”padding” field, and a variable-length ”MAC” field. SSH PDUs have an encrypted portion that includes every field except the MAC field. The padding field is a series of no less than 4 random Bytes which forces the encrypted portion of the PDU to be a multiple of a given number. This number is specified by the cipher used to encrypt each packet. If we assume high security is used (AES256-cbc as our cipher and HMAC_SHA1 as our MAC) then each SSH packet's encrypted portion must be a multiple of 32 Bytes with a MAC size of 20 Bytes. We use the following pseudo-code to calculate the total packet size for any given payload in our model.

---

```
int Get_SSH_Packet_Size(int payload_size)
{
   int ssh_pdu_size = 0;    // Size of SSH pdu in Bytes
   int remainder = 0;       // Paddable pdu portion mod cipher size

   // 4−Byte ”Payload Length” and 1−Byte ”Padding Length” fields
   ssh_pdu_size += 5;

   // X−Bytes of payload data
   ssh_pdu_size += payload_size;
```

```
    // Get remainder of paddable pdu portion mod the cipher size
    remainder = ssh_pdu_size % 32;

    // If cipher mod meets the minimum padding size (4) then add it
    // If not, add the remainder plus an addition 32
    if (remainder >= 4)
      ssh_pdu_size += remainder;
    else
      ssh_pdu_size += remainder + 32;

    // Add the MAC size
    ssh_pdu_size += 20;

    // Add the TCP header size
    ssh_pdu_size += 20;

    // Add the IP header size
    ssh_pdu_size += 40;

    return ssh_pdu_size;
}
```

Given a Telemetry payload of 512 Bytes, we calculate a packet size of 602 Bytes. Likewise, with a Command payload of 8 Bytes, we calculate a packet size of 112 Bytes. Each DATA packet sent by SCP will be followed by an SCP OK acknowledgement packet. OK packets have a payload size of 3 Bytes (Pechanec, 2007); giving a packet size of 112 Bytes. As mentioned above, SSH requires a connection to be established prior to sending data. This requires 6 additional packet exchanges between the client and server for each message being sent(Ylonen & Lonvick, 2006). These packets also have variable-length fields, but for simplicity, we assume each packet (request and response) is also 602 Bytes long. This sacrifices some granularity, but is a reasonable over-estimate for SSH and still allows us to illustrate the point. With these connection packets added, the client will send 7 packets per Command message and will receive 38 packets per Telemetry message.

Now that we have the size of each packet, we determine the expected rate of packet and message errors. We begin by calculating the probability that every bit in a given packet will arrive without error. From statistics, we know we can calculate the probability that every element in a series will succeed by multiplying the probability for success of each element in the series. If we think of each bit as an element and each packet as a series, we can derive the probability for success of the entire packet. To do this, we calculate the probability for success of a single bit (by subtracting BER from 1) and raise the result to a power equal to the number of bits in the packet.

We should be able to convert this value to PER by subtracting it from 1, however, in this study we consider both data and acknowledgement packets as part of the same stream; so both must arrive without error. We do this because if either packet is erroneous, both TFTP and SCP will kick-off a retransmission of the data packet. To calculate the probability that either packets will arrive erroneously, we treat each packet as an element in a series and use the same statistical principals as above. This allows us to calculate a PER for each protocol. Since SNMP has no acknowledgement packets, we use a packet series of 0 bits.

By treating each packet of a message as an element in a series, we can calculate MER the same way we calculated PER. There are, however, two additional adjustments. First, since we assume the provider implements SNMP Command retransmissions, we treat the data and acknowledgement messages as two elements in a series. Second, since SCP sets up a connection before sending data, we calculate the MER of the connection and the MER of the data exchange separately then treat them as elements in a series.

Given that our BER values are accurate approximations of the channel's bit error probability over a long period of time, we can use the simplified formulas below to calculate PER and MER.

**BER to PER Conversion:**

$$PER = 1 - (1 - BER)^{b_d + b_a}$$

$$MER = 1 - (1 - PER)^{p}$$

Where

- $b_d$ is the number of bits per data packet
- $b_a$ is the number of bits per acknowledgement packet
- $p$ is the number of packets per message

Now that we can calculate PERs and MERs for our protocols, we illustrate how NASA is capable of adjusting the communication channel so that each protocol can meet or exceed the 95% reliability criteria. This remains true even if the payloads of each protocol is larger than the frames used in our model. Figures 4.1 and 4.2 below represent

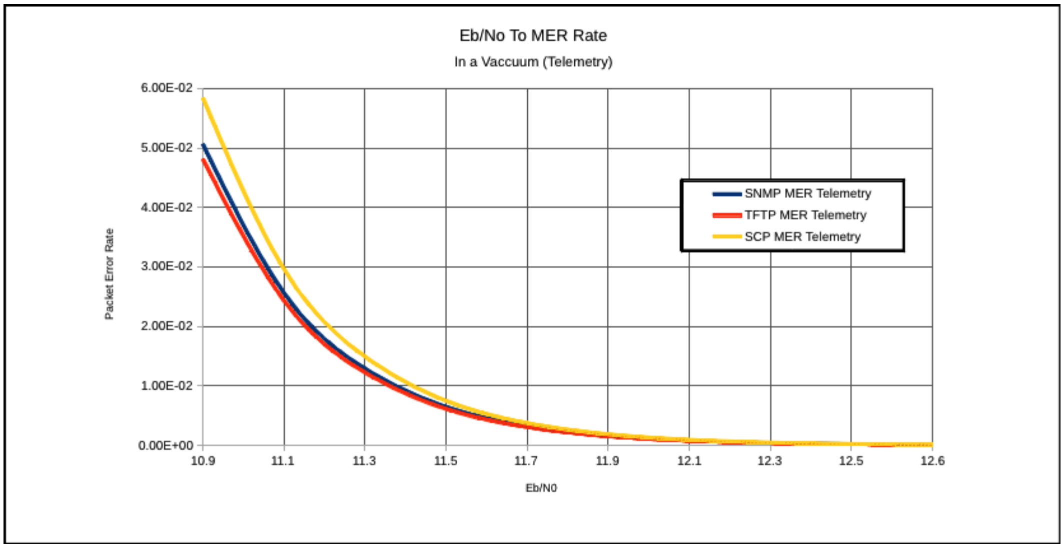

Figure 4.1: *Relationship Between Eb/No and MER (Telemetry)*

Graph illustrating the relationship between Eb/No and Message Error Rates for Telemetry using SNMP, TFTP, and SCP. All packets in a message have 512 Byte payloads and each message ignores Layer 1 through 4 connection setup. SCP's Layer 7 SSH connection, however, is included. TFTP and SCP packets include both data and acknowledgements. Telemetry messages in this model have 32 data packets.

the inverse relationship between Eb/No and MER for both Telemetry and Command using each protocol from our study. Notice that for each protocol, the plots converge toward a horizontal asymptote of 0 as Eb/No increases toward infinity. Command payloads larger than 8 Bytes, the resulting graphs shift and have varying slopes, but the plots still converge toward 0.

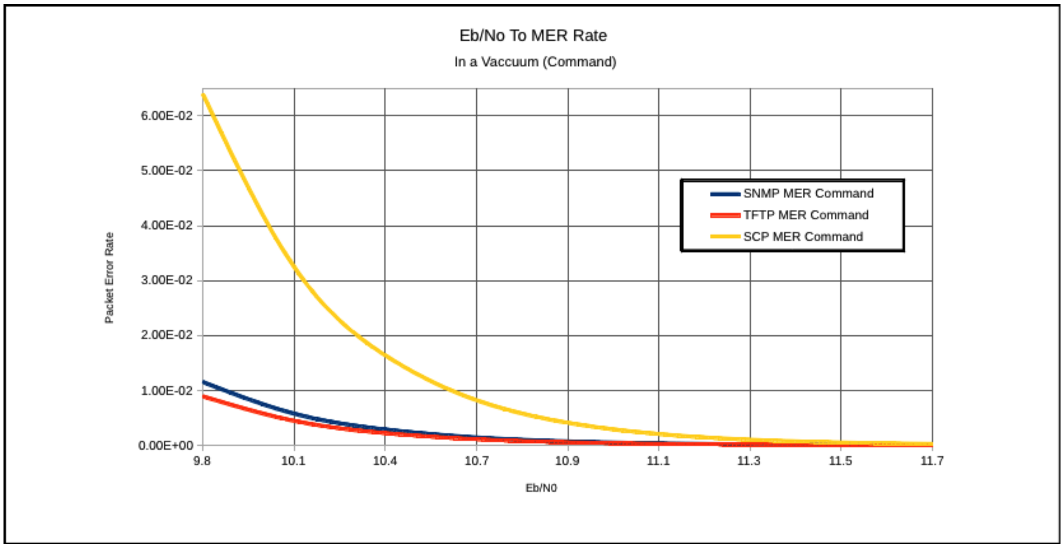

Figure 4.2: *Relationship Between Eb/No and MER (Command)*

Graph illustrating the relationship between Eb/No and Message Error Rates for Command using SNMP, TFTP, and SCP. All packets in a message have 8 Byte payloads and each message ignores Layer 1 through 4 connection setup. SCP's Layer 7 SSH connection, however, is included. TFTP and SCP packets include both data and acknowledgements. Command messages in this model have 1 data packet.

Though it's somewhat intuitive, we illustrate how the same is true for PER. Figures 4.3 and 4.4 below represent the inverse relationship between Eb/No and PER for both Telemetry and Command using each protocol from our study. The convergence toward 0 is especially important for our analysis because it shows that relatively low PER values are possible; meaning that second-order retransmissions (for SNMP Command, TFTP and SCP) will have significantly low enough probabilities that they can be ignored. For example, if we have a PER of $1e^{-5}$, given that an error does occur, the chances of losing the retransmitted packet is $1e^{-5} \times 1e^{-5} = 1e^{-10}$; which is magnitudes lower than the first-order retransmission. If we move that PER even higher, second-order retransmissions grow even less likely.

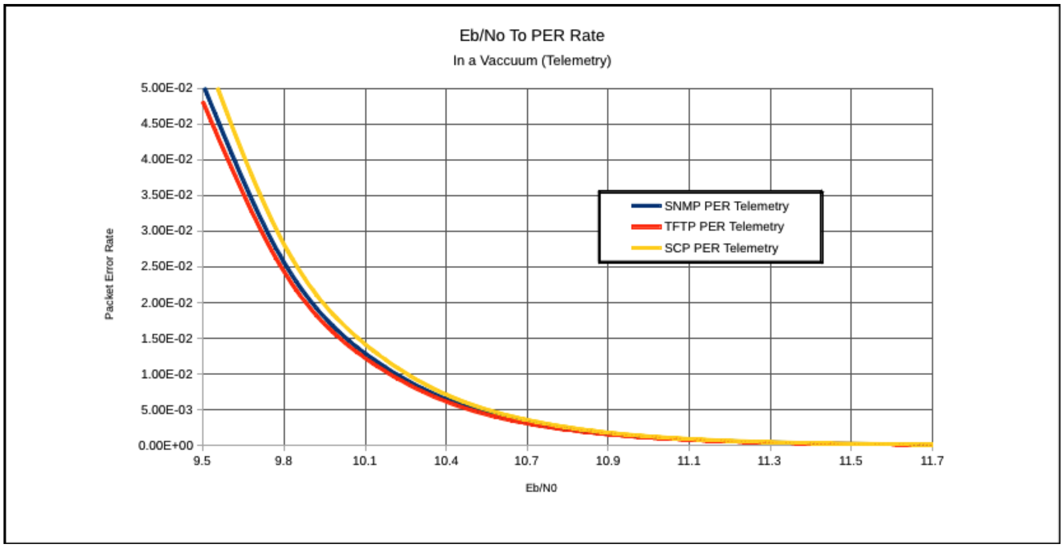

Figure 4.3: *Relationship Between Eb/No and PER (Telemetry)*

Graph illustrating the relationship between Eb/No and Packet Error Rates for Telemetry using SNMP, TFTP, and SCP. All packets have 512 Byte payloads. TFTP and SCP PERs include both the data and acknowledgement packet.

Now that we have better illustrated the error rates, we move on to analyzing the impacts of erroneous packets. We start with SNMP. For Telemetry, if any one of the 32 packets arrives erroneously, there is no way to recover the data sent. Since the Trap-based Telemetry model doesn't have retransmissions, there's no simple way to recover from the loss. This, however, does not prohibit SNMP from being used. Since NASA has an allowance for up to 5% message errors, they can adjust Eb/No such that MER is $5e^{-2}$ or lower. For Commanding, though SNMP SetRequests don't support retransmissions, the Operations Center may implement message-level retransmissions in the Management Station. In this case, since the SetRequest process is a two packet exchange (one SetRequest on the forward-link and one Response on the return-link), an erroneous packet will result in an additional delay equal to one RTT, or 360ms. Since UDP has error detecting mechanisms, the Operations Center will be able to identify and adjust for delays caused by erroneous Commands.

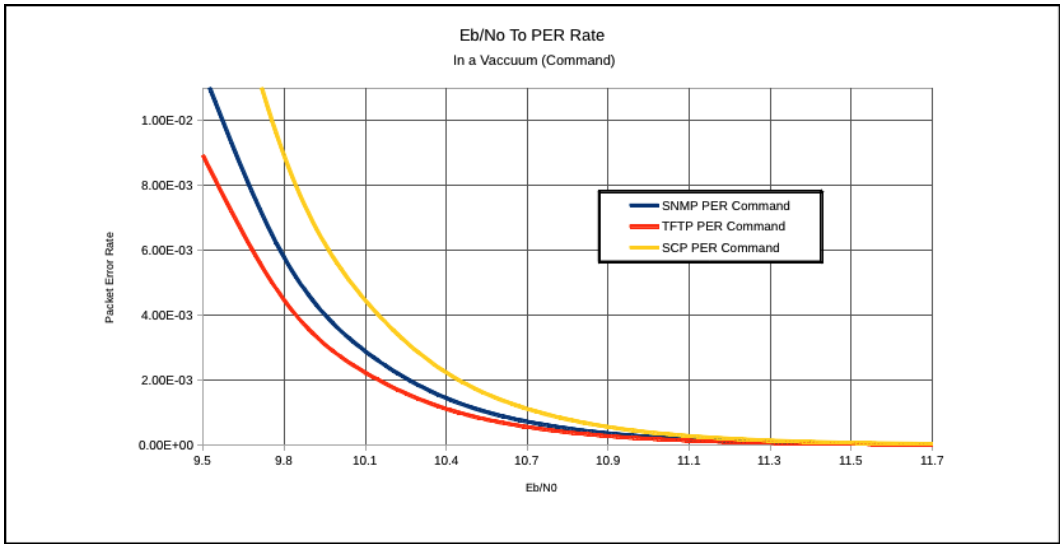

Figure 4.4: *Relationship Between Eb/No and PER (Command)*

Graph illustrating the relationship between Eb/No and Packet Error Rates for Command using SNMP, TFTP, and SCP. All packets have 8 Byte payloads. TFTP and SCP PERs include both the data and acknowledgement packet.

Unlike SNMP, TFTP and SCP do retransmissions at a packet-level (not at a message-level). This is true for both Telemetry and Command. In other words, an erroneous packet will only impact a single data-acknowledgement exchange. So, given a packet error in TFTP or SCP, the impact is an additional delay equal to one RTT, or 360ms. Similar to using SNMP, if the Operations Center uses TFTP (with its UDP mechanisms) or SCP (with similar TCP mechanisms), it will be able to identify and adjust for the impacts of corrupted packets.

Table 4.1 below summarizes the worst-case impacts to delay for each message type using each protocol (assuming PER has been set sufficiently low enough to ignore second-order retransmissions).

Table 4.1: *Non-Zero BER Worst-Case Impact Analysis Summary*

| Protocol and Message Type | Delay Impact for 1st-Order Retransmissions |
|---|---:|
| SNMP Telemetry | $0_{ms}$ |
| SNMP Command | $360_{ms}$ |
| TFTP Telemetry | $360_{ms}$ |
| TFTP Command | $360_{ms}$ |
| SCP Telemetry | $360_{ms}$ |
| SCP Command | $360_{ms}$ |

Summary of the expected worst-case impact to delay in the model given that a first-order packet retransmission (where applicable) occurs. The worst-case represents TT&C traffic between the SN Operations Center in New Mexico and the Guam Remote Ground Terminal in Guam.

## 4.2 Cost Analysis Results

Recall from the method that costs are separated into four categories which collectively form a financial profile. Here, we graph outputs of each category; showing the difference between the ongoing bent-pipe plan and the IP-enabled integration plan. Note that each point on the graphs is a discreet value and that the connecting lines only exist for readability.

The first category we discuss is deployment costs during integration which is shown in Figure 4.5. Given our deployment plan, integration takes place over 3, 5-year procurement cycles beginning at time $T = 0$ and ending at time $T = 14$. Notice that the first procurement cycle ($0 \leq T \leq 4$) experiences a greater cost difference than the remaining two integration cycles. As described in the method, this is explained by the 1-year offset, where the overlapping bent-pipe deployment costs are divided across each year of the first procurement cycle. Because of this, there are 2 values we derive for this profile category. First, we calculate the percent increase of deployment costs during each

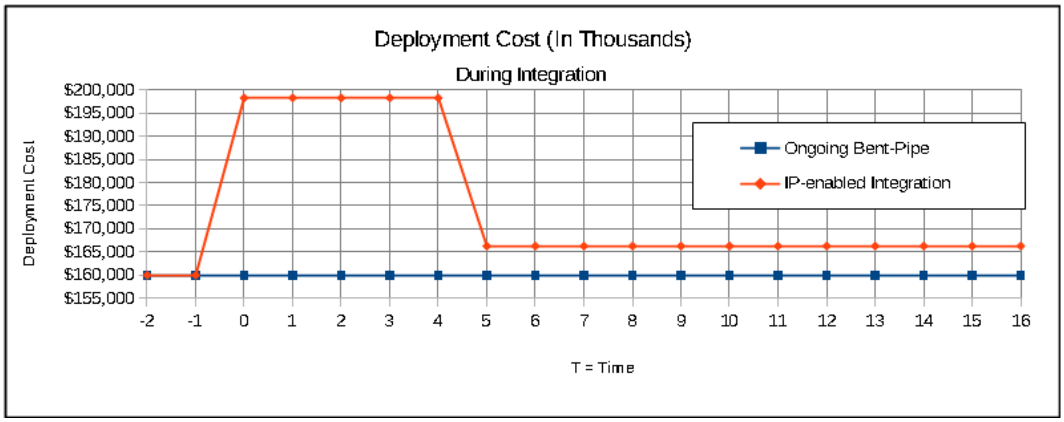

Figure 4.5: *Deployment Costs During Integration*

Graph showing the difference between IP-enabled and Bent-Pipe relay deployment costs during integration. Integration begins at $T = 0$ and ends at $T = 14$.

year of the first procurement cycle. Each of the five years has the same percent increase to deployment costs (23.94%). Second, we calculate the percent increase of deployment costs during each year of the two remaining integration cycles. Similar to the last value, each of the ten years has the same percent increase to deployment costs (3.94%).

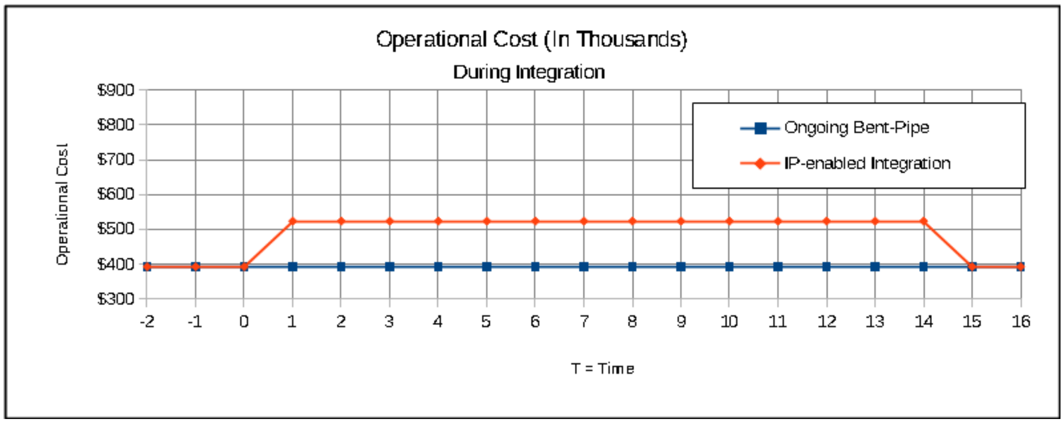

Figure 4.6: *Operational Costs During Integration*

Graph showing the difference between IP-enabled and Bent-Pipe relay operational costs during integration. Integration begins at $T = 0$ and ends at $T = 14$.

Next, we look at operational costs during integration which is shown in Figure 4.6. Recall from the method section that we assume the additional operator is brought on-board

1 year prior to the first IP-enabled spacecraft (so at $T = 1$ rather than $T = 2$). Likewise, recall that we assume the same number of operators remain onboard for the duration of integration (rather than constantly ”hiring and firing” operators throughout the process). Unlike the previous category, only one value can be derived from this category. Each year during $0 \leq T \leq 14$, has the same percent increase to operational costs (33.33%).

After integration is complete, all bent-pipe relays should have retired and all future procurements are assumed to be IP-enabled. Figure 4.7 shows the next category: deployment costs after integration. We can only derive a single value for this category as well. As the timeline extends beyond $T = 14$, each year has the same percent increase to deployment costs (3.94%).

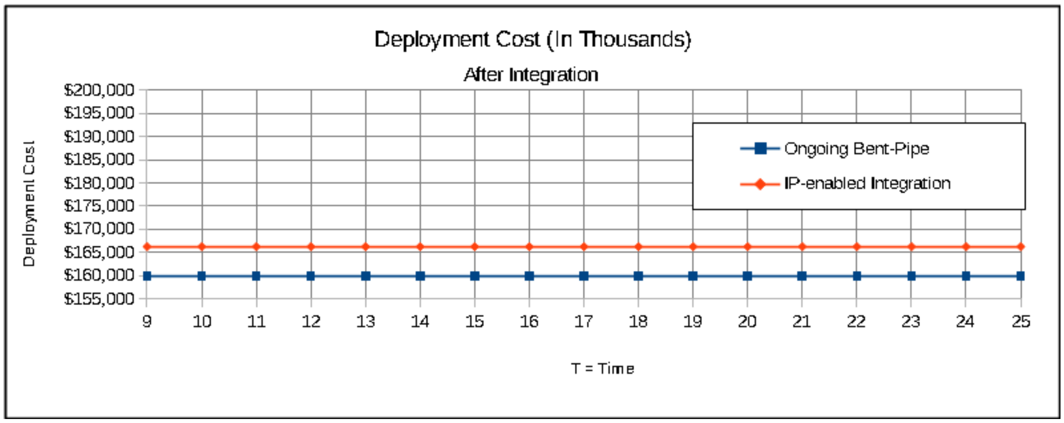

Figure 4.7: *Deployment Costs After Integration*

Graph showing the difference between IP-enabled and Bent-Pipe relay deployment costs after integration. Integration ends at $T = 14$.

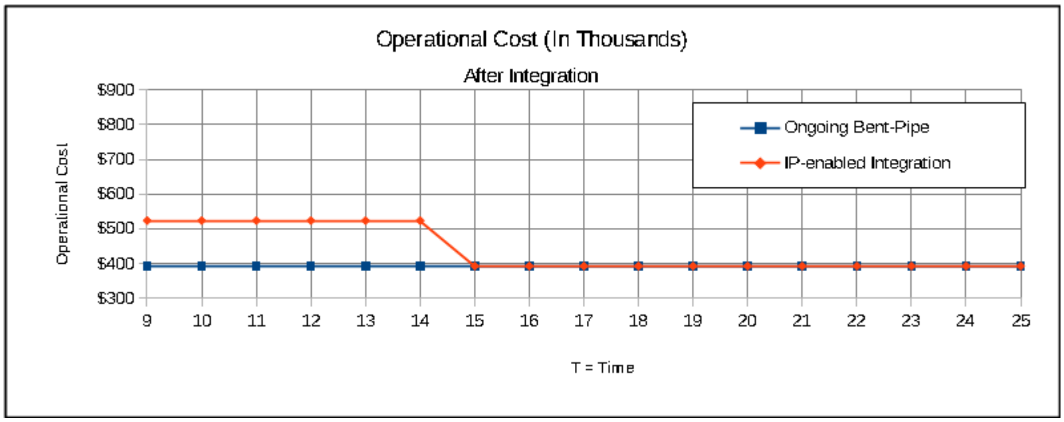

Figure 4.8: *Operational Costs After Integration*

Graph showing the difference between IP-enabled and Bent-Pipe relay operational costs after integration. Integration ends at $T = 14$.

Operational costs after integration, shown in Figure 4.8, is our last category. As you can see, while $T \geq 15$, the two plans show identical operational costs. Since bent-pipe operators and IP-enabled operators are assumed to have the same annual salary, and since the post-integration fleets are assumed to have the same number of operators, we know there's no increase for this category. In short, we can derive a single value from this category. Each year has the same percent increase to operational costs (0.00%).

Table 4.2: *Financial Profile Category Output Summary*

| **Profile Category** | % Incr |
|---|---|
| **Deployment costs during integration** | ——— |
| *First Procurement Cycle* | 23.94% |
| *Remaining Procurement Cycles* | 3.94% |
| **Operational costs during integration** | 33.33% |
| **Deployment costs after integration** | 3.94% |
| **Operational costs after integration** | 0.00% |

Summary of the financial profile of cost increases for integrating IP-enable relays to replace an existing Bent-Pipe relay fleet. Note, this profile is contingent on the model inputs used in this study.

In summary, with the deployment plan used in this research, the financial profile (and effectively the breakdown of costs associated with integrating IP-enabled relays) is shown in Table 4.2.

### 4.2.1 Sensitivity Analysis Definition

When addressing our second research question, we used a simple and convenient set of input variables for the deployment plan model. These inputs include: Spacecraft Lifetime, Phasing Offset, Procurement Cycle Length, and Deployment Frequency. Changing these inputs, however, has an impact on the model's outputs and, therefore, effectively impacts the financial profile. To demonstrate the impacts of changing these inputs, we do a Variance-based sensitivity analysis of the outputs for different permutations of inputs. In total, we determine the Mean, Variance, and Standard Deviation of 15 different sensitivity tests. These include: changing 1 variable at a time (4 tests); changing 2 variables at a time (6 tests); changing 3 variables at a time (4 tests) and; changing all variables at a time (1 test). During these tests, when an input remains static, it stays constant at the value used in the permutation from our research (15, 1, 5, and 3 respectively). When an input is non-static, it is given a range of 1 through 30. Since the permutation used in our research has inflexible relationships among its inputs, we make certain assumptions in order to expand our model. We define each input and its assumptions below:

**Spacecraft Lifetime:** The length at which a spacecraft remains in operations. Given a spacecraft launched at time $T$ and a Spacecraft Lifetime (let's call it $SL$), then that spacecraft will retire (and will no longer influence operational costs) at time $T + SL$. Since integration continues until the last bent-pipe relay retires, Spacecraft Lifetime has an impact on the length of time which integration occurs.

**Phasing Offset:** The number of years that integration cascades the original bent-pipe plan. In other words, since IP-enabled spacecraft are being phased in, there are a number of years where bent-pipe procurement activities take place side-by-side with the new IP-enabled procurement activities. We assume each of these bent-pipe years' costs are divided across the first procurement cycle (like they were for the permutation used in our research). Given a Phasing Offset larger than Procurement Cycle Length, we collapse the deployment costs of all overlapping years and divide it across the first procurement cycle. Also, since we can't offset to a time before bent-pipe spacecraft were launched, when offset is greater than Spacecraft Lifetime and Procurement Cycle Length, Phasing Offset is set to be the largest of the two. This is because the largest of these inputs determines how many years prior to integration the first bent-pipe relay was launched.

**Procurement Cycle Length:** The number of years under which a specified number of spacecraft are procured. Like Spacecraft Lifetime, this variable also has an influence on the length of integration. If the last bent-pipe relay retires before a procurement cycle ends, integration continues to the end of that cycle. However, if Spacecraft Lifetime is a multiple of Procurement Cycle Length, then integration ends the same year as the last bent-pipe relay retires.

**Deployment Frequency:** The number of spacecraft procured/launched each procurement cycle. This variable assumes that the same number of spacecraft are launched each cycle and that the pattern of launches remains consistent; with launches being divided evenly across each year with preference being given to later years. For example, with 5 year cycles and a Deployment Frequency of 3, launches occur as follows (from year 1 to 5): 0, 0, 1, 1, and 1. Assuming Deployment Frequency increases to 6, the launches will occur as follows: 1, 1, 1, 1, and 2.

In each permutation of our model, we reshape the timeline so that integration begins at time $T = 0$. We also assume that a full fleet (meaning the number of spacecraft expected to exist after integration - in our research this was 9 spacecraft) has been launched prior to integration. These and other factors cause the timeline to expand and contract in length, making side-by-side permutation comparisons difficult. Fortunately, we are able to collapse our analysis into certain areas of interest which we call hot-spots. Each year in a given hot-spot has the same cost, which allows us to run our sensitivity analysis on a single value from each hot-spot. This eliminates the problem of comparing varied length timelines. We define the hot-spots and their assumptions below:

**Pre-Integration:** The period of time preceding integration where both plans have only bent-pipe spacecraft. Since both plans yield identical deployment and operational costs prior to $T = 0$, this hot-spot exhibits a 0.00% increase regardless of the inputs given. This gives a Variance and Standard Deviation of 0 and therefore we remove it from our sensitivity analysis.

**First Integration Procurement Cycle:** The first procurement cycle of integration beginning at $T = 0$ and ending at $T = ProcurementCycleLength - 1$. This hot-spot is only concerned with deployment costs. The first integration cycle differs from the remaining integration cycles in that it includes the cost of any overlapping costs from the Phasing Offset. Since this hot-spot has the potential to change from one permutation to the next, we include it in our sensitivity analysis.

**Remaining Integration Procurement Cycles:** The period of time consisting of all procurement cycles during integration (excluding the first cycle). This hot-spot is only concerned with deployment costs. What's interesting about this hot-spot is that the percent increase for deployment and operations is the same, regardless of the inputs given. This hot-spot gives a Variance and Standard Deviation of 0 and therefore we remove it from our sensitivity analysis.

**Integration Operations:** The period of time consisting of all procurement cycles during integration. This hot-spot is only concerned with operational costs. Throughout integration, more operators are required in order to support the new and legacy spacecraft. This hot-spot begins one year prior to the first IP-enabled spacecraft launch. We do this to represent the time needed for training and on-boarding prior to mission-critical operations. Should the given permutation cause the first spacecraft to launch at $T = 0$, we assume no time is available for training and that the additional operators are brought onboard at $T = 0$. Regardless of where it starts, this hot-spot lasts until the end of integration. For most permutations, the year-to-year operational increases remain constant (thus the definition of a hot-spot). However, certain permutations exhibit slight jitter for Integration Operations. This jitter, however, can be ignored. We explain this in more detail later. Since this hot-spot has the potential to change from one permutation to the next, we include it in our sensitivity analysis.

**Post-Integration:** The period of time occurring after integration has finished. During this time, the fleet size of both plans, regardless of the inputs given, should be the same. Similar to the last hot-spot, certain permutations demonstrate slight jitter. which can be explained and ignored. This hot-spot gives a Variance and Standard Deviation of 0 and therefore we remove it from our sensitivity analysis.

Figure 4.9 shows the deployment and operational cost increases for the permutation used in our research. As you can see, we identify two points: $T = 0$ where integration begins and $T = 14$ where integration ends. Of the five hot-spots listed above, only two are relevant for the sensitivity analysis: First Integration Procurement Cycle and Integration Operations. The first can be seen on the Deployment Increase plot while $0 \leq T \leq 4$. The latter can be seen on the Operational Increase plot while $1 \leq T \leq 14$.

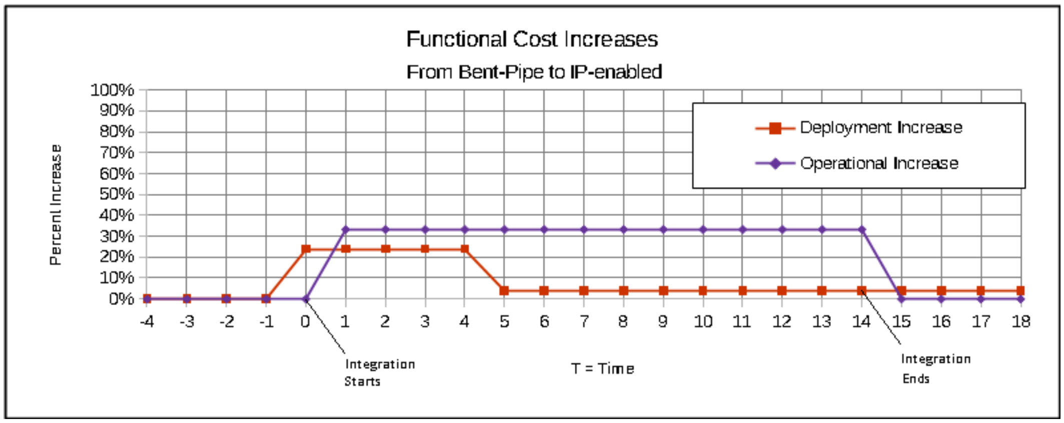

Figure 4.9: *Cost Increases from Research Permutation*

Graph of cost increases by function for the financial profile generated with inputs: Spacecraft Lifetime = 15; Phasing Offset = 1; Procurement Cycle Length = 5; Deployment Frequency = 3.

As mentioned earlier, certain hot-spots sometimes experience jitter. This jitter occurs due to the way certain permutations influence the number of operators required. When jitter occurs in the Integration Operations hot-spot, it's due to the constant ”hiring and firing” of operators. If you recall, we addressed this problem for our model by maintaining the maximum number of operators throughout integration. Though we did the same for each permutation's integration plan, we did not do this for the bent-pipe plan. To reconcile this, we eliminate the jitter by using the smallest output from the hot-spot. This effectively gives the cost increase where the maximum number of operators is maintained for both plans.

Figure 4.10 illustrates a permutation ($Spacecraft\ Lifetime = 3$, $Phasing\ Offset = 1$, $Procurement\ Cycle\ Length = 12$, and $Deployment\ Frequency = 29$) where Integration Operation's jitter is present. First, notice that integration ends at $T = 11$ for this graph. Because the last bent-pipe spacecraft retires prior to the end of the first procurement cycle, we continue to the end of that cycle. Now, while $3 \leq T \leq 7$, we see that the cost increase rises drastically then falls again. This is understood more easily by looking at the raw outputs in Table 4.3. By observing the values highlighted in light red, you can see how the integration plan has removed the ”hiring and firing” problem, but the bent-pipe plan has not.

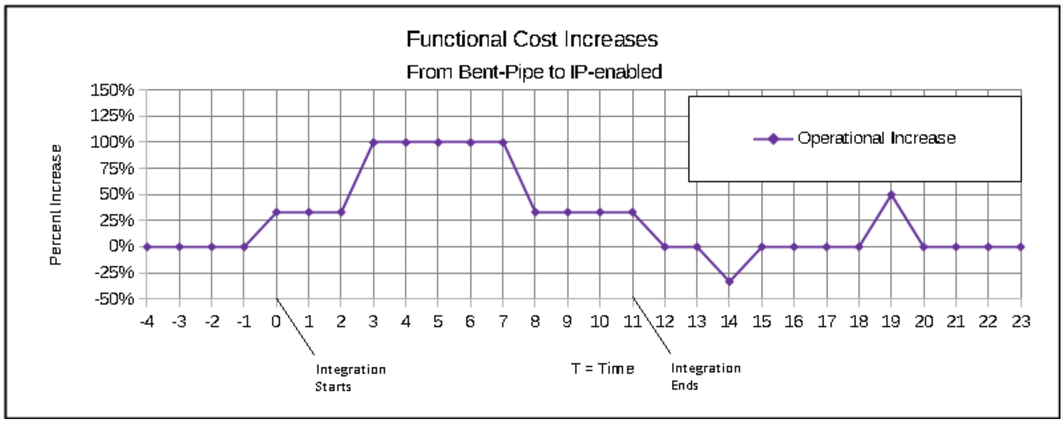

Figure 4.10: *Cost Increases for Permutation with Jitter*

Graph of operational cost increases for the financial profile generated with inputs: Spacecraft Lifetime = 3; Phasing Offset = 1; Procurement Cycle Length = 12; Deployment Frequency = 29.

Also in Figure 4.10, you can see jitter in the Post-Integration hot-spot which occurs while $T \geq 12$. The key drivers for this type of jitter are Phasing Offsets and the ”hiring and firing” problem. Since the bent-pipe plan wasn’t adjusted to maintain the max number of operators year-by-year, the offset results in the highs and lows of each plan being staggered. This means the jitter for this hot-spot is merely an alignment issue that is removed when the ”hiring and firing” problem is removed. This is understood more easily by looking at the raw outputs in Table 4.3. By observing the values highlighted in light green, you can see where the two are staggered by the 1-year offset.

Table 4.3: *Operational Costs for Permutation with Jitter*

| Year | Ongoing Bent-Pipe Plan | Phased Integration Plan |
|---|---|---|
| 2 | $393,000 | $524,000 |
| 3 | $262,000 | $524,000 |
| 4 | $262,000 | $524,000 |
| 5 | $262,000 | $524,000 |
| 6 | $262,000 | $524,000 |
| 7 | $262,000 | $524,000 |
| 8 | $393,000 | $524,000 |
| ... 9, 10, 11 and 12 ... | | |
| 13 | $393,000 | $393,000 |
| 14 | $393,000 | $262,000 |
| 15 | $262,000 | $262,000 |
| 16 | $262,000 | $262,000 |
| 17 | $262,000 | $262,000 |
| 18 | $262,000 | $262,000 |
| 19 | $262,000 | $393,000 |
| 20 | $393,000 | $393,000 |

Operational costs for a financial profile with inputs: Spacecraft Lifetime = 3; Phasing Offset = 1; Procurement Cycle Length = 12; Deployment Frequency = 29. Red values show integration operations jitter. Green values show post-integration jitter.

### 4.2.2 Sensitivity Analysis Results

Here, we give the results of our sensitivity analysis and discuss the implications of our findings. The results of each test are given in two different tables. Table 4.4 gives the Mean, Variance, and Standard Deviations of each test run on the First Integration Procurement Cycle hot-spot while Table 4.6 gives the same calculations for the Integration Operations hot-spot. Both tables are laid out in the same way; with the identification of static and non-static inputs on the left-hand side and statistical outputs on the right. Each test is given in its own row with non-static inputs denoted by a green checkmark and static inputs denoted by a red X.

Table 4.4: *Sensitivity Tests for First Integration Procurement Cycle*

| Which Variables Were Changed | | | | First Integration Procurement Cycle | | |
|---|---|---|---|---|---|---|
| **Lifetime** | **Offset** | **Cycle** | **Frequency** | **Mean** | **Var** | **SD** |
| ✓ | ✗ | ✗ | ✗ | 23.94 | 0.00 | 0.00 |
| ✗ | ✓ | ✗ | ✗ | 233.94 | 8633.33 | 92.92 |
| ✗ | ✗ | ✓ | ✗ | 3.94 | 360.05 | 18.98 |
| ✗ | ✗ | ✗ | ✓ | 23.94 | 0.00 | 0.00 |
| ✓ | ✓ | ✗ | ✗ | 218.94 | 18541.67 | 136.17 |
| ✓ | ✗ | ✓ | ✗ | 17.25 | 360.05 | 18.98 |
| ✓ | ✗ | ✗ | ✓ | 23.94 | 0.00 | 0.00 |
| ✗ | ✓ | ✓ | ✗ | 165.66 | 57774.56 | 240.36 |
| ✗ | ✓ | ✗ | ✓ | 233.94 | 8633.33 | 92.92 |
| ✗ | ✗ | ✓ | ✓ | 17.25 | 360.05 | 18.98 |
| ✓ | ✓ | ✓ | ✗ | 157.27 | 63839.94 | 252.67 |
| ✓ | ✓ | ✗ | ✓ | 206.90 | 17072.70 | 130.66 |
| ✓ | ✗ | ✓ | ✓ | 17.25 | 360.05 | 18.98 |
| ✗ | ✓ | ✓ | ✓ | 155.56 | 57107.85 | 238.97 |
| ✓ | ✓ | ✓ | ✓ | 152.10 | 63552.44 | 252.10 |

*Mean, Variance, and Standard Deviation outputs from all tests run for the sensitivity analysis of the First Integration Procurement Cycle hot-spot. A checkmark represents a non-static input and an X represents a static input. The red highlighted row represents the test with the most spread. Green rows represent the tests with the 2nd highest spread. Note, the 2nd highest rows do not include tests with non-static Phasing Offset.*

We begin by analyzing results from the First Integration Procurement Cycle. Notice in Table 4.4 that the first four rows represent ”1-variable at a time” tests. This means that all but one input remains static. Next we have rows five through ten which represent ”2-variables at a time” tests. Here, each test is run with two static and two non-static inputs. Rows eleven through fourteen represent the ”3-variables at a time” tests which, as you may have guessed, have one static and three non-static inputs. Lastly, we have row fifteen which is the ”all-variables at a time” test. For this test, all input are non-static.

Note that among these tests, we’ve highlighted multiple rows which we use to draw our conclusions. The row highlighted in light red represents the test having the largest Standard Deviation. This test is a 3-variables at a time test with a static Deployment Frequency. If you look closely at the table, you’ll notice that the rows which include a non-static Phasing Offset have the largest spread. This makes sense if you consider that for every additional year the provider offsets for integration, they will have an additional year’s deployment cost included in the First Integration Procurement Cycle. For those rows highlighted in light green, we originally intended to highlight the test with the second largest Standard Deviation. However, since Phasing Offset dominates this hot-spot, we feel it sheds more insight about the other inputs if we select the test with the second largest spread from among those with static Phasing Offset. As it turns out, there are four different tests with that share the second largest Standard Deviation. To narrow it down, we decided to go with the test(s) having the largest Mean.

If we assume that each test is normally distributed, we can derive the probability that a permutation selected by a provider (assuming the same range of inputs) has a hot-spot cost increase greater than or equal to the results given in our financial profile. For the First Integration Procurement Cycle, the financial profile gave a 23.94% increase. To calculate this probability, we use the Cumulative Distribution Function to evaluate the percent of permutations that fall between two distinct outputs along the distribution. We set these points as ranges, each starting at 23.94% and each ending at a given number of Standard Deviations above the Mean. Table 4.5 shows the results of running these calculations.

Table 4.5: *Worst-Case Cost Variance for First Integration Procurement Cycle*

| **Statistic** | **Test With 1st Highest Variance** | | |
|---|---|---|---|
| | 1 Std Dev | 2 Std Dev | 3 Std Dev |
| Percent of Permutations | 37.91% | 51.50% | 53.64% |
| Range of Outputs | 23.94 - 276.61 | 23.94 - 529.28 | 23.94 - 781.95 |

| **Statistic** | **Test With 2nd Highest Variance.*** | | |
|---|---|---|---|
| | 1 Std Dev | 2 Std Dev | 3 Std Dev |
| Percent of Permutations | 73.77% | 87.37% | 89.51% |
| Range of Outputs | 23.94 - 43.92 | 23.94 - 63.90 | 23.94 - 83.88 |

Gives the probability (as a percent) that a random permutation will fall within a range of cost increases bounded by the financial profile output and standard deviation(s) above the Mean for the First Integration Procurement Cycle hot-spot. This is done for the two tests from the First Integration Procurement Cycle which have the largest spread. * *The second test omitted non-static Phasing Offsets as an option*

There are several conclusions we can glean from this portion of our analysis. We begin by looking at the test with the largest spread. First, we've already identified that Phasing Offset has a significant impact on the First Integration Procurement Cycle. Second, we look at the outputs in Table 4.5. Consider a provider using our model that is able to change any input except Deployment Frequency. Given this distribution, we can say there's an approximate probability of 37.91% that the cost increase between the bent-pipe plan and integration plan will be ≥ 23.94% and ≤ 276.61%. To expand this, we can also say there's approximately a 13.59% (51.50% − 37.91%) probability that the increase would be > 276.61% and ≤ 529.28% and a 2.14% probability that the increase will be > 529.28% and ≤ 781.95%. These probabilities give insight into the likelihood that a provider might pay more than the estimate in our financial profile. This particular test is interesting because it helps to outline expectations for the worst-case deployment plans.

Next we look at the rows highlighted in light green. These tests help to illustrate the worst-case scenario given a static Phasing Offset. Looking closely at the table, it makes sense why these tests have the same output. Each row has a non-static input in common;

Procurement Cycle Length. If you look at the 1-variable at a time tests, you'll notice that Spacecraft Lifetime and Deployment Frequency have a Standard Deviation of 0. Since neither inputs has any impact on spread by itself, it makes sense that neither would impact spread when coupled with Procurement Cycle Length. In Table 4.5, you can see how the percent of permutations that fall within each range are much less than they are for the previous test. For instance, 73.77% (rather than 37.91%) of the permutations fell between 23.94% and the first Standard Deviation. Likewise, the stark difference on the first range's upper bound (43.92% vs 276.61%) is more reasonable and gives the provider more certainty about the const increase difference from our financial profile and a different deployment plan.

Before we move on to the next hot-spot, there are some anomalies in our results which may cause questions. First, since non-static Spacecraft Lifetime did not impact the spread in any of the tests we've looked at, you might expect that it works the same in all tests. However, if you look at the second and fifth rows down, there is a distinct difference in spread. The key reason for this is the way our model treats relatively large Phasing Offsets. Recall that any time an offset is greater than lifetime, the model sets offset equal to lifetime. By allowing both Spacecraft Lifetime and Phasing Offset to change across permutations in the distribution, we ultimately see a larger population of offsets with a much larger spread. This skew in spread is true for any test having both inputs set as non-static.

The Last anomaly from Table 4.4 is in the last row. Given that Deployment Frequency (which has had no impact on spread in any other test) is the only difference between this test and the test highlighted in light red. You might expect the two to have the same Standard Deviation, however, the result here is slightly less than the highlighted row. The difference is due to the increased population size from one test to the next. For every non-static input we add to a test, the resulting number of permutations grows by magnitudes. Since this row is the all-variables at a time test, the population size (and with it, the granularity of our statistics) increases.

Table 4.6: *Sensitivity Tests for Integration Operations*

| **Which Variables Were Changed** | | | | **Integration Operations** | | |
|---|---|---|---|---|---|---|
| **Lifetime** | **Offset** | **Cycle** | **Frequency** | **Mean** | **Var** | **Std. Dev.** |
| ✓ | ✗ | ✗ | ✗ | 24.50 | 891.14 | 29.85 |
| ✗ | ✓ | ✗ | ✗ | 87.78 | 480.25 | 21.91 |
| ✗ | ✗ | ✓ | ✗ | 20.42 | 16.89 | 4.11 |
| ✗ | ✗ | ✗ | ✓ | 15.80 | 42.62 | 6.53 |
| ✓ | ✓ | ✗ | ✗ | 73.11 | 819.85 | 28.63 |
| ✓ | ✗ | ✓ | ✗ | 15.98 | 468.25 | 21.64 |
| ✓ | ✗ | ✗ | ✓ | 18.67 | 432.90 | 20.81 |
| ✗ | ✓ | ✓ | ✗ | 78.20 | 769.14 | 27.73 |
| ✗ | ✓ | ✗ | ✓ | 80.79 | 760.84 | 27.58 |
| ✗ | ✗ | ✓ | ✓ | 10.98 | 75.16 | 8.67 |
| ✓ | ✓ | ✓ | ✗ | 74.97 | 882.67 | 29.71 |
| ✓ | ✓ | ✗ | ✓ | 70.32 | 751.22 | 27.41 |
| ✓ | ✗ | ✓ | ✓ | 14.71 | 367.24 | 19.16 |
| ✗ | ✓ | ✓ | ✓ | 69.95 | 931.30 | 30.52 |
| ✓ | ✓ | ✓ | ✓ | 69.17 | 974.96 | 31.22 |

*Mean, Variance, and Standard Deviation outputs from all tests run for the sensitivity analysis of the Integration Operations hot-spot. A checkmark represents a non-static input and an X represents a static input. The red highlighted row represents the test with the most spread. The green row represents the test with the 2nd highest spread.*

Next, we look at the sensitivity test results from the Integration Operations hot-spot shown in Table 4.6. As mentioned above, the tests in this table are laid out the same way they were for the previous hot-spot. Unlike the results from our last set of tests, each of the 1-variable at a time tests ran for Integration Operations results in Standard Deviation > 0. Likewise, each subsequent test (given in the fifth through fifteenth rows) has a spread different than any of its sub-tests.

No particular input dominates the tests in this hot-spot. For this reason, we highlight and analyzing the two tests which have the largest spread, regardless of which inputs are and are not static. Among the tests, the all-variables at a time test (highlighted in light red) has the largest spread. The second largest spread comes from the 3-variables at a time test (highlighted in light green) with a static Spacecraft Lifetime.

As we did for the last hot-spot, we assume a normal distribution for each test and use the Cumulative Distribution Function to derive the probability that a permutation will fall within given ranges of cost increases. From our financial profile, we know that the percent increase for Integration Operations is 33.33%. Table 4.7 gives the outcomes of the Cumulative Distribution calculations.

Table 4.7: *Worst-Case Cost Variance for Integration Operations*

| **Statistic** | **Test With 1st Highest Variance** | | |
|---|---|---|---|
| | 1 Std Dev | 2 Std Dev | 3 Std Dev |
| Percent of Permutations | 69.85% | 83.84% | 85.58% |
| Range of Outputs | 33.33 - 64.55 | 33.33 - 95.77 | 33.33 - 126.99 |

| **Statistic** | **Test With 2nd Highest Variance.**[*] | | |
|---|---|---|---|
| | 1 Std Dev | 2 Std Dev | 3 Std Dev |
| Percent of Permutations | 70.36% | 83.95% | 86.09% |
| Range of Outputs | 33.33 - 63.85 | 33.33 - 94.37 | 33.33 - 124.89 |

Gives the probability (as a percent) that a random permutation will fall within a range of cost increases bounded by the financial profile output and standard deviation(s) above the Mean for the Integration Operations hot-spot. This is done for the two tests from the First Integration Procurement Cycle which have the largest spread.

Lastly, we draw conclusions regarding Integration Operations cost increases using the probabilities in Table 4.7. Notice that the differences between the two tests is much less drastic than it was for the last set of tests. Since these tests help represent the worst-case scenario, their similar outputs give more certainty about the risk that cost increases will be larger than those given in our financial profile. For Integration Operations, we can say that in the worst-case, approximately 69.85% of all permutations will have an operator cost increase ≥ 33.33% and ≤ 64.55%. Likewise, in the worst-case approximately 13.59% will have an increase > 64.55% and ≤ 95.77% and approximately 2.14% will have an increase > 95.77% and ≤ 126.99%.

# 5 FURTHER RESEARCH

Throughout this thesis, we've brought up certain topics that could greatly enhance the fields of SI and SSI, but were beyond the scope of our study. In this section, we present these topics as areas of further research for future studies.

While discussing the layered approach to protocol abstraction, we briefly covered the concepts of static and dynamic routing. Depending on the specific needs or restrictions of network users, terrestrial networks will often vary in their topologies and the way their nodes obtain routing tables. Since this will likely be true for future SI and SSI networks, future research should consider the theoretical and practical approaches to IP-enabled relay routing. By theoretical, we refer to finding the best approaches for various nodal configurations, regardless of the current and/or planned spacecraft hardware. By practical, we refer to finding the same, but under the constraints of current and/or planned spacecraft hardware.

On the same note, as SI and SSI topologies and requirements grow increasingly more complex, the future of physical spacecraft hardware will need to adapt. We discussed one such example in this thesis: opportunistic, ad-hoc communications requires enhanced phased array antennas. Future research should consider the desired network capabilities of SI and SSI, determine the limiting qualities of current equipment, and investigate the potential for overcoming these limitations.

As discussed earlier, security is an important component of information and telecommunication systems. This holds especially true in sensitive, mission-critical environments like space communication. Though we addressed the TT&C requirements set by NASA Systems Engineers, we did not address the security of the protocols in our study. Further research should be conducted to investigate IP management protocol's ability to meet industry-standard security requirements for spacecraft communication.

Our static analysis of IP management protocols revealed that each protocol is capable of meeting NASA TT&C requirements for geostationary relays. The analysis did not, however, address which protocol works best for managing geostationary relays, nor did it define the boundaries and limitations of each protocol to meet these requirements. As SI expands beyond geostationary distances, these boundaries and limitations become increasingly more important. For this reason, future SI and SSI IP management studies should consider and benchmark the strengths, weaknesses, boundaries, and limits of each protocol. It may be found that a new protocol is required for SSI.

It should be remembered that our study has focused solely on the provider-driven requirements of single-hop IP management. In other words, we have not considered the implications of using IP forwarding for the delivery of customer data nor have we considered how IP management requirements are affected in multi-node, multi-hop networks. This is a valuable consideration for providers before offering IP services to end-users. Future researchers should consider the implications of packetized multiplexing on customer data.

Regarding the financial analysis, it should be noted that phased integration is only one of many technology adoption / replacement methods that can be used to convert a bent-pipe fleet to an IP-enabled fleet. Future studies may want to define a financial profile using other systems integration paradigms to more accurately represent their specific needs. The same is true if the researcher hopes to use a less uniform deployment plan (without repeating cycles) than we used to address our research.

Lastly, we want to be clear that our financial profile represents a real-world provider, however, the outcomes are closely tied to the way a public entity would pay for the system redesign. Given a private entity, the financing of relay hardware and launches is likely to differ. For instance, without a solid, up-front budget, the provider is likely to take out loans to pay for deployment costs. This causes the private sector provider to consider interest rates that the public sector provider did not. Likewise, since the public sector provider is part of the government, dividing deployment costs across procurement cycles makes sense. The private sector provider, however, is more likely to use depreciation in their financial profile. Future research should consider how much a private sector relay provider would pay to replace their bent-pipe geostationary fleet with an IP-enabled geostationary fleet.

# REFERENCES

Baker, F., & fairhurst, G. (2014). *Ietf recommendations regarding active queue management, draft-ietf-aqm-recommendation-08, work in progress* (Tech. Rep.). IETF.

Bhasin, K., & Hayden, J. (2004). Evolutionary space communications architectures for human/robotic exploration and science missions. *AIP Conference Proceedings*, 893–904.

Cisco & Astrium. (2011). *Iris space router key capabilities and benefits.* Retrieved 05 15, 2014, from `https://www.cisco.com/web/strategy/docs/gov/IRIS_Space_Router_Key_Capabilities_and_Benefits_wp.pdf`

*Cisco 18400 space router.* (2010). Retrieved 10 19, 2011, from `http://www.cisco.com/web/strategy/docs/gov/datasheet_c78-632235.pdf`

Clark, A. C. (1961). *Extra-terrestrial relays*. Wireless World.

Cuevas, E., & Tang, Z. (2009). Considerations for the performance evaluation of a geostationary satellite with an on-board internet router. *IET and AIAA International Communications Satellite Systems Conference (ICSSC)*, 232.

Deering, S., & Hinden, R. (1998). *Internet protocol, version 6 (ipv6) specification* (Tech. Rep.). IETF.

*Delay-tolerant networking research group (dtnrg).* (n.d.). Retrieved 10 05, 2014, from `https://irtf.org/dtnrg`

Eddy, W. (n.d.). *Conversations on spacecraft engineering.* at NASA GRC, Personal Communication between 10 19, 2011 and 04 30, 2014.

Eddy, W., & Fuentes, M. (n.d.). *Conversations on geostationary spacecraft tt&c modelling.* at NASA GRC, Personal Communication between 10 19, 2011 and 11 30, 2012.

Fortescue, P., Swinerd, G., & Stark, J. (2011). *Spacecraft systems engineering*. West Sussex, UK: John Wiley and Sons.

Harlacher, M. (2002). The low power transceiver (lpt) for space applications. *Spaceops2002*.

*High-power spacecraft for the 21st century.* (n.d.). Retrieved 05 10, 2014, from `http://www.boeing.com/boeing/defense-space/space/bss/factsheets/601/601fleet.page`

Hogie, K., Criscuolo, E., & Parise, R. (2005). Using standard internet protocols and applications in space. computer networks. *Spaceops2002*, *47*, 603–650.

*Improving communications effectiveness and reducing costs with internet routing in space: A dod joint technology capabilities demonstration.* (2011). Retrieved 06 13, 2014, from `http://www.cisco.com/web/strategy/docs/gov/c11-644555_wp.pdf`

*International organization for standardization.* (n.d.). Retrieved 04 12, 2014, from `http://www.iso.org/iso/home.html`

*Internet routing in space (iris).* (n.d.). Retrieved 06 13, 2014, from `http://www.intelsatgeneral.com/service-offerings/hosted-payloads/heritage/internet-routing-space-iris`

IOAG, Space Internetworking Strategy Group (SISG). (2010a). *Operations concept for a solar system internetwork (ssi)* (Tech. Rep.). IOAG.

IOAG, Space Internetworking Strategy Group (SISG). (2010b). *Recommendations on a strategy for space internetworking* (Tech. Rep.). IOAG.

Ippolito, L. (2008). *Satellite communications systems engineering: Atmospheric effects, satellite link design and system performance.* Online Library: Wiley Pubishing.

Isreal, D., & Spinolo, M. (2004). An architecture for nasa space network ip services. *AIAA*.

Isreal, D., & Zillig, D. (2002). Candos experiment overview. *AIAA SpweOps200*.

Jackson, J. (2005). The interplanetary internet (networked space communications). *IEEE Spectrum*.

Johnson, J., Connary, J., Thompson, J., & Donner, P. (2009). Internet routing in space nms architecture. *IEEE Aerospace Conference*, 1-16.

Jones, R. (2009). *Disruption tolerant network technology validation report - dinet* (Tech. Rep.). NASA.

Joseph, J., & Lazbin, J. (2004). Tcp/ip router for space applications). *estc2004*.

King, R. (n.d.). *Internet routing in space (iris), ccsds.* Retrieved 05 15, 2014, from `http://public.ccsds.org/meetings/2010Spring/Tech/lunchtime/King-Cisco-ccsds-lunch-talk.pdf`

Kunath, R., Reinert, J., & Barnes, P. (n.d.). *Sn operator costing.* at NASA GRC, Personal Communication on 03 11,2011.

Mead, D. (1999). *Direct broadcast satellite communications: An mpeg enabled service*. New Jersey: Prentice Hall.

NASA. (2008). *Nasa procedural requirements* (Tech. Rep.).

NASA, Exploration and Space Communications Projects Division 450. (2007a). Space network handbook [Computer software manual]. Greenbelt, Maryland.

NASA, Exploration and Space Communications Projects Division 450. (2007b). Space network users guide (snug) [Computer software manual]. Greenbelt, Maryland.

NASA HQ. (2007). *Nasa systems engineering handbook.* Washington, D.C.: NASA HQ.

NASA, International Space Station (ISS). (2000). *International space station operations local area network (ops lan)* (Tech. Rep.). Housten, Texas: NASA JSFC.

NASA, Space Communications and Navicagion (SCAN). (2007). *Space communications and navigation (scan) architecture definition document version 0.8* (Tech. Rep.). District of Columbia: NASA HQ.

NASA, Space Communications and Navicagion (SCAN). (2010). *Space communications and navigation (scan) integrated network architecture definition document (add): Volume 1 executive summary* (Tech. Rep.). District of Columbia: NASA HQ.

NASA, Space Communications and Navicagion (SCAN). (TBD 2011). *Space communications and navigation (scan) network systems requirement document (srd): Version 1, revision 1* (Tech. Rep.). United States: NASA SCaN.

*National aeronautics and space administration media kit; tdrs-j.* (2006). Retrieved 05 10, 2014, from `http://tdrs.gsfc.nasa.gov/assets/files/PressKits/TDRS%20J.pdf`

Pechanec, J. (2007). *How the scp protocol works.* Retrieved 10 24, 2014, from `https://blogs.oracle.com/janp/entry/how_the_scp_protocol_works`

Postel, J. (1980). *User datagram protocol* (Tech. Rep.). IETF.

*Radiocommunication sector (itu-r).* (2011). Retrieved 09 30, 2011, from `http://www.itu.int/ITU-R/index.asp?category=information&rlink=rhome&lang=en`

Rash, J., Hogie, K., Criscuolo, E., & Parise, R. (2003). *Proposal to add an hdlc framing option)* (Tech. Rep.). Greenbelt, Maryland: NASA GSFC.

Rinne, T., & Ylonen, T. (2013). *scp(1) - linux man page.* Retrieved 10 24, 2014, from `http://linux.die.net/man/1/scp`

*Satellite communication.* (n.d.). Retrieved 03 06, 2012, from `http://www.sscspace.com/products-services/satellite-management-services/communications-1/satellite-communications-1`

Schier, J., Rush, J., Williams, W., & Vrotsos, P. (2005). Space communication architecture supporting exploration and science: Plans and studies for 2010-2030. *AIAA*.

*Senior network administrator salary in las cruces, nm.* (n.d.). Retrieved 05 10, 2014, from `http://www.indeed.com/salary?q1=Senior+Network+Administrator&l1=Las+Cruces%2C+New+Mexico`

*Services - inmarsat.* (n.d.). Retrieved 04 12, 2014, from `http://www.inmarsat.com/services/`

*Snmp version 3 (snmpv3) message format.* (2005). Retrieved 10 24, 2014, from `http://www.tcpipguide.com/free/t_SNMPVersion3SNMPv3MessageFormat.htm`

Sollins, K. (1992). *The tftp protocol (revision 2)* (Tech. Rep.). IETF.

Space Communication Architecture Working Group (SCAWG). (2006). *Nasa space communication and navigation architecture recommendations for 2005-2030* (Tech. Rep.). NASA.

*Space communication and navigation home page.* (n.d.). Retrieved 04 12, 2014, from `http://www.nasa.gov/content/scan-homepage/#.U0mM7ceSK_U`

*Space link extension (sle) data transfer services protocol and application program interface (api) specifications.* (n.d.). Retrieved 04 13, 2014, from `http://public.ccsds.org/sites/cwe/rids/Lists/RPA520RPA520/NASAUSOverview.aspx`

*Space segment.* (n.d.). Retrieved 10 3, 2011, from `http://esc.gsfc.nasa.gov/space-communications/sn-sne/space-segment.html`

Spearing, R., Regan, M., Hawes, M., Luther, M., Akers, G., Brackey, T., … Smith, P. (2005). *Communication and navigation capability roadmap executive summary* (Tech. Rep.). NASA.

*Tcp/ip guide.* (2005). Retrieved 10 24, 2014, from `http://www.tcpipguide.com/`

*Tracking and data relay satellite: Continuing the critical lifeline.* (n.d.). Retrieved 04 07, 2014, from `http://tdrs.gsfc.nasa.gov/`

*Tracking and data relay satellite (tdrs) characteristics.* (n.d.). Retrieved 05 10, 2014, from `http://www.nasa.gov/directorates/heo/scan/services/networks/txt_tdrs_characteristics.html#.U58DLqiSK_V`

*Tracking and data relay satellite (tdrs) - h, i, j.* (n.d.). Retrieved 05 10, 2014, from `http://msp.gsfc.nasa.gov/TUBE/tdrshij.htm`

*Understanding simple network management protocol (snmp) traps.* (2006). Retrieved 10 24, 2014, from `http://www.cisco.com/c/en/us/support/docs/ip/simple-network-management-protocol-snmp/7244-snmp-trap.html`

US Office of Personnel Management. (n.d.). *2014 general schedule (gs) locality pay tables.* Retrieved 05 10, 2014, from `https://www.opm.gov/policy-data-oversight/pay-leave/salaries-wages/2014/general-schedule/`

Welch, A., Brooks, D., Beering, D., Hoder, D., & Zernic, M. (1997). *Experimental results of running tcp/ip over atm on nasa acts hdr* (Tech. Rep.). Cleveland, Ohio: NASA GRC.

Wood, L. (n.d.). *Iris project - work with intelsat.* Retrieved 06 13, 2014, from `http://personal.ee.surrey.ac.uk/Personal/L.Wood/iris/`

Wood, L., Eddy, W., & Holliday, P. (2009). A bundle of problems. *IEEE Aerospace Conference*, 1-17.

Wood, L., Holliday, P., Floreani, D., & Eddy, W. (2009). Sharing the dream: The consensual networking hallucination offered by the bundle protocol. *IEEE ICUMT*, 1-2.

Wood, L., Ivancic, W., Eddy, W., Stewart, D., Northam, J., & Jackson, C. (2008). *Investigating operation of the internet in orbit:five years of collaboration around cleo* (Tech. Rep.). SSC Newsletter.

Wood, L., Peoples, C., Parr, G., Scotney, B., & Moore, A. (2007). *Tcps protocol radius: the distance where timers prevent communication* (Tech. Rep.). Delay Tolerant Networking session, Third International Workshop on Satellite and Space Communications.

Wyatt, J., Burleigh, S., Jones, R., Torgerson, L., & Wissler, S. (2009). Disruption tolerant networking flight validation experiment on nasas epoxi mission. *IEEE SPACOMM*, 187-196.

Ylonen, T., & Lonvick, C. (2006). *The secure shell (ssh) transport layer protocol* (Tech. Rep.). IETF.