An Approach for Measuring IP Security Performance in a Distributed Environment

Brett L. Chappell, David T. Marlow, Philip M. Irey IV, and Karen O'Donoghue

System Research and Technology Department
Combat Systems Branch
Naval Surface Warfare Center, Dahlgren Division
Dahlgren, Virginia 22448-5000
{chappellbl, marlowdt, ireypm, kodonog}@nswc.navy.mil

Abstract. The Navy needs to use Multi Level Security (MLS) techniques in an environment with increasing amount of real time computation brought about by increased automation requirements and new more complex operations. NSWC-DD has initiated testing of a security protocol based on the commercial standard, IPSEC, which is becoming available in Commercial Off The Shelf (COTS) computing products. IPSEC is viewed as a critical component towards providing MLS capabilities. Current implementations of IPSEC are implemented in software as part of the kernel system software. The system engineer must carefully develop security policies versus applying this technology in a brute force way. This paper describes the security issues, the IPSEC standard, testing performed at NSWC-DD and provides an approach to using this technology in the current resource constrained environment using today's COTS products.

1 Introduction

As the U.S. Navy develops new and upgraded shipboard systems for its surface combatants, reduction of total ownership costs has become an over-all driving concern. This has lead to a Navy-wide direction to reduce manning and use Commercial Off The Shelf (COTS) products. The reduced manning direction has lead to an over-all increase in the amount of computing needed, and the COTS mandate has lead to the use of commercially developed computers versus the previous use of computers which were designed to specification.

The Navy is expecting a great increase in the amount of computing required, not only due to the increased emphasis on automation as described, but also due to the increasing functionality that is being added to meet the needs of changing military requirements. For example, joint operations with nations that we were not previously allied with may be needed in the future. This presents the need to be able to quickly adjust to sharing (and to stop sharing) specific information with military units (e.g. ships) from a variety of countries. Thus, policies applied by the operational personnel must be able to enable (or disable) the sharing of specific information. In addition to the existing requirements of military security where information is classified (e.g. unclassified, confidential, secret,...) there are concerns of protecting all information from computer hackers who have the expertise with the military computers now that COTS computers are being used.

Currently, classified and mission critical applications reside on a physically separate platform from unclassified or non-mission critical applications, as shown in Figure 1. This approach is unacceptable for the future where increased computational requirements are forcing the use of information at a variety of classification levels by a large number of the shipboard computers. In addition, the network infrastructure used by mission critical computing units must have inherent redundancy in order to survive after battle damage or the failure of any one network component. Such network infrastructure cannot be replicated for each classification level with the cost constraints that are being imposed in all current and future developments. There are other reasons to migrate away from designs based on the physical separation of equipment. There is a shift throughout the Navy from the current "stove-pipe" systems to tightly integrated systems. Stovepipe designs result in non-interoperable subsystems, duplication of resources, and difficulty in incorporating changes that are needed to meet new requirements.

The nature of the envisioned computing environment is such that application to application multilevel security (MLS) is required. Figure 2 illustrates a system which provides application to application MLS without the use of physical separation. The long term goal is to provide security services, such as confidentiality and authentication, to applications. In Figure 2, the dotted and dashed lines illustrate communication streams between applications executing at different security levels. This implies that security mechanisms must be in place to secure the data as it is transferred across the networking infrastructure.

Another concern is to provide the maximum flexibility for resource management in order to dynamically reconfigure the binding of computing tasks to physical computers even after events occur that result in battle damage where computers are lost. This implies that all applications must be capable of running on any computer. Thus it is desired that any computer be capable of executing both classified and unclassified applications simultaneously.

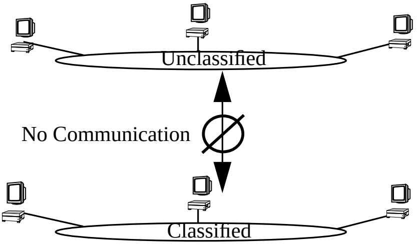

Figure 1. Infosec by physically separate networks

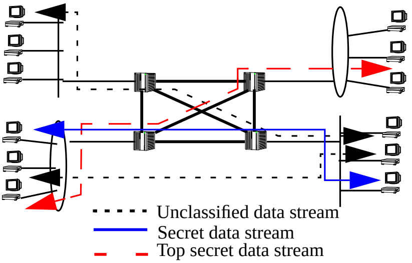

Figure 2. Application to application MLS

The IP Security Protocol (IPSEC) is a technology that may be used to provide security services at the network layer in a system such as the one shown in Figure 2. The protocol is an output of a working group within the Internet Engineering Task Force (IETF) whose charter is to provide security to the Internet Protocol (IP).[1, 2, 3] The focus of this paper is to develop a method in which the IPSEC protocol can be evaluated to determine its suitability for use in an MLS environment. The Naval Surface Laboratory (NRL) has provided an open source version of the IPSEC protocol which was used by NSWC-DD for experiments described in this paper.

2 IPSEC Overview

The primary services provided to the IP data packet by IPSEC are data confidentiality and authentication. In addition, IPSEC has been designed in such a way to minimize the risk of replay attacks and IP spoofing.

IP authentication allows the recipient of an IP packet to be certain that the contents of the headers and data field have not been modified. Also, the authentication service allows the recipient to be sure that the packet actually came from the host identified by the source IP address. IP confidentiality ensures that the data portion of the IP packet is not readable by unauthorized entities. Both the authentication and confidentiality services are achieved through the use of cryptographic techniques.

IPSEC uses a one way cryptographic hash function to provide authentication services.[2] The IPSEC protocol supports any number of one way hash functions. For interoperability, there is a function that is mandatory to implement. Currently, the mandatory function to implement is MD5, and the NRL implementation uses MD5 as the one way hash function.

In order to provide data confidentiality, IPSEC supports a number of encryption algorithms.[3] Currently, the Digital Encryption Standard (DES) is mandatory to implement for interoperability. The NRL implementation of IPSEC uses DES to provide the confidentiality services. DES is a symmetric key encryption algorithm, meaning that the same key is used to encrypt as well as decrypt the data. This implies that the sender and receiver must securely exchange the key. The IPSEC protocol includes a key exchange protocol [7], however, the NRL implementation used employs manual keying. A noteworthy event concerning DES occurred recently (mid-1998), where a system was built that is capable of cracking DES encrypted data by brute force in a matter of days.[4] The DES encryption algorithm will most likely be replaced by a more secure algorithm in the near future.

Before the IPSEC services can be invoked, a *security association (SA)* must to be established. A security association is a simplex connection that provides a specific IPSEC service to the traffic carried by it. An SA has three components: the type of service provided, either an Encapsulated Security Payload (ESP) service or Authentication service; the destination address of the channel; and a security parameters index (SPI). The ESP service provides data confidentiality for the data portion of an IP packet. The SPI is an integer number that is used by the end system to internally identify the communications stream. The three components together describe a unique SA. In order to communicate security between two systems using an IPSEC service, two SAs are required, one in each direction. Also, only one IPSEC service can be provided by a single SA, and this results in a separate SA for each IPSEC service desired.

2.1 IPSEC Implementation

A topic that deserves careful consideration is how and where to implement IPSEC. This section discusses some of the issues that should be considered when adding functionality to the IP layer.

IPSEC may be implemented by several different methods in a host. There are three common methods: 1) integration into the native IP stack of the operating system, 2) "bump-in-the-stack," meaning that IPSEC is implemented in software directly below the IP layer, and 3) "bump-in-the-wire," meaning that IPSEC is implemented in special hardware that acts as an external processor to the system.[1] The bump-in-the-wire implementation may be partially or completely separated from the resources of the computer. Figure 3 illustrates the three choices.

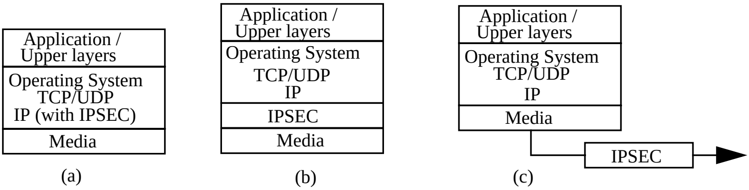

Figure 3. (a) Native IPSEC, (b) bump-in-the-stack, and (c) bump-in-the-wire

Each method of implementation has advantages and disadvantages. The first two methods listed are lower cost in terms of equipment as they are implemented in software. Integration of IPSEC into the native IP stack requires that source code for the operating system be available. Often, when dealing with COTS equipment, this code is either not available or prohibitively expensive. This method will most likely become the most popular method adopted by major OS vendors, such as Sun Microsystems or Microsoft. The second method, the bump-in-the-stack, is comparable to buying an enhancement to an operating system, possibly from a third-party vendor. If the operating system source code is not available, this may be a wise choice.

The third method may be the most expensive to produce after the design has been completed, since it requires additional hardware to be purchased with each computer system. However, specialized hardware implementations generally provide higher performance in processing cryptographic functions.

The NRL IPSEC implementation is a native IP stack implementation. Since the source code for some versions of Unix is freely obtainable (e.g., NetBSD), it is a suitable platform for an integrated implementation. However, a result of this type of implementation is that the cryptographic algorithms are implemented as system calls in the kernel. As will be noted later, this can have a significant impact on the proper operation of the kernel.

3 Testbed Configuration

The stand-alone test performed by NSWC-DD consisted of measuring the network performance of TCP connections between two Sun Sparcstation 20 workstations. The workstations have been configured with the NetBSD operating system, version 1.2. The operating system has been modified to integrate the alpha-five version of the IPSEC implementation from the Naval Research Laboratory (NRL) An in-house software package, the Communication Analysis and Simulation Tool (CAST), was integrated into the testbed by porting it to the NetBSD environment. CAST measures several different network performance related metrics, however, this paper will concentrate on two: end-to-end throughput and one-way latency.[5] Also, it is important to note that the NRL implementation of IPSEC has not been optimized for performance; it is a proof of concept that is still evolving.

3.1 IPSEC Configuration

There were three types of tests performed - one with no IPSEC service, one with IPSEC authentication, and one with IPSEC confidentiality. TCP connections between the sender and receiver were used for each of the three tests. Figure 4 illustrates the security associations required if two end systems desire to communicate using an IPSEC service. Specifically, the testbed configuration consisted of one Sparcstation 20, burke, connected to another Sparcstation 20, ollie, via 10 Mbps Ethernet and an Ethernet switch.

3.2 Network Time Protocol (NTP) Configuration

The CAST tool requires that the sender and receiver have synchronized clocks or that the offsets between the two clocks can be accurately measured. In addition, the frequency of the system clocks on the sender and receiver must change at the same rate. In this testbed, NTP version 3 was used to synchronize the system clocks. [6]

Figure 5 shows the configuration of NTP in this testbed. A GPS receiver is used to provide an accurate and stable notion of UTC time via an IRIG-B signal to an oscillator board in the time server. NTP was used to synchronize the system clock of the time server to the oscillator board and the system clocks of the two test machines to the system clock of the time server. For the purposes of this experiment, maintaining true UTC time is not required, however, this NTP configuration was available and used.

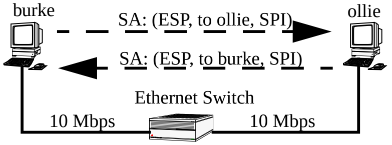

Figure 4. The testbed configuration with ESP security associations.

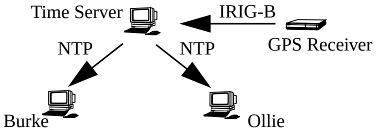

Figure 5. The NTP configuration

4 Analysis of results

The CAST tool has three phases of operation: 1) calculate clock offset, 2) calculate one-way latency, and 3) calculate end-to-end throughput. The calculated clock offset is used to compute both the one-way latency and end-to-end throughput phases.

End-to-end throughput is the amount of data that can be sent over the TCP connection in a time period. One-way latency is the amount of time that it takes to send data from one end of the connection to the other. Figures 6, 7, and 8 show the measured end-to-end throughput and one-way latency with no IPSEC being used, IPSEC authentication, and IPSEC confidentiality.

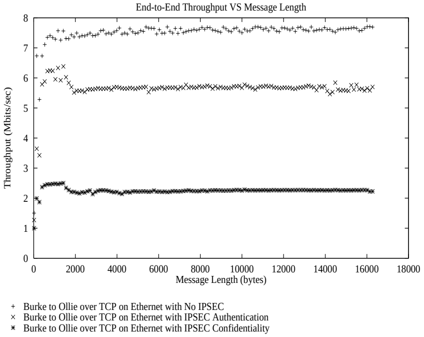

Figure 6. End-to-end throughput for the three tests from the sender to the receiver

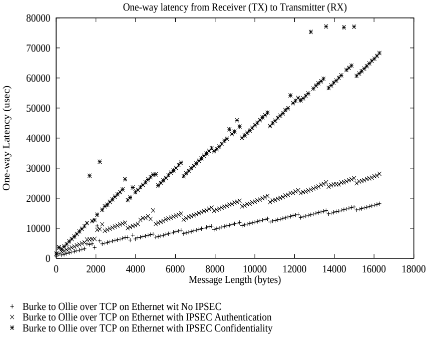

Figure 7. One-way latencies for the three tests from the sender to the receiver

4.1 Effect on End-to-end Throughput and One-way Latency

Figures 6, 7, and 8 show that both the end-to-end throughput and one-way latency results are significantly impacted when using the IPSEC protocols.

The effect upon the end-to-end throughput is a direct relationship between the three test runs. The one-way latencies were measured from the sender to the receiver, and vice-versa, as shown in Figures 7 and 8. In both cases, the one-way latencies increase when using the IPSEC authen-

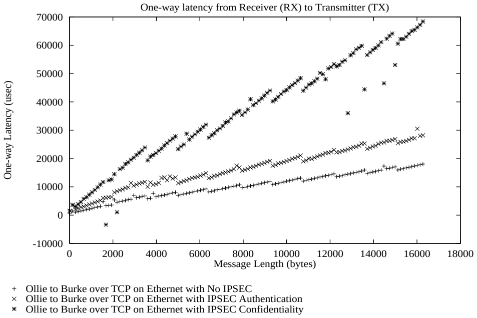

Figure 8. One-way latencies for the three tests from the receiver to the sender.

tication and the IPSEC confidentiality. In addition, as shown in Figures 9 and 10, the standard deviation in the latency measurements is increased when IPSEC is used, most notably when software encryption is employed.

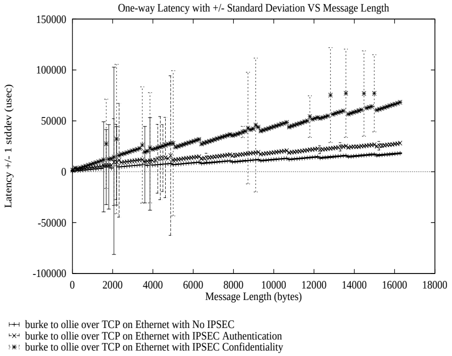

Figure 9. The standard deviation of the latencies between the sender and receiver

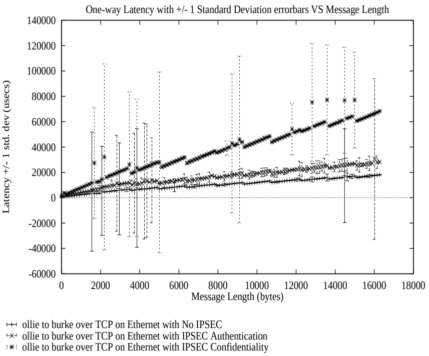

Figure 10. The standard deviation of the latencies between the receiver and the sender

5 Conclusions

A number of conclusion and lessons learned can be drawn from the results of this experiment. Based on the results, certain recommendations can be made on the use of the IPSEC protocols in future Navy systems.

First, it was shown that the IPSEC protocol can be used to secure communication between two end points. The beta release of the NRL code is a proof of concept that the protocol will work. Since many major operating system vendors have adopted IPSEC, such as Microsoft and a number of Unix vendors, it is a viable component of the solution to the application-to-application MLS requirement. However, it is important to realize that the total solution will be an integration of several components, such as IPSEC and operating system mechanisms.

Secondly, it was shown that the software implementation of IPSEC in the kernel significantly impacts network performance. In order to use such an implementation in an MLS environment as shown in Figure 2, a method must be developed to minimize the computational requirements. Specifically, real-time communications between two applications may not be possible while using the IPSEC services.

The method used to measure and analyze IPSEC performance is scalable beyond the single sender, single receiver example used throughout this paper. Since CAST is executed in the end system, this approach can used to evaluate IPSEC performance in a more distributed environment.

5.1 An Approach to Minimizing Computational Requirements used by IPSEC

As shown by the test results, a security policy which applies authentication and encryption on all information transferred is not appropriate in a resource constrained environment. The computational load of applying authentication and encryption is significant given today's COTS computing products. A straightforward approach to applying this technology with current products is to carefully choose the proper configuration of IPSEC that is well suited for the applications of interest. By trading off security with reduced performance in throughput and latency, a system engineer can work out a solution which balances the system's real-time requirements. In order to avoid unnecessary overhead, security policies need to be developed that describe which types of information flows require each type of protection. Also, a security management scheme to enforce these policies would be useful. For example, in most systems, the routine data being transferred is unclassified and does not require protection. In this situation, it is unnecessary and inefficient to encrypt and authenticate all data. It is possible that information exchanged within a given subsystem could be transferred with less IPSEC services than information exchanged between subsystems. Appropriate use of firewall technology may be needed in order to practically implement such an approach.

In the system shown in Figure 2, each system may have applications requiring multiple levels of service. The NRL implementation of IPSEC enables security associations (SAs) on a per-socket basis. Therefore, each application could potentially communicate with another application running on a different host using different data streams, each with a different security association via sockets, as shown in Figure 11.

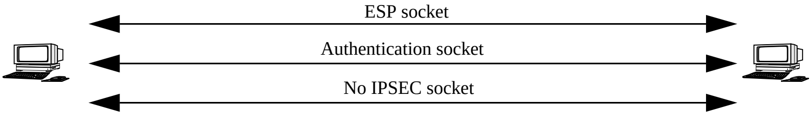

Figure 11. A simple approach to managing IPSEC security services

This type of approach will reduce the computational load required by IPSEC in order to provide more resources for real-time processing. However, it requires more effort in the analysis of the network design.

6 Future Work

During the testing of the NRL IPSEC implementation, observations were made about the system performance that are beyond the scope of this paper. Specifically, there is a significant impact upon basic system services, such as timekeeping and CPU scheduling, when software encryption is performed in the kernel. This impact is worthy of further exploration. In addition, an evaluation of the dynamic key management protocol of IPSEC is planned as the implementations mature.

7 References

1. Kent, S., and Atkinson, R., RFC2401, *Security Architecture for the Internet Protocol*, November, 1998.

2. Kent, S., and Atkinson, R., RFC2402, *IP Authentication Header*, November, 1998.

3. Kent, S., and Atkinson, R.,RFC2406, *IP Encapsulated Security Payload*, November, 1998.

4. Gilmore, J., *Cracking DES: Secrets of Encryption Research, Wiretap Politics, and Chip Design*, Electronic Frontier Foundation, July, 1998.

5. Irey, P., Harrison, R., Marlow, D., *Techniques for LAN Performance Analysis in a Real-Time Environment*, Real-Time Systems - International Journal of Time Critical Computing Systems, Volume 14, Number 1, pp. 21-44, January, 1998.

6. Mills, D., *RFC-1305, Network Time Protocol (Version 3) Specification, Implementation, and Analysis*, March, 1992.

7. Maughan, D., Schertler, M., Schneider, M., and Turner, J., RFC2408, *Internet Security Association and Key Management Protocol,* November9, 1998.