NIST Project SB1341-02-C-0081

Impact of Information Security Systems On Real-Time Process Control

Final Report

Prepared by

William F. Rush
and
Aakash Shah

Gas Technology Institute
1700 S. Mt. Prospect Road
Des Plaines, Illinois 60018-1804

GTI Project Numbers 61160 and 15335.1.01
FERC Project Numbers 30802-05 and 15063.1.01

For

National Institute of Standards and Technology

April 2005

Executive Summary

The objective of this program was to develop detailed procedures for evaluating the impact of cryptography in real-time control systems. The primary concern is that introducing encryption into such systems may increase slow communications to the point that the ability to operate in real time may be degraded unacceptably. To assess the impact of adding cryptography, GTI began with a list of the operating requirements of real-time control systems and developed a set of 13 tests. These tests may be roughly classified into three categories: functionality, performance, and operability tests. The tests developed under this project do not address the ability of the cryptographic system to withstand attacks.

The tests are described individually in a separate appendix for the convenience of the testing entity. Each test includes the rationale for the test, a summary of the technical background, a list of the required test equipment, the general approach to doing the test, a detailed test procedure, an example of typical test results presented as a completed test report form, notes on interpreting and analyzing the test results, and a blank test report form to be used by the person performing the test. Procedures include options for performing these tests in both laboratory and field environments. Methods by which the test results can be used to estimate performance in actual systems are also outlined.
In addition to the specific tests described, GTI developed two tools to assist in performing and understanding the tests. The first is software that automates some of the test procedures. This was deemed necessary when it became apparent that many of the test procedures required running the same test many times and collecting the data in files for subsequent analysis. Using this software greatly speeds the performance of the tests. The second tool is a computer program that models the latency of an encrypted message given the clear text message and the details of the cryptographic protocol. This is useful both because it establishes the maximum performance that can be expected of the protocol under ideal conditions and also enables estimates of the economic impact of the cryptographic protection on the control system. The test automation and message length programs are posted on a GTI web site for free download.

Table of Contents

Page

1 Introduction and Programmatic Outline 1
2 Rationale 4
3 Tests 4
3.A Overview 4
3.B List of Tests 5
3.C Types of Tests 5
3.C.1 Lab Tests 5
3.C.2 Testbed Evaluation 6
3.C.3 Field Tests 6
3.C.4 Performance Predictions 6
3.D Test Format 6
3.E Test Configurations 6
3.F Test Equipment 10
3.F.1 Hardware 10
3.F.2 Software 11
4 Models 13
4.A Operational Model 13
4.B Economic Model 18
5 Recommendations and Conclusions 19
Appendix A References 20
Appendix B Acronyms and Definitions 21
B.1 Acronyms 21
B.2 Definitions 21
Appendix C Test Procedures 25
C.1 Functionality Tests 25
C.1.1 Functionality Test 26
C.1.2 Jitter Tests 27
C.1.3 Interoperability Tests 34
C.1.4 Backup / Failover System Test 39
C.2 Performance Tests 43
This section describes the performance tests in detail. 43
C.2.1 Block Length Probing 44
C.2.2 Effect of Message Content on Latency 59
C.2.3 Throughput Tests 68
C.2.4 Clock Synchronization Test 82
C.2.5 Susceptibility of Adverse Conditions 89
C.2.6 Effect of Noise 90
C.3 Operability Tests 99
C.3.1 Bottleneck Identification 100
C.3.2 Regression Tests 107
C.3.3 Reliability Tests 110
Appendix D AGA 12, Part 2 RFC 120

1 Introduction and Programmatic Outline

The Program Objective Was to Develop Test Procedures

The objective of this program was to develop detailed procedures for evaluating the impact of cryptography in real-time control systems. In particular, the goals were to specify the test protocol, validate it in lab tests, carry out field measurements, and to analyze and publish the results. In addition, GTI was to develop a spreadsheet that predicts the operational and economic impact of using various cryptographic modules in conjunction with various communication systems.

This report is divided into two parts. After a brief introduction to the technical background of the project, the first section deals with the programmatic aspects of the project. The second section of the report contains a set of appendixes that are the recommended test procedures that are the primary technical results of the project.

Introduction and Programmatic Summary

It Is Important To Protect Process Control Systems

The small computers in process control systems make attractive potential targets for threat agents because these devices control systems and components that affect chemical processes, manufacturing operations, and utility operations. These computers and their associated communication links, designated as real-time process control systems here, control the flows of gas, water, and electric networks and chemical and manufacturing processes. Thus, a successful attack against such systems can have serious adverse economic and social impacts on society.

Process control computers differ from traditional information processing systems in several important respects. They are designed for low cost and time-critical applications with relatively little consideration for security. These small processors do not have the memory, computational power, or speed to support most cryptographic systems. This makes it difficult to add security to existing control system without incurring some performance degradation.

This Project Developed Recommended Test Procedures

It is important to have some method of assessing quantitatively how significantly the introduction of cryptographic protection will affect such systems – the need that led NIST to launch this project. The primary concern is that introducing encryption into such systems may increase slow communications to the point that the ability to operate in real time may be degraded unacceptably. A second concern is to determine whether the introduction of cryptography has negative impact on functionality of the process control system. For example, some systems operate under the control of a central host processor but include a back-up computer that assumes control of the system should the host be incapacitated. A poorly designed cryptographic system may lock up the system in the event of a fail-over from the host to the back-up. Thus, it is necessary to recognize several types of tests.

There are essentially three types of tests contained in this report: performance tests, functionality tests, and operability tests. Note that this report does not address the ability of the cryptographic system to withstand attacks. Functionality tests determine whether introducing cryptography into the control system still allows the system to function. These tests generally are "yes/no" tests that determine whether the cryptographic system under test does or does not interfere with the normal operation of the process control system. Performance tests compare the performance of the control system prior to introducing cryptography to the performance afterwards. These tests often have quantitative results that indicate the degradation numerically, such as how many fewer polls per time are possible after the introduction of cryptography than were possible before. Operability tests focus on longer term performance issues, such as reliability and changes introduced by manufacturers.

The Project Was Delayed by External Forces

Overall, the project was executed as planned in the original proposal, with the exception of a series of delays driven by the AGA 12 cryptographic project. The tests described in this project are generally applicable to evaluating any process control system using any cryptographic protocol. However, GTI evaluated the tests using a specific cryptographic protocol known as AGA 12. This protocol is being developed by the American Gas Association (AGA) and is designed for retrofit application to Supervisory Control And Data Acquisition (SCADA) systems. SCADA systems are a subset of real-time process control systems.

GTI chose to use the AGA 12 protocol as the prototypical process control system for several reasons. First, GTI is intimately involved in its development and is thus familiar with it. GTI is also testing the system in both the laboratory and the field, allowing for evaluation of the test protocol in a wide variety of applications over which GTI had control over the field test configurations. Finally, AGA 12 is designed for low speed (down to 300 baud) communications – conditions at which latency issues become most important.

While there were many benefits to using the AGA 12 system as the prototype, there proved to be one serious drawback – delays in developing the field test units. In particular, when the AGA 12 protocol was submitted to Sandia National Laboratory (SNL) for an evaluation of the security level of the protocol, a potential attack was recognized. As a result, the development of equipment that could be used in field tests was delayed while the security problems were remedied. GTI considered submitting this report prior to conducting actual field tests in the hope that field tests would not alter the test procedures, but decided that this was not a technically sound decision.

The field tests indicated that delaying release of the report was a technically important decision. In particular, while everyone in the AGA 12 group felt that increases in message length (as indicated by latency) were the best indicator of impact on process, this was not how system operators viewed performance. SCADA operators regard message throughput (messages per unit time) as the parameter of greatest importance. Field tests showed that the operators who claimed to be doing “continuous polling” and who believed that any increase in message length would result in a corresponding decrease in poll rate were pleasantly surprised by throughput results. In fact, when SCADA manufacturers claim continuous polling, they generally build into the system delays between polls to allow for corrections to noise, line delays, etc. As a result, adding even 100 milliseconds to a message may not decrease the throughput at all; the message simply “fits” into the delay that has been deliberately built into the system.

In summary, while the delays resulting from the development of the AGA 12 prototypes for field test delayed this project, the result was a set of test procedures that are better suited to evaluating of the cryptographic protection systems.

2 Rationale

The first step in developing the test protocol was to develop a rationale for determining which parameters must be measured. This was done in conjunction with the AGA 12 project. With the help of the AGA 12 committee members, GTI developed a list of operating realities of SCADA systems. These realities were reviewed by both gas and electric SCADA operators and updated several times. Once the list of operating realities was established, GTI and the AGA 12 group developed a list of implications (called constraints in the document) of each reality on the cryptographic system. Finally, once the requirements and constraints was developed, the AGA 12 Field Test Committee developed the series of tests to measure how well the cryptographic system under test compared to the user requirements.

The document listing the user requirements and associated cryptographic constraints is known as the AGA 12 RFC (for Request For Comment). While it is posted on the GTI web site, it has not been widely circulated. GTI believes this to be the best single starting point for anyone interested in developing a list of operating requirements for SCADA systems. The document is included as Appendix A to this report.

3 Tests

3.A *Overview*

To assess the impact of adding cryptography, GTI began with a list of the operating requirements of real-time control systems and developed a set of 13 tests. These tests are roughly classified into three categories: functionality, performance, and operability tests. Each test is described in a separate appendix for the convenience of the testing entity. Each test procedure includes –

- ➢ The rationale for the test
- ➢ A summary of the technical background
- ➢ A list of the required test equipment
- ➢ The general approach to doing the test
- ➢ A detailed test procedure
- ➢ An example of typical test results presented as a completed test report form
- ➢ Notes on interpreting and analyzing the test results, and
- ➢ A blank test report form to be used by the person performing the test.

Procedures include options for performing these tests in both laboratory and field environments. Methods by which the test results can be used to estimate performance in actual systems are also outlined.

In addition to the specific tests described, GTI developed two tools to assist in performing and understanding the tests. The first is software that automates some of the test procedures. This was deemed necessary when it became apparent that many of the test procedures required running the same test many times and collecting the data in files for subsequent analysis. Using this software greatly speeds the performance of the tests. The second tool is a computer program that calculates the length of an encrypted message given the clear text message and the details of the cryptographic protocol. This is useful both because it establishes the maximum performance that can be expected of the protocol under ideal conditions and also enables estimates of the economic impact of the cryptographic protection on the control system. The test automation and message length programs are posted on a GTI web site for free download.

3.B List of Tests

The tests required to effectively evaluate the impact of retrofit cryptographic modules on real-time control systems are listed below. The tests are divided into three categories – functionality, performance and operability tests. The tests and the respective test procedures are described in detail in Appendix C.

- C.1 Functionality Tests
    - C.1.1 Functionality Test
    - C.1.2 Jitter Tests
    - C.1.3 Interoperability Tests
    - C.1.4 Backup / Failover System Test
- C.2 Performance Tests
    - C.2.1 Block Length Probing
    - C.2.2 Effect of Message Content on Latency
    - C.2.3 Throughput Tests
    - C.2.4 Clock Synchronization Test
    - C.2.5 Susceptibility of adverse conditions
    - C.2.6 Effect of Noise
- C.3 Operability Tests
    - C.3.1 Bottleneck Identification
    - C.3.2 Regression Tests
    - C.3.3 Reliability Tests

3.C Types of Tests

As many as 4 steps may be required for a thorough evaluation of cryptographic protection of process control systems – Lab tests, test bed evaluations, field test evaluations, and performance predictions. While every evaluation will not necessarily entail all of these steps, this report details how to conduct each of these procedures and to understand the results of the evaluation. The four steps are described below.

3.C.1 Lab Tests

The lab tests focus entirely on the cryptographic modules and do not require any process control equipment.

3.C.2 Testbed Evaluation

The test bed evaluation tests the cryptographic modules with duplicate equipment used by the process control system.

3.C.3 Field Tests

The field test is conducted in the actual production system.

3.C.4 Performance Predictions

Performance prediction uses lab and/or test bed results as well as results from the operational model to predict/compare to field performance.

3.D Test Format

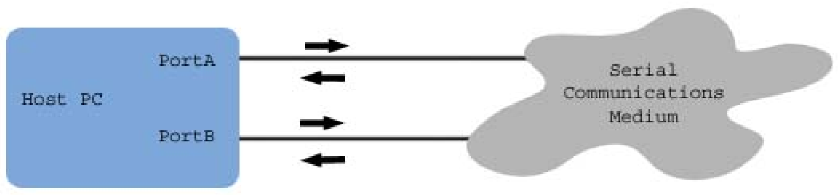
All of the test descriptions in Appendix C have a similar format. This format is outlined below.

1. Introduction
2. Purpose
3. Background
4. Test Equipment
5. Test Procedure
    a. General Test Procedure
    b. Detailed Test Procedure
6. Test Example
7. Interpreting and/or analyzing the results
8. Definitions
9. Blank Test Results Sheet

3.E Test Configurations

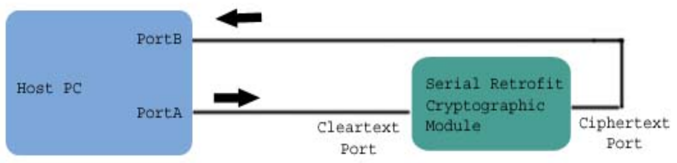
The test configurations needed for the different tests are described below. The communications medium used to communicate data from the master to the slave is not specified, as it will change from one test environment to another. For comprehensive test results, the tester should test all of the different communications media available, such as dial-up modem, leased line, radio and/or satellite. This will not only ensure functionality with the equipment associated with each communications media, but provide performance results unique to each communications medium.

Configuration 1 –Loop



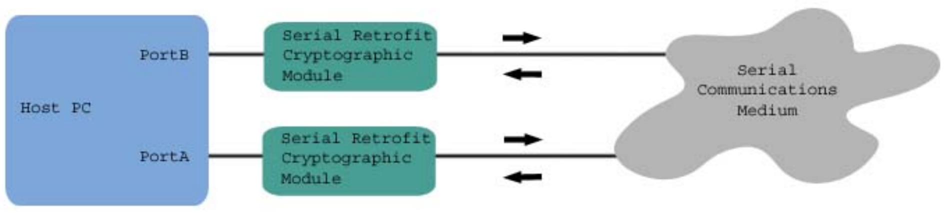
In this configuration the two ports on the PC are interconnected. A null modem is generally required to make this configuration function properly.

Configuration 2 – Single CM loop



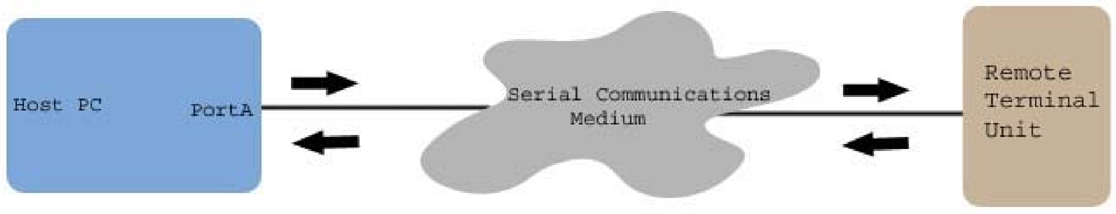
In this configuration, Port A of the PC connects to the cleartext port of the CM. The ciphertext port of the CM connects back to Port B of the PC.

Configuration 3- Secure loop


Host PC
PortB
PortA
Serial Retrofit Cryptographic Module
Serial Retrofit Cryptographic Module
Serial Communications Medium

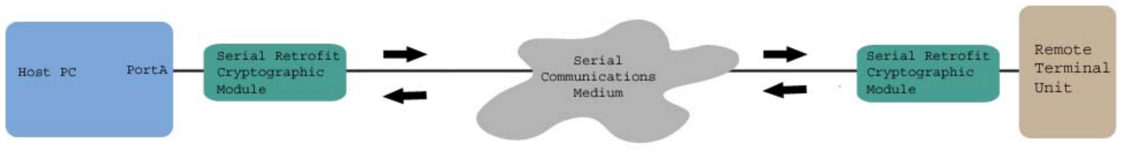
This is the cryptographically secure version of Configuration 1.

Configuration 4 – Point to Point



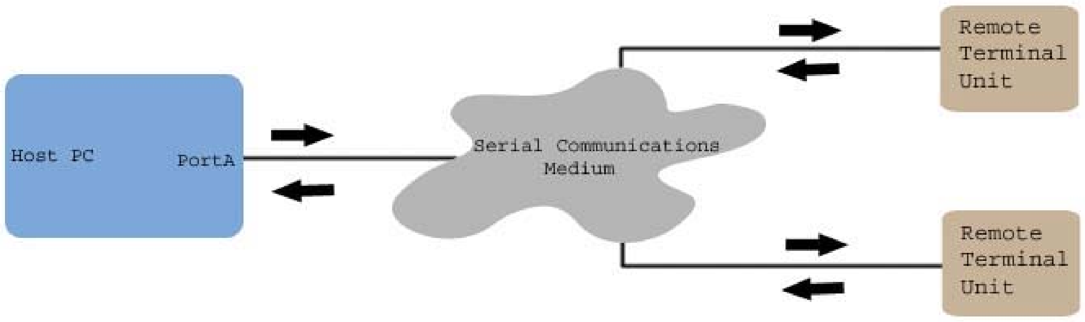
In this configuration, the real-time process control system master connects to a slave in a point to point connection.

__Configuration 5- Secure Point to Point__


Host PC
PortA
Serial Retrofit Cryptographic Module
Serial Communications Medium
Serial Retrofit Cryptographic Module
Remote Terminal Unit

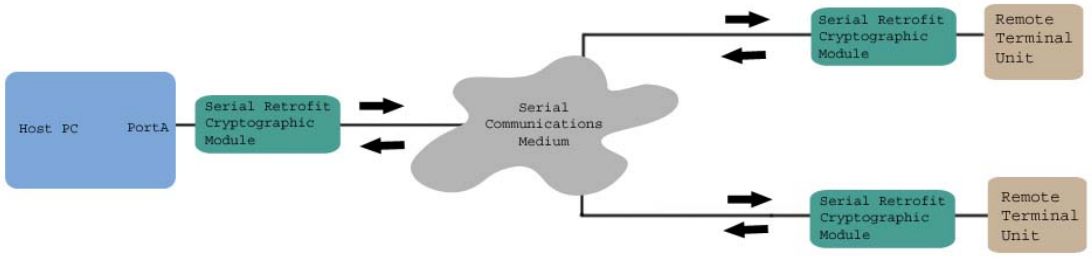
This is the cryptographically secure version of Configuration 4.

__Configuration 6 – Point to Multipoint__



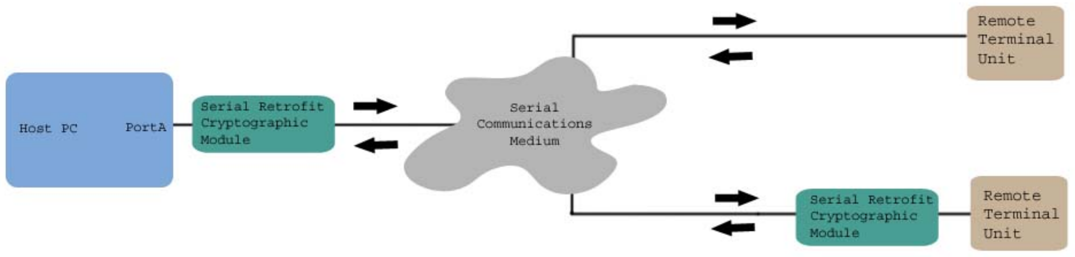
In this point to multipoint configuration, the real-time process control system master connects to two slaves via a communications medium.

__Configuration 7 – Secure Point to Multipoint__


Host PC
PortA
Serial Retrofit Cryptographic Module
Serial Communications Medium
Serial Retrofit Cryptographic Module
Remote Terminal Unit
Serial Retrofit Cryptographic Module
Remote Terminal Unit

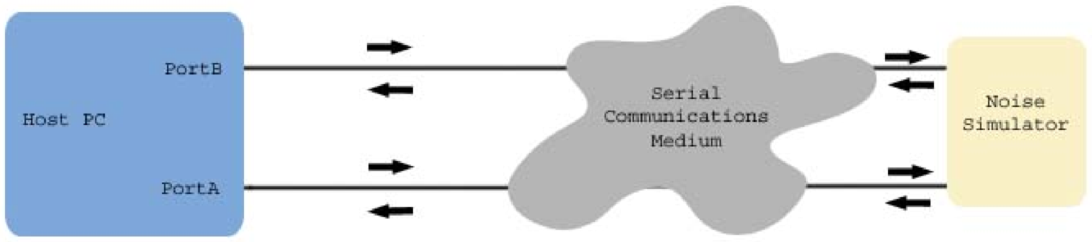
This point to multipoint configuration is the cryptographically secure form of Configuration 6.

<u>Configuration 8 – Partially Secure Point to Multipoint</u>


Host PC
PortA
Serial Retrofit Cryptographic Module
Serial Communications Medium
Remote Terminal Unit
Serial Retrofit Cryptographic Module
Remote Terminal Unit

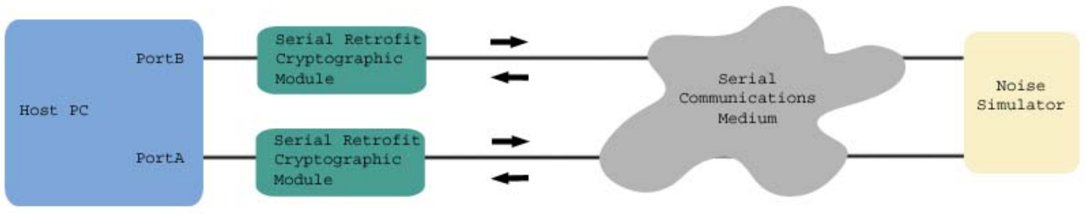
This point to multipoint configuration is similar to Configuration 7. The only difference is that one of the slaves is not cryptographically secured, requiring to the master CM to operate in mixed-mode.

<u>Configuration 9 – Loop with inline Noise Simulator</u>



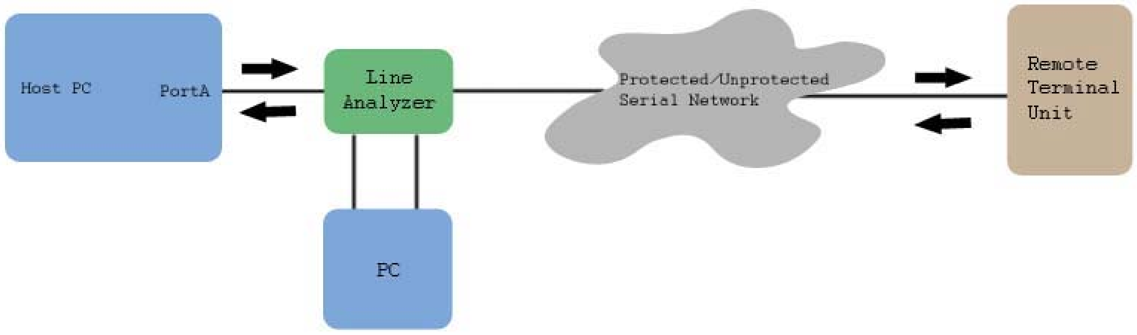
In this configuration, the port A on the master is connected back to port B via a noise simulator.

<u>Configuration 10 – Secure Loop with inline Noise Simulator</u>


Host PC
PortB
PortA
Serial Retrofit Cryptographic Module
Serial Retrofit Cryptographic Module
Serial Communications Medium
Noise Simulator

This is the cryptographically secure version of Configuration 9.

Configuration 11 – Point to Point with Line Analyzer Box



In this configuration, the line analyzer box is connected inline on a point to point connection between the master and slave. Another PC is connected to the line analyzer box to capture the traffic going across the point to point connection. This PC runs two instances of the GTI/NIST Software tool to capture the incoming and outgoing traffic. The message parser function of the GTI/NIST software tool is used to capture and store the messages. For further information, refer to the help files associated with the software.

3.F Test Equipment

This section describes the complete set of test equipment needed to run the tests. This equipment is described here in detail once to avoid repeating the descriptions. Individual test procedures will call on this equipment as needed.

3.F.1 Hardware

- PC
  A x86 architecture computer running Microsoft Windows 95/Windows NT 3.1 or higher with two open serial ports
- Cryptographic Modules
  Multiple interoperable retrofit serial cryptographic modules under test
- Serial Cables
  Serial cables are used to connect the various test equipment
- Null Modem
  The null modem crosses select signals on the serial port cable. It allows the connection of two DTE devices by emulating the physical connections of a DCE device.
- Level Shifting Circuit / Line analyzer box
  The level shifting circuit is used to capture the data over a line. It is also referred to as the line analyzer box in the tests described in Appendix C. The purpose of the level shifting circuit is described below.

  To capture the data on a data acquisition system for analysis, the voltages must correspond to voltages at levels called TTL voltages. However, most cryptographic modules produce voltages that are higher (known as RS 232 level voltages). In order for GTI to see the time-varying voltages that carry the data streams from both the cryptographic modules (which are at the RS 232 communication levels,) it is necessary to convert the TTL signal levels to RS 232 levels.

- Micro Seven LS200 PBX (or equivalent hardware to introduce controlled noise)
  A Private Branch Exchange apparatus that allows the tester to introduce variable levels of noise in the dial-up modem communications.
- Dial-up Modems
  Dial-up modems are used for long distance serial communications. They are specifically used in conjunction with the PBX described above to assess the effect of noise on the CMs.
- Wireless/Radio Modems (optional)
  Wireless/Radio modems are also used for long distance serial communications. They can be used to test the CMs within specific real-time control system network topologies.
- Lease Line Modems (optional)
  Lead lines modems are also used for long distance communications. They can be used to the CMs within specific real-time control system network topologies.
- Serial Breakout Box (optional)
  A testing device that permits the user to monitor the status of the various signals between two communicating devices and to cross and tie interface leads, using jumper wires. The breakout box can be used to quickly construct custom serial connectors. Most breakout boxes draw power from the serial line. However, battery powered breakout boxes may be preferred to ensure that the components on the line receive signals with appropriate voltages.
- Datascope (Optional)
  A data scope may be used to accurately examine any time differences.

3.F.2 Software

- GTI/NIST Software Tool
  A software tool was developed by GTI to conduct the tests in Appendix C. The software may be downloaded from the following website:
  http://gtiservices.org/security/aga12_wkgdoc_homepg.shtml
  Refer to the associated README file on how to use this testing tool.
- Operational Model
  Software model designed to predict the operational impacts of serial retrofit cryptographic modules. This software may be downloaded from the following website:
  http://gtiservices.org/security/aga12_wkgdoc_homepg.shtml
  Refer to the README file on how to use the model.
- Real-time control system master software
  Commercial software used in real-time host applications
- Software to Generate Noise (optional)
  Software to simulate noise over a line in a controlled manner.

- Protocol Simulator

  Commercial protocol simulator to conduct clock synchronization tests. The example described in Appendix A, Section C.2.4.1.5 uses the Triangle Microworks DNP3 Test harness to conduct clock synchronization tests for DNP3.
- "Gold Standard" Implementation

  For interoperability tests of AGA 12, Part 2 compliant CMs, the tester may require the ScadaSafe/Gold standard implementation of the cryptographic protocol. The ScadaSafe implementation can be downloaded from www.scadasafe.sf.net. The ScadaSafe implantation is programmed in Java and therefore can run on many different platforms. Instructions on how to install the ScadaSafe implementation can be found in the documentation associated with it.

4 Models

4.A Operational Model

4.A.1 Overview

The purpose of this model is to predict the operational impacts of adding cryptography (specifically, serial retrofit cryptographic modules) to a real-time control system. This model has several potential advantages:

- It can be used to predict the latency that results from cryptographic modules that require a particular time to encrypt given the cryptographic protocol, baud rate, and message length.
- The model can be combined with a few measurements to allow rapid extrapolation and interpolation from a limited number of measurements to a complete curve.
- Marked deviation between the model and measured values would allow rapid identification of bad measurement techniques, modules that have a problem or modules that have a poor design.
- The model can provide insight into the gross behavior of the module under test, such as the amount of time required to encrypt and decrypt messages, the overhead due to protocol, and buffering time.

It is unrealistic to create a general operational model that accounts for every type of cryptographic module design and real-time control system. Therefore, the following model focuses particularly on the impact of AGA 12 compliant cryptographic modules to secure SCADA messages. However, the model is general enough to accommodate a new model, which can easily be derived from it.

The model relies on specific terminology to describe the behavior of an AGA 12, Part 2 compliant cryptographic system. All terms in this discussion comply with the definitions in the AGA 12 document.

The model is divided into 5 sections. The first section describes two different types of message formats. The second section describes the constraints that apply to this model. The third section provides a list of terms used in the model. The fourth section describes how the latency is modeled given specific criterion. The fifth section describes how the throughput is estimated for a given situation. GTI has developed software that models the latency and throughput as described in sections 4.A.1.4 and 4.A.1.5. Refer to the documentation associated with the model for further information.

4.A.1.1 Message Format

The model relies on particular message formats, as specified by AGA 12. They are listed below in tables 1 and 2.

Optional Framing bytes	SCADA UNIT ADDRESS (‘To’ field)	SCADA UNIT ADDRESS ('From' field)	SCADA Message	CRC, Checksum etc	Optional framing bytes

Table 1: Cleartext Message Format

Header						Payload					Trailer		
Start of message	Ver.& Flags 1 byte	CM Dest. Address 2 bytes (‘To’ field)	CM Source Address - 2 bytes (‘From’ field)	Session Id 1 byte	Seq. # 4 bytes	16 byte block of ciphertext	…	16 byte block of ciphertext	Last block of ciphertext		End of Payload /Start of Trailer	Hash, CRC, HMAC etc.	End of Message
									Encrypted cleartext	Padding			

Table 2: Link layer message format

4.A.1.2 Constraints and Assumptions

The following constraints apply to the cleartext real-time control system message:

1. The real-time control system message consists of two parts: the address and the actual message. The address portion precedes the message.
2. The cleartext data is encrypted using a block cipher.

Assumptions made in this model:

1. Session establishment is assumed to be infrequent. Over a long period of time session establishment will have negligible effect on the added latency due to cryptography and therefore, is not factored into the model.
2. It is assumed that calculations related to integrity verification and/or authentication such as the HMAC, CRC, checksum etc. take negligible time.
3. The process of padding as well as composing the header and trailer are also assumed to take negligible time and are therefore ignored in the equations.
4. The encryption and decryption times are assumed to be constant for any block of cleartext.
5. A CM can only encrypt & decrypt one block of data at a time.
6. The CM has an infinite buffer.
7. The real-time control system network is assumed to be half duplex (in most cases this assumption has no effect on the latency and throughput modeling).
8. The CM does not use any prediction algorithms to predict the real-time control system message.

4.A.1.3 List of terms

Below is a list of terms used in the model and their definitions. Many constants specified by AGA 12 are treated as variables in this model, so as to achieve a comprehensive analysis of performance.

b_{cm}	Bit rate in bits /second on the link between the two CMs.
b_{scada}	Bit rate in bits/second on the link between the CM and real-time process control system
L	Total Latency for a single unidirectional message
L_{Query}	Latency for Master Query
$L_{Resp.}$	Latency for Slave Response
N_a	Number of bytes used for addressing in the cleartext message
N_{bl}	Block length (16 bytes for all current cipher suites for AGA 12-1)
N_{block}	Number of blocks being transmitted from CM to CM (If the last block consists entirely of padding, then this block is not counted here)
N_c	Number of bytes in the cleartext message
N_{endfr}	Number of framing bytes used by CM to designate end of payload (this is the SOT for the AGA 12-1 protocol)
N_h	Number of header bytes (according to current protocol specification this adds up to 14 bytes)
N_{last}	Number of cleartext message bytes that are transmitted in the last block of CM to CM communication.
N_p	Payload for CM (16 bytes for all current cipher suites for AGA 12-1)
N_{pl}	Payload for the last block of CM to CM communication (N_p bytes according to AGA 12 specification)
N_{tr}	Number of trailer bytes (14 bytes for PE mode for AGA 12-1)
Ps	Packet size in bits.
T_{slave}	Slave delay in Response
t_{cm}	Transmission time for 1 packet (character) over the communication channel between the two CMs
t_{scada}	Transmission time for 1 packet (character) over the communication channel between the real-time control system equipment and CM.
T_a	Transmission time for N_a addressing bytes
T_{bl}	Transmission time for N_{bl} block length bytes
T_c	Transmission time for cleartext message
T_{de}	Time for decryption
T_{en}	Time for encryption
T_{endfr}	Transmission time for the N_{endfr} framing bytes used by CM to designate end of message
T_h	Transmission time for N_h header bytes
T_{host}	Host delay between polls (i.e. the time between the receipt of the last byte of the previous response and the output of the first byte of the next poll)
T_{last}	Transmission time for the number of cleartext bytes in the last block of CM to CM communication
T_{out}	Time needed for CM or real-time process control system to conclude a time out
T_{poll}	Set time for each poll/response cycle
T_p	Transmission time for N_p payload bytes
T_{pl}	Transmission time for N_{pl} payload bytes
T_{SysOv}	System overhead time attached to the serial port communication (or any other fixed overhead to the process)
T_{tr}	Transmission time for N_{tr} trailer bytes

4.A.1.4 Latency

For a given set of assumptions, this section describes the time it takes to receive cleartext messages in a real-time process control system with cryptographic protection using retrofit serial cryptographic modules. The model makes it possible to calculate the difference between message delivery times with and without cryptographic protection.

<u>Understanding the latency model</u>

Without cryptography the asynchronous serial communication is straightforward and the latency is just the time it takes to transmit the cleartext data from the sender to the receiver i.e. it is the product of the packet size (Ps) and number of cleartext bytes (N_s) divided by the bit rate (b).

When cryptographic modules are installed, the asynchronous serial communications process becomes complex. The procedure used to calculate latency will vary significantly for different CM designs. Hence, the latency model developed by GTI is designed to calculate the lowest possible latency, given the above mentioned constraints, the message length, encryption/decryption times as well as the bit rate for transmission over the different channels. The latency model is based on a CM design that runs multiple tasks in parallel to improve performance. Figure 1 shows an example of asynchronous serial communication using such a CM design for cryptographic protection.

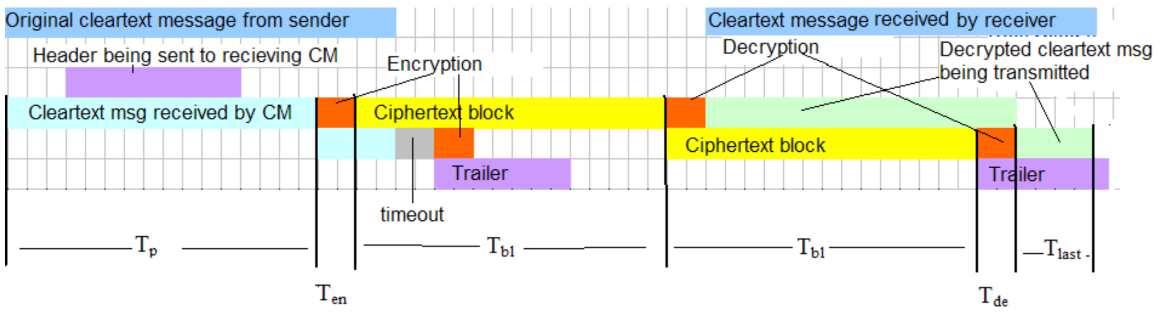

Original cleartext message from sender
Cleartext message received by receiver
Header being sent to recieving CM
Encryption
Decryption
Decrypted cleartext msg being transmitted
Cleartext msg received by CM
Ciphertext block
Ciphertext block
Trailer
Trailer
timeout
T_p
T_{b1}
T_{b1}
T_{last}
T_{en}
T_{de}

Figure 1: Example asynchronous serial communication with cryptography.

In this example a cleartext real-time control system message is being transmitted from the sender to the receiver. The horizontal axis of the figure is in units of time, starting from the transmission of the first byte of data from the real-time control system host, until the reception of the very last byte of data by the RTU. The vertical rows show the transmission of each block of data.

The sender beings sending cleartext to the encrypting CM and as soon as the CM receives the address portion of the cleartext message (N_a) it formulates and transmits the message header. Once the CM has a payload of cleartext data (N_p), it encrypts the data and then transmits the ciphertext while still receiving cleartext from the sender in parallel. This process is repeated until the end of message is indicated by a preset timeout (T_{out}). For messages that are a multiple of the payload in length the timeout may be concluded during the encryption phase. Once the end of message is determined the last block is filled with padding if needed, encrypted and serially transmitted followed by the trailer.

On the other end, the decrypting CM serially receives blocks of ciphertext data. Once a block of ciphertext data has been received by this CM, it decrypts the ciphertext and transmits the cleartext to the receiver while still receiving ciphertext belonging to the next block. If the decrypting CM receives the 'end of payload' framing bytes (N_{endfr}) of the trailer while decrypting a block, it recognizes that it is decrypting the last block and separates the payload from the padding before sending the cleartext to the receiver.

The latency for a particular message is the total time it takes for entire message to cross the network connection. It is clear from the above description that this latency is not just a straight addition of all the times associated with the various processes that take place during the transmission of the message.

4.A.1.5 Throughput

For a given set of assumptions, this section describes how to estimate the throughput for a particular cryptographic system. The throughput varies on account of many factors, some being - message length, system configuration as well as the type of message. The model will estimate the maximum throughput for a single message or a sequence of messages. All of the constraints listed in the above sections apply to this discussion.

In a point to point configuration and poll/response situation, the inverse of the total latency for the poll and response is the throughput of the system. This is because the master will not send a new query to the slave until the response to the previous query is received from the slave. That is,

Maximum Throughput = (Total latency for the poll and the response) $^{(-1)}$
$= (L_{Query} + L_{Resp} + T_{slave} + T_{host})^{(-1)}$ - (3)

The above, is also true of multi-drop configurations where no other slaves are queried while waiting for a response from a particular slave. But, if the slave response size and slave delay in responding, varies based on the slave this equation has to be adjusted to the particular situation. Equation (3) can easily be modified for a sequence of messages by having the total latency represent the total latency for a sequence of polls and responses to obtain the throughput.

Adjustments for equation (3):

1. If $T_{tr} > T_{last} + T_{slave} + T_p + T_{en}$ then the difference needs to be added to the total latency, where N_p (used to calculate T_p) is replaced by N_c for the response if $N_c < N_p$.

In situations where data is only communicated one way, and a response is not expected to a message sent by the master, the above equation may not apply. For AGA 12-1 compliant cryptographic modules, the ciphertext message will always be greater than or equal to the cleartext message, and therefore, continuous sustained throughput is not possible without a inserting a delay between continous unidirectional messages. All such CMs will crash eventually when messages are continuously transmitted via the cryptographic system without any added delay. Hence, such situations are not modeled here.

4.B Economic Model

The operational model can be used for rough estimates of the economic impact of adding cryptographic protection to a real-time process control system. Because this impact involves highly subjective input, this approach is more of an outline of an approach than a highly structured model.

The economic assessment involves both an objective and a subjective component. The objective component is the set of test results contained in this report. For example, the first step in assessing the economic impact of cryptography would be to measure the (objective) decrease in throughput that cryptography will introduce. The system operator must then decide subjectively whether this is acceptable for the particular process control system that is being used. If the throughput decrease is acceptable, the only impact of cryptographic protection will be the initial costs of purchasing and installing the cryptographic system. Quite often, the communication link cost is fixed, so if the throughput decrease is acceptable, there is no economic impact. However, for instances in which the decrease can not be tolerated, it will be necessary to install a second, parallel link, purchase higher speed modems, or take some other remedial steps. The system operator may also conclude (as in the risk assessment methodology recommended in AGA 12, Part 1) that the cost of protection exceeds the benefit of reducing the risk and decide that the protection risk is not economically justified.

5 Recommendations and Conclusions

GTI believes that the test procedures in this report constitute a sound technical basis on which to evaluate cryptographic systems that will be deployed for the protection of real-time process control systems. As such, the community of experts and users of such systems should be informed of the existence of these tests and encouraged to use them. Accordingly and with NIST's permission, GTI will

- ➢ Post this report on the AGA 12 web site, along with the models and automation software
- ➢ Publicize the value of the AGA 12 RFC as a good starting point for a list of requirements to be used by those concerned with process control security.
- ➢ Maintain the web site by updating procedures as indicated by the user community
- ➢ Encourage those interested in evaluating the impact of cryptography on real-time process control systems to use these procedures in evaluating systems and reporting results.

GTI anticipates that there may be specific needs for testing in the future, particularly in the AGA 12 area. This possible future work includes –

- ➢ Establishment of an AGA 12 User's Group
- ➢ Development of an AGA 12 Compliance Test protocol to determine the extent to which commercial products comply with the standard, and
- ➢ Establishment of recognized testing and compliance centers (similar to CMVP) to certify compliance to the AGA 12 standard in accord with the Compliance Test Protocol.

Appendix A References

[1] AGA 12 Cryptographic Protection of SCADA Communications Part 1: Background, Policies and Test Plan.

[2] AGA 12, Part 2 RFC.

Appendix B Acronyms and Definitions

B.1 Acronyms

AES	Advanced Encryption Standard specified in FIPS PUB 197
AGA	American Gas Association
CM	Cryptographic Module
HMAC	Keyed-Hashed Message Authentication Code
NIST	National Institute of Standards and Technology
PCMCIA	Personal Computer Memory Card International Association
RTU	Remote Terminal Unit
SCADA	Supervisory Control And Data Acquisition System

B.2 Definitions

AGA 12 series: A recommended practice published by the American Gas Association, which is comprised of a series of documents. AGA 12, Part 1 includes background information, general security policies, and the cryptographic system test plan. AGA 12, Part 2 includes requirements to retrofit existing asynchronous serial communications. AGA 12, Part 3 and subsequent documents will address other configurations.

Authentication: A process that establishes the origin of information, or validates an entity's identity.

Block: A group of contiguous characters formed for transmission purposes.

Bottleneck: A component of a control system network that limits performance on that network.

Boundary: The point at which increasing the length of a cleartext message by one byte, will cause the ciphertext message length to increase by one block length bytes.

Ciphertext: Data in its encrypted form.

Ciphertext port: The CM communications port connected to a protected communications link. Communications on this port may be in plaintext or ciphertext.

Cleartext: Unencrypted data without format additions or changes, such as framing or padding.

Cold backup: A method of redundancy in which the secondary (i.e., backup) system is only called upon when the primary system fails. The system on cold standby receives scheduled data backups, but less frequently than a warm standby. Cold standby systems are used for non-critical applications or in cases where data is changed infrequently.

Cryptographic key (key): A parameter used in conjunction with a cryptographic algorithm that defines the transformation of plaintext data into ciphertext data, the transformation of ciphertext data into plaintext data, a digital signature computed from data, the verification of a digital signature computed from data, an authentication code computed from data, or an exchange agreement of a shared secret.

Cryptographic Module (CM): The set of hardware, software, and/or firmware that implements approved security functions (including cryptographic algorithms and key generation) and are contained within the cryptographic boundary.

Decryption: The process of changing ciphertext into plaintext using a cryptographic algorithm and key.

Degradation: The deterioration in performance of a real-time control system network.

Encryption: The process of changing plaintext into ciphertext using a cryptographic algorithm and key.

Hot backup: A method of redundancy in which the primary and secondary (i.e., backup) systems run simultaneously. The data is mirrored to the secondary server in real time so that both systems contain identical information.

Integrity: The property that sensitive data has not been modified or deleted in an unauthorized and undetected manner.

Interoperability: The ability of CMs from multiple vendors or CMs from the same vendor but with different versions to facilitate successful secure communication on a network using standardized cryptographic protocols.

Jitter: The measure of the variability of the latency across a network.

Key: See cryptographic key

Keyed-Hash Message Authentication Code (HMAC): A mechanism for message authentication using cryptographic hash functions, in combination with a shared secret key.

Latency: The time it takes for a packet to cross a network connection, from sender to receiver.

Master: A device that initiates communications requests to gather data or perform controls.

Message: An ordered series of characters used to convey information.

Operational reliability test: This test evaluates cryptographic system operation under maximum sustained load.

Overhead: Any data added to the encrypted cleartext bytes so as to direct or control the transfer of encrypted data or for error checking.

Payload: The maximum number of encrypted cleartext bytes that fit into one block of encrypted data.

Plaintext: Unencrypted data with format additions or changes, such as framing or padding.

Plaintext key: An unencrypted cryptographic key. [1]

Plaintext port: The cryptographic module communications port connected to a protected device. All communications on this port are in cleartext.

Plaintext: Unencrypted data with format additions or changes, such as framing or padding.

Port: A physical entry or exit point of a cryptographic module that provides access to the module for physical signals, represented by logical information flows (physically separated ports do not share the same physical pin or wire).

Slave: A device that gathers data or performs control operations in response to requests from the master, and sends response messages in return. A slave device may also generate unsolicited responses.

User: An individual or process acting on behalf of the individual that accesses a cryptographic module in order to obtain cryptographic services.

Standard Deviation: Standard deviation is calculated as the square root of the unbiased sample variance. The unbiased sample variance is calculated as follows –

$$s_{N-1}^2 \equiv \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \bar{x})^2,$$

N is the sample size and x bar is the mean.

Stressed reliability test: This test evaluates cryptographic system operation under peak load.

Supervisory Control and Data Acquisition (SCADA) system: A system operating with coded signals over communication channels so as to provide control of remote equipment (using typically one communication channel per remote station). The supervisory system may be combined with a data acquisition system, by adding the use of coded signals over communication channels to acquire information about the status of the remote equipment for display or for recording functions. SCADA systems are a type of real-time process control systems.

Throughput: The amount of real-time control system messages communicated successfully through the given network over a period of time. Throughput is measured in messages per hour (MPH), message sequences per hour (MSPH) or payload bits per second.

Warm backup: A method of redundancy in which the secondary (i.e., backup) system runs in the background of the primary system. Data is mirrored to the secondary server at regular intervals, which means that there are times when both servers do not contain the exact same data.

Appendix C Test Procedures

C.1 Functionality Tests

This section describes the functionality tests in detail.

C.1.1 Functionality Test

C.1.1.1 Introduction

The purpose and background of functionality tests of a real-time process control system with cryptographic protection are described.

C.1.1.1.1 Purpose

The purpose of these tests is to determine impact of cryptography on the functionality of a real-time process control system.

C.1.1.1.2 Background

Functionality testing examines the extent to which your real-time control system hardware and software meet expected functional requirements. The tester should develop a functionality checklist or a functional specification that identifies all the key real-time control system as well as CM application features that need verification. Once the CMs are installed, the tester should verify the functionality of all the features listed in the checklist. Results should be reported as not supported, not applicable, pass, or fail with optional qualifying remarks. Functionality tests should be conducted anytime there is a major change in the real-time control system network and/or a firmware upgrade on the CMs.

C.1.2 Jitter Tests

C.1.2.1 Introduction

The purpose, background, procedure and evaluation as well as interpretation of results of the jitter tests are described.

C.1.2.1.1 Purpose

The purpose of this test is to determine impact of cryptography on the jitter of a real-time process control system.

C.1.2.1.2 Background

The basic idea behind the jitter tests is to send a large number of messages via the cryptographic system and measure the latency variations. AGA 12 defines jitter as "…the standard deviation of at least 100 samples of a particular latency test." The jitter test should be repeated with and without the cryptographic modules, and standard deviation compared.

The jitter tests are highly dependent on the accuracy and resolution of the measuring equipment. The user should be aware of these factors in order to distinguish between variation caused by the measuring equipment and the jitter caused by cryptography.

C.1.2.1.3 Test Equipment

The following test equipment is required for this test:

- PC
- GTI/NIST test software OR any other tools such that the latency can be measured as messages of various lengths are passed through the cryptographic system.
- Two identical CMs to be tested
- Serial Cables
- Null Modem (if necessary)

C.1.2.1.4 Procedure

C.1.2.1.4.1 General Test Procedure

The general procedure for testing the effect jitter is as follows:

- Connect the two CMs and any other needed equipment such that the latency can be measured accurately when a chosen cleartext message is transmitted via the CMs.
- For various message lengths, send a known message through the cryptographic system at least a 100 times per message length and measure the latency each time.

- For each message length, calculate the average and standard deviation of the measured latencies.
- Record the results in a table or as a chart.

C.1.2.1.4.2 Detailed Test Procedure

The following section describes the test procedure in detail. This procedure utilizes the software tool developed by GTI.

Setup the test equipment as shown in Configuration 1.

Run the GTI/NIST software tool to conduct this test. Configure the settings for the program appropriately. Refer to the software documentation for further information.

Select the range of message lengths to be tested as well as the number of times to test each message length.

Run the test. When the test is finished the latency results will be available in the log files in table form. These results can easily be imported to a spreadsheet program such as MS Excel®. The averages and standard deviations can be graphed for further examination.

Repeat the test with the same software configurations but with the test equipment setup as shown in Configuration 3.

C.1.2.1.5 Test Example

GTI conducted the jitter tests on the AGA 12 prototype retrofit cryptographic modules. The prototype CMs ran the ScadaSafe implementation of the AGA 12, Part 2 protocol on the Arcom Viper® development boards. The software developed by GTI was used to help perform the test. The test results are provided below.

C.1.2.1.5.1 Results
Test Form

Test Name	Test subset (if applicable)
Jitter Tests	Bench

Date: 02/16/05
Company/Organization: Gas Technology Institute
Test Performed By: Aakash Shah Position: Asst. Electronics Engineer

Cryptographic Module information

Make: Prototype Model: Type:
Firmware Version: (ScadaSafe version) 0.6.7
ScadaSafe Message Timeout: 20 character times

ScadaSafe Prebuffer: 14 character times

Cryptographic Protocol information

AGA 12, Part 2 - SSPP protocol
- Encryption algorithm used: AES
    - Key length: 128
- Authentication algorithm: 4 byte HMAC
- Mode of operation: PE Mode

Test Configuration

Note the following information:

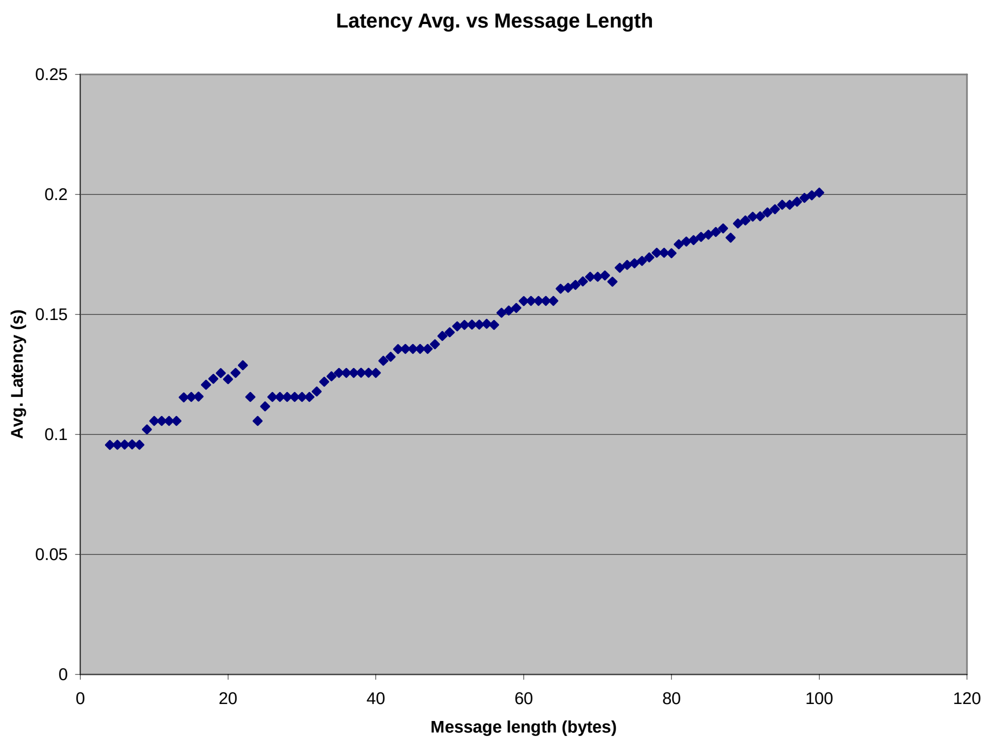
- Make, model and/or type of the different kinds of communications equipment
  RS-485, B&B 485OT9L 232-485 converters
- Use of custom connectors and/or Null modems
  Null modem between host PC and CMs
- Host controller information
    - Make, Model, Processor and Operating System
      Dell Demension, 2.4 GHz Intel Pentium, Win2k
    - Host software used
      GTI software tool
- Communication parameters on various links
    - Baud rate : 9600
    - Data bits : 8 bits
    - Stop bits : 2
    - Parity bits : n
    - Full/half duplex: half duplex
    - Flow control: none
    - Are communications parameters negotiated or ever changed? - No
- Note all relevant parameters used for the GTI software tool.
    - Timeout: 5 s
    - Time between consecutive messages: 4 ms
    - Test message pattern used to measure jitter: Random
    - Number of times each test message repeated: 200
    - Start Message Length: 4
    - End Message Length: 100

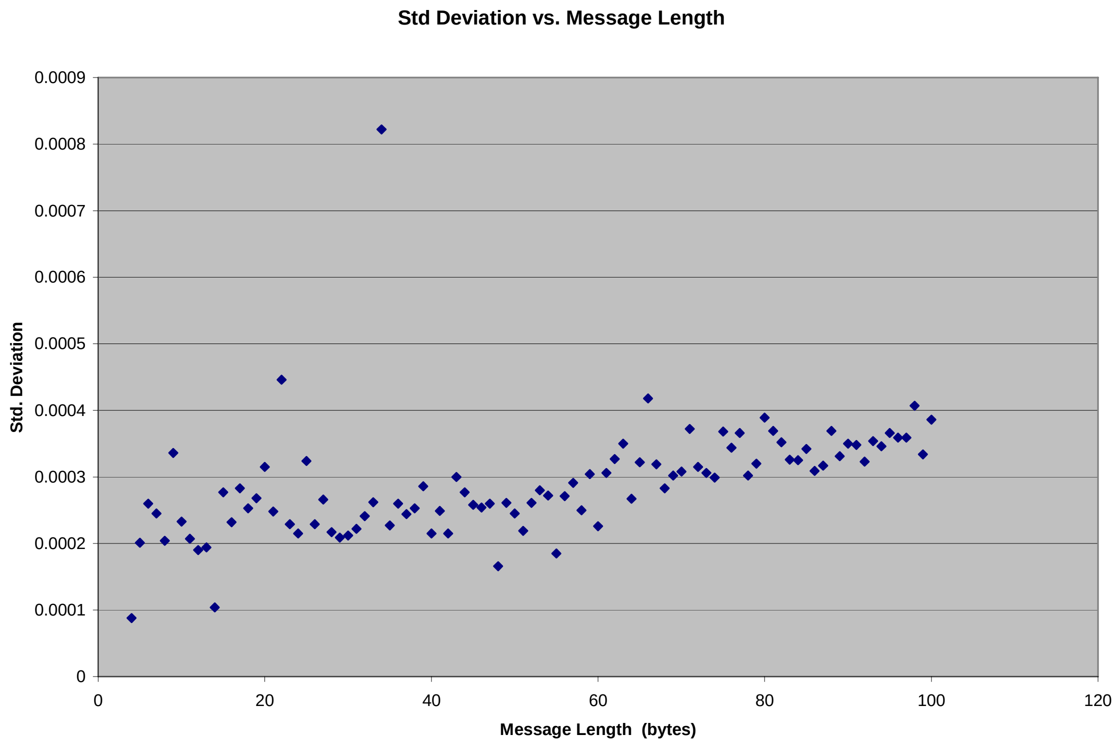
Results

Latency Averages and Standard Deviations (in table or chart form): (see next page)

Notes

The jitter tests were conducted on the AGA 12, Part 2 compliant ScadaSafe prototype. This prototype is not optimized for performance. Therefore, the results of these tests may not be characteristic of commercial serial retrofit CMs.





C.1.2.1.6 Interpretation of Results

As discussed earlier, it is critical that the tester have knowledge regarding the accuracy and resolution of the measuring equipment (refer to the documentation associated with the software for information related to the timing measurements with GTI software). Apart from the measuring equipment, the PC used to generate and receive messages may also affect the jitter results. But, all such effects would also be observed in the jitter results for baseline case in Configuration 1 where the system is tested without the CMs. These effects can be evaluated by comparing the results from the baseline case with the expected/theoretical latency. The theoretical latency can easily be calculated knowing the message size, packet size and bit rate (the operational model may also be used to calculate this value - see the README for the operational model software). Hence, when analyzing the results for Configuration 3, the user must take into account the results from Configuration 1.

Depending on the real-time process control system, jitter above a certain level may have adverse effects on performance. The tester should determine the acceptable level of jitter for his/her real-time process control system.

C.1.2.1.7 Definitions

Jitter: The measure of the variability of the latency across a network.

Latency: The time it takes for a packet to cross a network connection, from sender to receiver.

C.1.2.1.8 Test Results Sheet

Test Form

Test Name	Test subset (if applicable)
Jitter Tests	Bench

Date:
Company/Organization:
Test Performed By: Position:

Cryptographic Module information

Make: Model: Type:
Firmware Version:
Other information:

Cryptographic Protocol information

Note all relevant information regarding the CM's cryptographic protocol including:

- Encryption algorithm used:
    - Key length:
- Authentication algorithm
- Integrity method
- Mode of operation

Test Configuration

Note the following information:

- Make, model and/or type of the different kinds of communications equipment
- Use of custom connectors and/or Null modems
- Host controller information
    - o Make, Model, Processor and Operating System
    - o Real-time control system host software used
- Communication parameters on various links
    - o Baud rate
    - o Data bits
    - o Stop bits
    - o Parity bits
    - o Full/half duplex
    - o Flow control
    - o Are communications parameters negotiated or ever changed?
- Note all relevant parameters used for the GTI software tool.
    - o Timeout:

  - Time between consecutive messages:
  - Test message pattern used to measure jitter:
  - Number of times each test message repeated:
  - Start Message Length:
  - End Message Length:

Results

Latency Averages and Standard Deviations (in table or chart form):

Notes

C.1.3 Interoperability Tests

C.1.3.1 Introduction

The purpose, background and procedure for testing the interoperability of a cryptographic module are described.

C.1.3.1.1 Purpose

The purpose of this test is to verify the interoperability of cryptographic modules manufactured by different vendors.

C.1.3.1.2 Background

Some cryptographic module designs adhere to standards that require interoperability. AGA 12 is one such standard. The following excerpt is from AGA 12, Part 1:

> AGA 12 enforces limited cryptographic interoperability by requiring all compliant components to exchange encrypted messages using at least one common cryptographic algorithm, and to exchange session keys using at least one common key exchange method. While operating within one session, AGA 12, Part 1 requires at least one mode in which the shared session key shall, as a minimum, be used for encryption and decryption of SCADA messages between cryptographic modules at the master station and the cryptographic modules at the remote locations.

In such cases, it is important for a real-time control system operator to verify that the cryptographic module indeed conforms to the standard and that it is interoperable. Interoperability tests can be conducted between two different vendor CMs or between different CM versions from the same vendor.

In the case of AGA 12, a "gold standard" implementation[1] of the AGA 12, Part 2 cryptographic protocol is freely available on the web (http://scadasafe.sf.net). This implementation being in Java can be run on virtually any computer system with two serial ports. Tests can then be conducted to prove interoperability between the cryptographic module under test and the "gold standard." The detailed test procedure described below will focus on testing AGA 12 interoperability using the "gold standard" but can easily be generalized for any other standard.

C.1.3.1.3 Test equipment

- CMs to test

[1] Also known as the ScadaSafe implementation. This implementation was developed by Dr. Andrew Wright as part of Cisco Systems Critical Systems Assurance Group.

- A PC running the “gold standard”/ ScadaSafe implementation of the AGA 12, Part 2 protocol
- Another PC
- GTI/NIST software tool
- Serial cables
- Null modems

C.1.3.1.4 Procedure

C.1.3.1.4.1 General Procedure

Generate a checklist of interoperability criterion to be tested based on the standard the CMs adhere to.

Setup both CMs to be tested with compatible settings such that they could communicate with each other.

Configure the equipment as shown in Configuration 3.

Verify basic interoperability by sending messages back and forth.

Verify each item listed in the checklist.

C.1.3.1.4.2 Detailed Procedure

Download and install the GTI “gold standard” / ScadaSafe implementation of the AGA 12, Part 2 protocol on a computer system.

Configure the equipment as shown in Configuration 3.

Note: It may be beneficial to install a line analyzer on the ciphertext line (see Configuration 11).

Configure both the GTI “gold standard” and the CM being tested such that they have compatible settings:

- Cipher Suite 1
- Same pre-shared key
- Same serial port settings
- Same SOM, EOP and EOM bytes
- Same Mac Key
- Update the CM address tables in both CMs so that they are allowed to talk to each other
- Setup the “gold standard” CM to be the master and the CM being tested to be the remote CM.

Using the ‘Send Test Pattern’ feature, continuously send a series of messages from one port to another. Leave enough time between messages and increase the host timeout setting (e.g. 1 s between each message and a timeout of 5 s). Verify that messages are received on the other end.
Note: The first few messages may be dropped due to session establishment.

Assuming the CMs are configured correctly, if no messages are received on the receiving port of the host PC, the interoperability test has failed.

If messages are being communicated correctly, reboot the CM being tested. Verify that communication is restored once the CM reboots and re-establishes a session.

Set the session timeout to 2 min on the master CM. Verify that a new session is successfully established and that communication is restored.

Send a broadcast message and verify communication.

Configure the CM being tested to be the master CM, and the “gold standard” to be the remote CM. Change the test path on the GTI software tool and run the above tests again.

C.1.3.1.6 Interpretation of Results

A successful interoperability test between two CMs only confirms interoperability between those two CM models and firmware versions. Any updates to the CM firmware require the interoperability tests to be conducted again. Interoperability should not be assumed just because two different CMs are interoperable to a common implementation such as the “gold standard” described above.

Similarly, in the above procedure, a CM’s failure to interoperate with the “gold standard” implementation of the AGA 12, Part 2 protocol does not necessarily establish non-compliance with the AGA 12, Part 2 specifications. The failure may be the result of ambiguity in the AGA 12 standard.

C.1.3.1.7 Definitions

Interoperability: The ability of CMs from multiple vendors or CMs from the same vendor but with different versions to facilitate successful secure communication on a network using standardized cryptographic protocols.

C.1.3.1.8 Test Results Sheet

<u>Test Form</u>

Test Name Interoperability Test	Test subset (if applicable)

Date:
Company/Organization:
Test Conducted By:

Cryptographic Module Information

CM 1 information:

Make:	Model:	Type:

Firmware Version:
Other:

CM 2 information:

Make:	Model:	Type:

Firmware Version:
Other:

Cryptographic Protocol information

Note all relevant information regarding the CM's cryptographic protocol including:

- Encryption algorithm used:
  - Key length:
- Authentication algorithm
- Integrity method
- Mode of operation

Note all CM settings configured to allow interoperability:

Test Configuration

Note the following information:

- Use of custom connectors and/or Null modems
- Host controller information
  - Make, Model, Processor and Operating System

  - Real-time control system host software used
- Communication parameters on various links
  - Baud rate
  - Data bits
  - Stop bits
  - Parity bits
  - Full/half duplex
  - Flow control
  - Are communications parameters negotiated or ever changed?

Results

Interoperability tests successful (y/n)?

Session negotiation successful (y/n)
Session re-establishment successful (y/n)
Session timeout successful (y/n)

If not, describe the any problems encountered

C.1.4 Backup / Failover System Test

C.1.4.1 Introduction

The purpose, background and procedure for testing the functionality of the backup/failover system of a network protected by retrofit cryptographic modules are described.

C.1.4.1.1 Purpose

The purpose of this test is to ensure that the addition of cryptography retains the backup/failover system functionality.

C.1.4.1.2 Background

Many real-time process control networks have a backup/failover system that can be used if a component on the primary communications channel fails. The functionality of this system must be preserved when cryptographic protection is added. Three common types of backup systems exist: hot, warm and cold backups (see C.1.4.1.5).

The backup/failover topologies are specifically addressed in the functionality tests as they pose a complex problem to the design of retrofit serial cryptographic modules. It is critical that the tester verify the functionality of the CM in his/her backup/failover network.

This test will not describe the topology of the backup/failover network to be tested as this will vary based on the type of backup system. Nor will it describe how the CMs are installed in such a network as this will also vary based on the CM. This test is meant to be conducted in the field where CMs are already installed on the primary as well as back-up communications channels. The following test procedure will provide a series of simple steps to ensure that the backup system is completely functional once cryptographic protection is introduced.

C.1.4.1.3 Test equipment

- CMs to be installed
- No extra equipment is needed as all necessary equipment is already installed.

C.1.4.1.4 Procedure

Perform the following steps before and after the CMs are installed across the real-time process control network.

Disconnect or disable a key component on the primary channel such that the backup system is engaged e.g. pull the power plug on the primary server.

Force a shift to the backup system.

Ensure that communication is restored. If the backup system does not automatically poll all the slaves, poll each slave and ensure communication. Perform any other operations related to the backup system and ensure functionality.

Restore functionality of the primary system and force a shift back to the primary communications channel.

Ensure proper communication by polling each slave, if the host system does not automatically do so.

C.1.4.1.5 Definitions

Hot backup: A method of redundancy in which the primary and secondary (i.e., backup) systems run simultaneously. The data is mirrored to the secondary server in real time so that both systems contain identical information.

Warm backup: A method of redundancy in which the secondary (i.e., backup) system runs in the background of the primary system. Data is mirrored to the secondary server at regular intervals, which means that there are times when both servers do not contain the exact same data.

Cold backup: A method of redundancy in which the secondary (i.e., backup) system is only called upon when the primary system fails. The system on cold standby receives scheduled data backups, but less frequently than a warm standby. Cold standby systems are used for non-critical applications or in cases where data is changed infrequently.

C.1.4.1.6 Test Results Sheet

<u>Test Form</u>

<u>Test Name</u> Backup system Test	<u>Test subset (if applicable)</u>

Date:
Company/Organization:
Test Performed By: Position:

Cryptographic Module information

Make: Model: Type:
Firmware Version:
Other information:

Cryptographic Protocol information

Note all relevant information regarding the CM's cryptographic protocol including:

- Encryption algorithm used:
  - Key length:
- Authentication algorithm
- Integrity method
- Mode of operation

Test Configuration

Draw a detailed communications system diagram depicting the serial real-time process control network topology along with any LAN connections.

Note the following information:

- Approximate geographic distances between components.
- Communications protocols used on various links
- Make, model and/or type of the different kinds of communications equipment
- Use of custom connectors and/or Null modems
- Host controller information
  - o Make, Model, Processor and Operating System
  - o Real-time control system host software used
- Slave device information (if applicable)
  - o Make, model and/or type of all secured slave devices
- Communication parameters on various links
  - o Baud rate
  - o Data bits
  - o Stop bits

    - Parity bits
    - Full/half duplex
    - Flow control
    - Are communications parameters negotiated or ever changed?
- Any specific operating modes of the real-time control system
- Polling scheme (master poll, report-by-exception, etc.), timeouts, and polling rates (e.g. poll every second, as fast as possible, etc.)
- Types of polls and responses and average lengths. If the set of polls is recurring, record the poll/response lengths. If possible note the slave processing time for each poll.

Results

Did the shift to the backup system work (y/n)?

If not, describe the any problems encountered

Did the shift back to the primary system work (y/n)?

If not, describe the any problems encountered

C.2 Performance Tests

This section describes the performance tests in detail.

C.2.1 Block Length Probing

C.2.1.1 Introduction

The purpose, background, procedure and evaluation as well as interpretation of results for block length probing are described.

C.2.1.1.1 Purpose

The purpose of block length probing is to determine the payload per block of ciphertext data and assess the associated overhead in the encryption process.

C.2.1.1.2 Background

Some information on block encryption, overhead and payload is described.

C.2.1.1.2.1 Block Encryption

A cryptographic module, utilizing block encryption, will divide an incoming cleartext message into blocks. For example, if the cryptographic system can fit a payload (see C.2.1.1.2.2 or C.2.1.1.8) of 12 cleartext bytes into a single-block, then a 13 byte message will require two blocks. Thus a 12 byte cleartext message can be encrypted and transmitted in approximately half the time required to encrypt and transmit a cleartext message of 13 bytes. The point at which increasing the length of a cleartext message by one byte, causes the ciphertext message length to increase by one block length bytes is called a block boundary. In above case, the block boundary is between the 12 and 13 byte cleartext messages.

Typically, four cases characterize the encrypted blocks in a cryptographic system.

- A single-block message
- The first block of a multi-block message
- The last block of a multi-block message
- The middle block(s) of a message spanning three or more blocks

To accurately perform block length probing on a cryptographic module, as a minimum, block boundaries need to be identified for one, two and three blocks in order to test all of the above cases.

C.2.1.1.2.2 Payload

For the purposes of this test, payload is defined as the maximum number of incoming cleartext bytes in the cryptographic module that fit into one block of the outgoing encrypted data. The payload should be constant throughout all blocks but may vary in select cases[2]. The payload for a single block message as well as the first, last and middle blocks of a multi-block message may be different. In these cases, all payload variations should be documented. For messages spanning from one to a few blocks, performance of the cryptographic system is strongly dependent on how the payload fits within the block structure.

C.2.1.1.2.3 Overhead

For the purposes of this test, overhead is described as any data added to the encrypted cleartext bytes so as to direct or control the transfer of encrypted data or for error checking.

The overhead within a block may vary depending on its location within the message. Overhead can be classified two categories:

- Fixed Overhead
- Variable Overhead

C.2.1.1.2.1.1 Fixed Overhead

Fixed overhead is the overhead that remains constant for a given message length. It can be split into two categories:

- Overhead independent of cleartext message length
- Overhead dependent on cleartext message length

C.2.1.1.2.1.1.1 Overhead Independent of cleartext length

Overhead independent of cleartext message length is the overhead that is constant in every message regardless of the message length.

- Some cryptographic modules may add message header and trailer bytes to the encrypted cleartext. The header may include CM addressing bytes, message counters, session Ids, etc., while the trailer portion of the message may contain a pad count for the last block and a number of bytes used to validate the integrity of the message.

- The encrypted message may also contain framing bytes to delineate the beginning and end of the encrypted message.

[2] The payload may be different for the first and the last block. For example, in the test shown in C.2.1.1.4 the payload is 13 bytes for the first block and 14 bytes then onwards. The payload for any middle block should be the same. Therefore when referring to the payload of a CM this is the value referred to.

The sum of the header, trailer as well as the framing bytes described above represent the message overhead independent of the cleartext length. Because such overhead is constant in a cryptographic module, it is easy to measure.

C.2.1.1.2.1.1.2 Overhead Dependent on cleartext length

Overhead dependent on cleartext length varies based on message length.

- Some of the bytes in a block may be reserved by a cryptographic protocol for use by the protocol. Examples of this are a pad count byte and CRC bytes.
- The framing mentioned above may also further delineate ciphertext blocks in the message. This will increase the overhead per block.
- When a cleartext message is broken into blocks for the purpose of encryption, the last block may need padding bytes to finish the block.

Except for the padding bytes in the last block, overhead dependent on the cleartext length remains constant per block.

C.2.1.1.2.1.2 Variable Overhead

Variable overhead is unpredictable for a given cleartext message.

- The use of framing leads to variable overhead. Framing requires that at least one byte value, the escape value, receive special treatment by the transport layer of the cryptographic protocol if the byte value occurs within the message. If the transport layer detects the selected byte value within the message the byte is transmitted twice. Thus, the receiving transport layer can then differentiate between byte sequences used for framing and occurrences of escape bytes within a message.
- Other characters utilized by the underlying real-time control system protocol may also receive similar special treatment by the transport layer of the cryptographic protocol (possibly in the mixed-mode scenario) and would add to the variable overhead.

The variable overhead is generally negligible if the ciphertext generated for a given cleartext is random for every case. Nevertheless, it should still be examined to assess performance of a CM. Section C.2.1.1.8 gives a formula to calculate the probability of an escape character appearing in the ciphertext.

C.2.1.1.3. Test Equipment and tools

This test requires the following equipment:

- One CM to test
- PC
- GTI/NIST software tool or any other similar tools.
- Serial cables
- Null Modem (if necessary)

C.2.1.1.4 Procedure

The basic idea behind this test procedure is to monotonically send plain text messages of increasing length and examine the cipher text to evaluate the payload as well as overhead. Below is the general procedure for this test, followed by a detailed procedure that utilizes the GTI software.

C.2.1.1.4.1 General Procedure:

Send cleartext messages of increasing length, starting from the smallest possible message. To get a statistically accurate data, repeat each message length a number of times. The cleartext message can be constructed from any test bytes, but it may need to formatted to account for the real-time process control system protocol the CM is setup for.
Examine the ciphertext to determine the block boundaries as well as identify variable overhead.
Evaluate the payload for different blocks.
Perform data analysis and generate two charts:
Cleartext message length vs Ciphertext message length
Variable overhead vs Cleartext message length

C.2.1.1.4.2 Detailed Procedure

The following procedure utilizes a test program created by GTI to run the block length probing test. Refer the documentation associated with the tool for further information. Section C.2.1.1.4 contains a detailed example of this test and can be useful in understanding the test procedure.

Set up the real-time process control system as shown in Configuration 2.

Run the block length probing test program (blocklengthprobe.exe). Select the appropriate configurations in the *Ports*, *Baud Rate*, *Log File* as well as *Test Values* tabs.

- Test 1
  In the *Test Values* tab set the *Start Message Length* to 1 and pick a reasonably high *End Message Length* so that at least three block boundaries will be recorded in the log file when the test is run. To find a block boundary in the logged data, observe the ciphertext message lengths for a significant jump as the cleartext message lengths increase. If the *End Message Length* chosen was not long enough to record at least two block boundaries, increase the *End Message Length* and run the test again.

  More than one block boundary indicates that the message is a multi-block message with a first block, last block and middle block(s). Record the lengths of all three types of blocks by examining the first two block boundaries (see example in C.2.1.1.5).

The length of any middle block (given that there is no variable overhead) is the block length.

Record the payload for the first, last and middle blocks. The payload for a block is just the increase in cleartext message length since the previous block boundary up till the next one (see example in C.2.1.1.5).

It is possible for some of the numbers recorded in this test to fluctuate slightly due to variable overhead. Therefore, the user should run this test enough times to be certain about these numbers.

In most cases, it is easy to spot the instances where the ciphertext message length is longer due to variable overhead just by looking at the statistical mode of ciphertext message lengths in a given interval where the number of blocks in a message is constant. Another way to determine this overhead is to view the ciphertext generated, identify the framing bytes and look for the framing escape byte appearing in the ciphertext.

- Test 2
  Run some tests with various start and end message lengths. Record the ciphertext output.

  Examine the collected ciphertext for any obvious common patterns. This can help the user gain insight into the internal operations of the CM.

  Identify framing bytes, if used. This can be done easily by looking to see if the message starts or ends with the same byte sequence. Similarly, it can be checked to see if each block is framed individually.

  Determine the payload of the third block in a multi-block message that is longer than three blocks. This is the payload for the CM. Check the payloads for various arbitrary blocks to ensure that the payload remains constant throughout all middle blocks[3].

- Test 3
  Run a comprehensive test with a very high *End Message length* and *Number of messages with the same length.* Select the options on the log files sensibly so as to ensure that the files do not get too large.

  Get the Standard deviation mean value from the log file.

  To better examine the data, import the excel log file into Microsoft Excel and graph the cleartext message length vs. the ciphertext message length. Also, create a

[3] If this is not the case then, refer to the CM documentation for payload information or determine a pattern in the payloads.

histogram of the measured deviations from the expected ciphertext message length as the cleartext length increases.

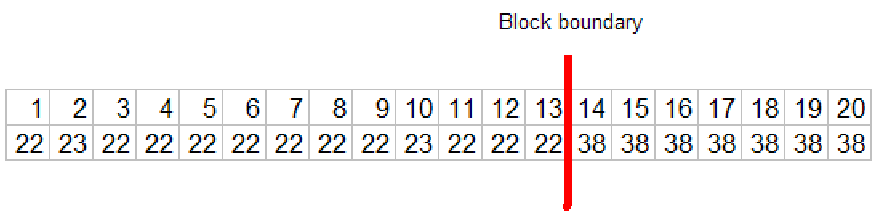
- Other calculations

  Section C.2.1.1.8 provides some formulas that may be helpful to the user. Refer to the first formula and calculate the probability of a framing escape byte appearing in a block length of ciphertext data.

C.2.1.1.5 Test Example
The block length probing test was conducted on an unidentified CM. The test is documented below.

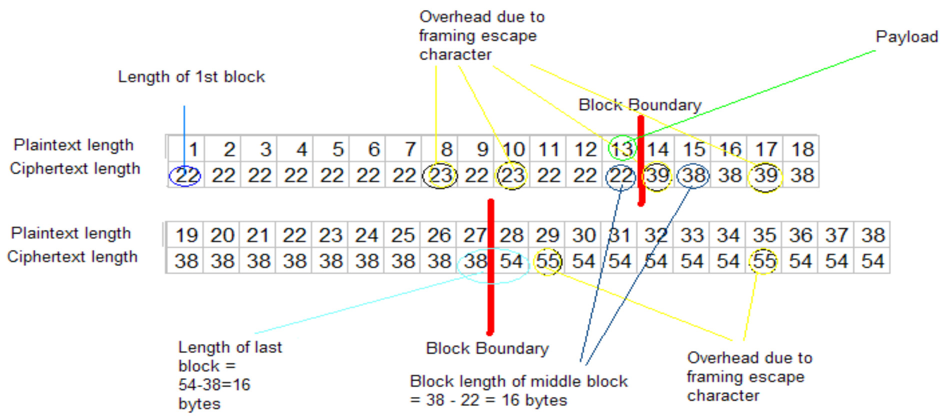
- Test 1

  The *End message length* for this test was initially selected to be 20 bytes. But since the data did not contain two block boundaries (see figure 1), we ran the test again after increasing the *End message length* to 38 bytes (see figure 2).



1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
22	23	22	22	22	22	22	22	22	23	22	22	22	38	38	38	38	38	38	38

Figure 1


Overhead due to framing escape character
Payload
Length of 1st block
Block Boundary
Block Boundary
Length of last block = 54-38=16 bytes
Block length of middle block = 38 - 22 = 16 bytes
Overhead due to framing escape character

Plaintext length	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Ciphertext length	22	22	22	22	22	22	22	23	22	23	22	22	22	39	38	38	39	38

Plaintext length	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38
Ciphertext length	38	38	38	38	38	38	38	38	38	54	55	54	54	54	54	54	55	54	54	54

Figure 2

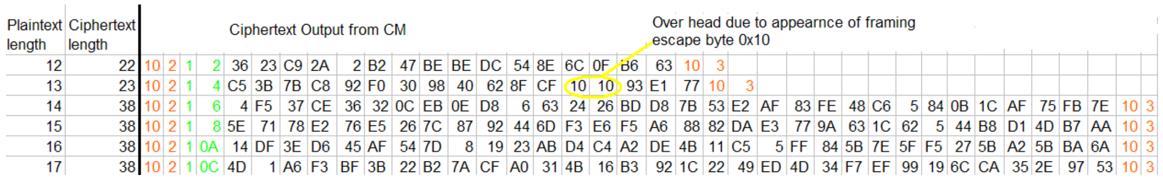
In the example data shown in figure 2, the length of the first block is just 22 bytes. The length of the last block is the difference between the ciphertext length of a two block message and a single block message, which is 38-22 = 16 bytes (this is because a two block message only has a first and last block). Then the length of a middle block is just the ciphertext length of a three block message minus that of a two block message, i.e. 54-38 = 16 bytes.

The first block boundary occurs between 13 and 14 bytes. Since a maximum of 13 bytes of data fit into the first block, the payload for that block is just 13 bytes. The payload for the second (last) block is 27 – 13 = 14 bytes.

The data in figure two also contains some of the fluctuations mentioned above. In all the messages spanning one block in figure 1 and 2, the ciphertext message length is mostly 22 bytes with a few exceptions where it is 23 bytes. It is safe to assume (for shorter messages) that those few cases are due to variable overhead caused by the framing escape character appearing in the ciphertext[4]. This can be verified by looking at the ciphertext output (see Table 1). Similarly, looking at the mode for messages spanning two and three blocks, the expected ciphertext message length is 38 and 54 bytes respectively.

- Test 2

  Here are some of the recorded ciphertext outputs for test 2.



Plaintext length	Ciphertext length	Ciphertext Output from CM																		Over head due to appearnce of framing escape byte 0x10																			
12	22	10	2	1	2	36	23	C9	2A	2	B2	47	BE	BE	DC	54	8E	6C	0F	B6	63	10	3																
13	23	10	2	1	4	C5	3B	7B	C8	92	F0	30	98	40	62	8F	CF	10	10	93	E1	77	10	3															
14	38	10	2	1	6	4	F5	37	CE	36	32	0C	EB	0E	D8	6	63	24	26	BD	D8	7B	53	E2	AF	83	FE	48	C6	5	84	0B	1C	AF	75	FB	7E	10	3
15	38	10	2	1	8	5E	71	78	E2	76	E5	26	7C	87	92	44	6D	F3	E6	F5	A6	88	82	DA	E3	77	9A	63	1C	62	5	44	B8	D1	4D	B7	AA	10	3
16	38	10	2	1	0A	14	DF	3E	D6	45	AF	54	7D	8	19	23	AB	D4	C4	A2	DE	4B	11	C5	5	FF	84	5B	7E	5F	F5	27	5B	A2	5B	BA	6A	10	3
17	38	10	2	1	0C	4D	1	A6	F3	BF	3B	22	B2	7A	CF	A0	31	4B	16	B3	92	1C	22	49	ED	4D	34	F7	EF	99	19	6C	CA	35	2E	97	53	10	3

Table 1

  In the above table, it is clear that the CM uses framing (highlighted in red). The messages always start with the bytes 0x10 and 0x02 and end with 0x10 and 0x03.

  Also, by looking at the third and fourth bytes in every message it can be observed that they do not change as randomly as most of the other bytes do (highlighted in green). These bytes are probably part of the fixed overhead in the first block.

  The payload was determined to be 14 bytes.

- Test 3

  The test was performed up to a cleartext message length of 255 bytes and each message was repeated a 100 times.

  The Standard Deviation mean was recorded as .749853.

[4] This is not a good assumption for large messages but is reasonable for messages spanning just a few blocks. See chart 2.

The charts as well as completed test form are attached below.

- Calculation results

  The probability of one or more framing escape bytes appearing in a block of ciphertext is .064364861 for this cryptographic module.

C.2.1.1.4.3 Results

Test Name	Test Type
Block Length Probing	(Bench / ~~Testbed~~ / ~~Field~~)

Date: 02/09/04
Company: Gas Technology Institute
Test Performed by: Aakash Shah
Position: Assistant Electronics Engineer

Cryptographic Module information

Make: --	Model: ---	Type: ---

Firmware Version: ---
NOTE: Module NOT AGA 12, Part 2 compliant

Encryption algorithm information

Algorithm used: AES
Key length: 128
Other:

Test Configuration

Note the following information:
- Specific communications protocols used on various links (if any)
  None
- Use of custom connectors and/or Null modems
  Null modems on connection between CM and PC
- Host controller OS and real-time process control host software
  Win2k, GTI test software
- Communications parameters: baud rates, start/stop/parity bits, full/half duplex, flow control, if negotiated, ever changed
  9600n2, half dulplex, no flow control
- Any specific operating modes

None

- Note all relevant parameters used for the GTI software tool for Test 3.
    - Start Message Length:
    - End Message Length:
    - Number of times each message repeated:
    - Timeout:
    - Time between read and write states:

Results

Payload for 1st block: 13 bytes
Payload for last block: 14 bytes
Payload for any middle block: 14 bytes
Block Length: 16 bytes
Length of first block: 22 bytes
Length of last block: 16 bytes

Test 2:

End Message Length for Test 2: 255 bytes
Number of times message is repeated: 100
Mean length of ciphertext messages for the longest cleartext message: 311 bytes
Standard Deviation Mean: .749853

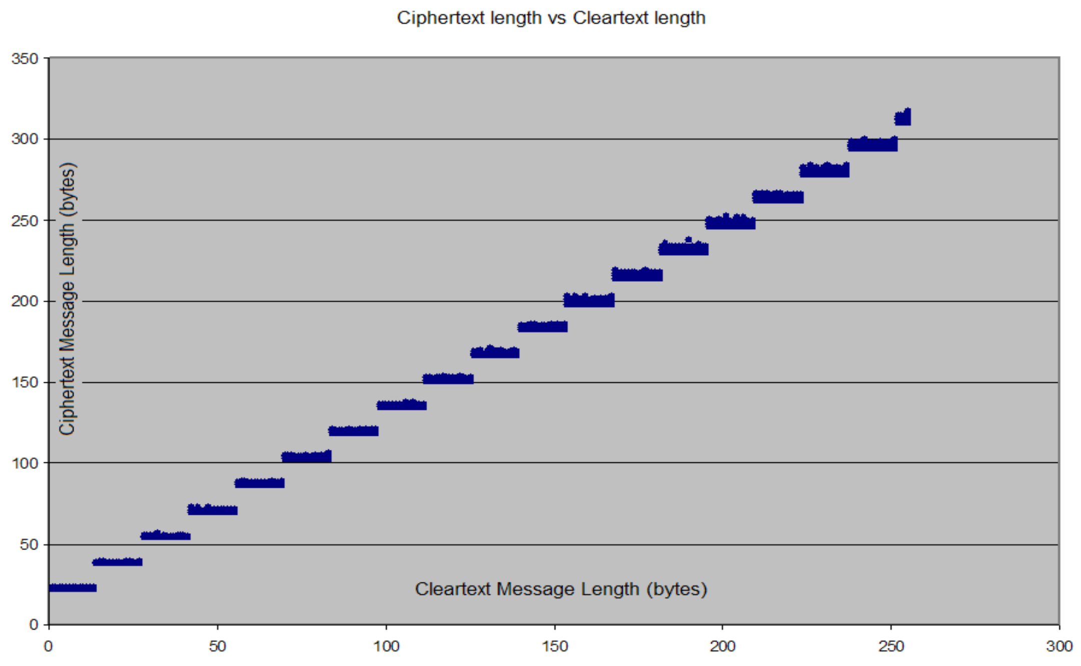
Overhead

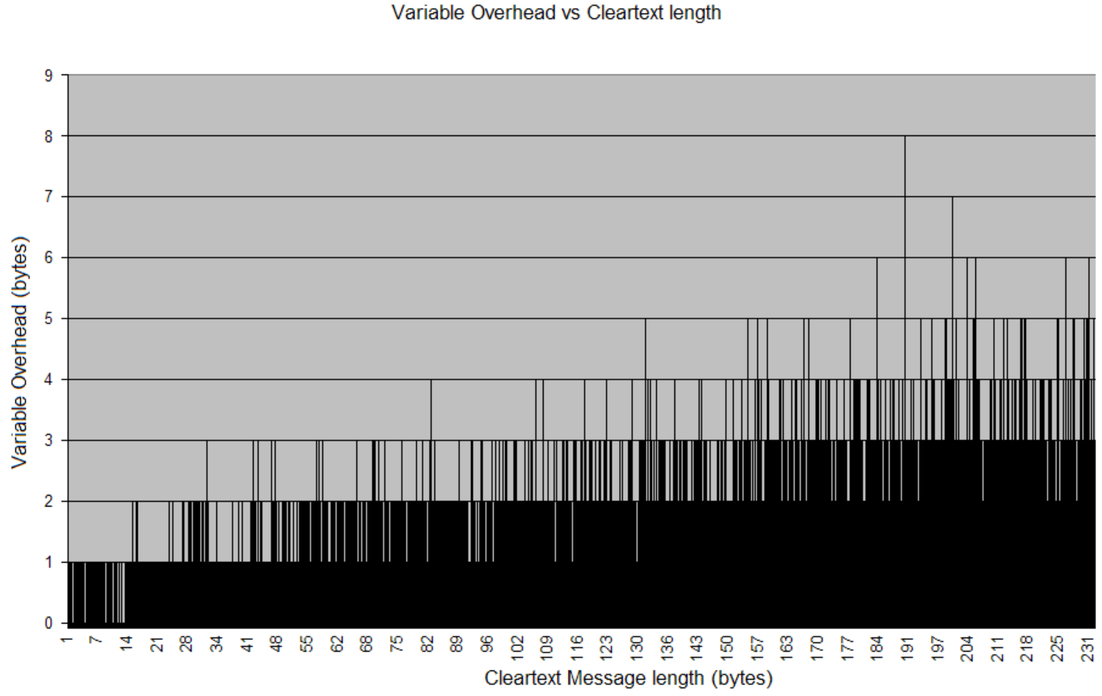
Total overhead independent of message length: 9 bytes
Fixed overhead per block (in this case, middle and last blocks) of data: 3 bytes
Framing Bytes used: Yes. Message starts with the two bytes “0x10 0x02” and ends with “0x10 0x03”
Probability of one or more framing escape characters appearing in a block of ciphertext: .060702

Charts Attached

Notes:
In initial tests the first block for every message was 44 bytes long. This seemed too long and it was deduced that the CM was running session establishment for every message. This issue was resolved by setting the CM session inactive timeout value appropriately.

Over the course of these tests, it was also noted that the third and fourth bytes (the two bytes after the 0x10 and 0x02 framing bytes) in every message did not change randomly enough. It is assumed that these bytes are just part of the CM overhead.


Ciphertext length vs Cleartext length
Ciphertext Message Length (bytes)
Cleartext Message Length (bytes)
0
50
100
150
200
250
300
350


Variable Overhead vs Cleartext length
Variable Overhead (bytes)
Cleartext Message length (bytes)
0
1
2
3
4
5
6
7
8
9
1
7
14
21
28
34
41
48
55
62
68
75
82
89
96
102
109
116
123
130
136
143
150
157
163
170
177
184
191
197
204
211
218
225
231

C.2.1.1.6 Interpreting and analyzing the results

Payload

The payload is the number of cleartext bytes per block of ciphertext data. A higher payload to block length ratio correlates to more cleartext data being communicated per message.

Total overhead independent of cleartext message length:

The overhead independent of cleartext message length can be easily evaluated by examining the fixed overhead in a single block message. This is just the ciphertext length of a single block message minus the payload for that block. In the example in C.2.1.1.4.2 this value is 22 – 13 = 9. In systems where most messages are short and a typical message just spans a few blocks of data, this overhead can have a considerable impact on performance.

Fixed overhead per block:

This is the fixed overhead in any middle block. It is just the block length minus the payload. The fixed overhead per block for the example in C.2.1.1.4.2 is 16-14= 2 bytes.

Chart 1: Cleartext message length vs. Ciphertext message length

This data, when graphed, should represent a step function as in Chart 1 for all CMs that use block encryption. The chart provides a better understanding of block encryption. The block boundaries lie between each step. Each step corresponds to a block in the ciphertext. The width of each step is essentially the same throughout the chart, and is the payload of the CM.

Chart 2: Histogram of variable overhead as cleartext message length increases.

This histogram helps to assess the variable overhead in Test 3. Each vertical line in this chart corresponds to a cleartext message length from Test 3 and represents the variable overhead in bytes for that message.

C.2.1.1.7 Definitions:

Block: A group of sequential bytes. Block encryption algorithms encrypt blocks of cleartext data into an encrypted block of data.

Block length: Length of a middle block with no variable overhead in bytes.

Boundary: The point at which increasing the length of a cleartext message by one byte, will cause the ciphertext message length to increase by one block length bytes.

Cleartext: Unencrypted data without format additions or changes, such as framing or padding.

Payload: The maximum number of encrypted cleartext bytes that fit into one block of encrypted data.

Plaintext: Unencrypted data with format additions or changes, such as framing or padding.

Overhead: Any data added to the encrypted cleartext bytes so as to direct or control the transfer of encrypted data or for error checking.

C.2.1.1.8 Formulas

Probability (P) of (m) framing escape bytes or more appearing in a block of cipher text[5]:

$P(x) = \sum_{n=m}^{x} [(xCn\,(255)^{x-n})/256^{x}]$ where $xCn = [x!/((x-n)!\,n!)]$

x is the block length
P is the probability
The equation calculates the probability for (m) or more framing escape bytes appearing in the cleartext.

Calculating the expected ciphertext length (without variable overhead) for a given cleartext length:

$$L = \begin{cases} C_1 & \text{for all } x \mid 0 < x \leq p_l \quad x \in N \\ C_2 & \text{for all } x \mid p_l < x \leq p_f + p_l \quad x \in N \\ 16 * \text{ceiling}[(y - p_l - p_f)/BL] + C_1 + C_2 & \text{for all } x \mid x > p_f + p_l \quad x \in N \end{cases}$$

L is the expected ciphertext length
C_2 is ciphertext output for a small message spanning just two blocks (the first and last one)
C_1 is ciphertext output for a small message spanning just one block (the last one)
p_f is the payload for the first block
p_l is the payload for the last block
p is the payload for any middle block[6]
BL is the block length
y is the cleartext length

$C_1 = H + Tr + BL$
$C_2 = H + 2*BL + Tr$
H is the number of header bytes
Tr is the number of trailer bytes

[5] This probability is assuming that the ciphertext output for a given cleartext is pseudorandom in every case.
[6] If the payload is not constant for all middle blocks a new formula needs to be formulated.

C.2.1.1.9 Test Results Sheet

Test Form

Test Name Block Length Probing	Test Type (Bench / ~~Testbed~~ / ~~Field~~)

Date:
Company/Organization:
Test Performed By: Position:

Cryptographic Module information

Make: Model: Type:
Firmware Version:
Other information:

Cryptographic Protocol information

Note all relevant information regarding the CM's cryptographic protocol including:

- Encryption algorithm used:
  - Key length:
- Authentication algorithm
- Integrity method
- Mode of operation

Test Configuration

Note the following information:

- Communications protocols used on various links (if any)
- Use of custom connectors and/or Null modems
- Host controller information
  - o Make, Model, Processor and Operating System
  - o Real-time control system host software used
- Communication parameters on various links
  - o Baud rate
  - o Data bits
  - o Stop bits
  - o Parity bits
  - o Full/half duplex
  - o Flow control
  - o Are communications parameters negotiated or ever changed ?

- All relevant parameters used for the GTI software tool for Test 3
    - Start Message Length
    - End Message Length
    - Number of times each message repeated
    - Test pattern used as message
    - Timeout
    - Time between consecutive messages
    - Software version
    - Other

Results

Payload for the 1st block:
Payload for the last block:
Payload for the middle blocks:
Block length:
Length of first block:
Length of last block:

End Message Length for Test 2:
Number of times message is repeated:
Mean length of ciphertext messages for the longest cleartext message:
Standard Deviation Mean:

Total overhead independent of cleartext message length:
Fixed overhead per block:
Message framing used: (Y/N)
Message framing bytes:
Probability of the framing escape byte appearing in a block of ciphertext:
Block framing used: (Y/N)
Block framing bytes:

Notes

C.2.2 Effect of Message Content on Latency

C.2.2.1 Introduction

The purpose, background, procedure and evaluation as well as interpretation of results of testing the effect of message content on latency are described.

C.2.2.1.1 Purpose

The purpose of this test is to determine whether the latency associated with cryptographic modules in a cryptographic system varies based on the message content.

C.2.2.1.2 Background

For a given message length, comparing the latency average and standard deviation for various data patterns should reveal any variation in latency due to message content [1]. These variations can help evaluate the performance of a CM.

Variations in latency due to message content do not correlate to a poor CM design. However, large variations could be the result of a poor or flawed design and therefore, should be investigated.

It is not feasible to test every possible native protocol message for variations in latency. Therefore, test messages are used instead. The test messages, constructed from bytes containing the following data patterns, should reveal most variations in latency due to the message content [1]:

1. All zeros
2. All 1's
3. Alternating 1's and zeros with bit zero equal to 1
4. Alternating 1's and zeros with bit zero equal to 0
5. Ascending binary count
6. Descending binary count
7. Random values

C.2.2.1.3 Test Equipment and Software

This test requires the following equipment:

- PC
- Two identical CMs to be tested
- Serial cables
- Null modem (if necessary)
- GTI/NIST software tool or similar tool(s) to help run this test

C.2.2.1.4 Procedure

C.2.2.1.4.1 General Procedure
The general procedure for testing the effect of message content on latency is as follows:

- Connect the two CMs and any other needed equipment such that the latency can be measured accurately when a chosen cleartext message is transmitted via the CMs.
- For a particular message length, construct a test message by repeating bytes consisting of one of the data patterns discussed above. Send the test message through the cryptographic system and measure the latency. Repeat this process for all the other remaining data patterns.
- Calculate the average and standard deviation of the measured latencies for each message length. Trace any statistically significant variations[7] back to the corresponding test message.
- Repeat the above two steps for various different message lengths.

C.2.2.1.4.2 Detailed Procedure

GTI has developed a software tool to automate this test. The following steps make use of this tool. Refer to the documentation associated with the tool for further information.

Setup the real-time process control equipment as shown in Configuration 1.

Select the appropriate configurations in the software tool.

In the *Test Values* tab select the range of message lengths to be tested. The user should choose to test each test message multiple times per message length to obtain more accurate data.

Start the test. When the test is completed, the latency results will be available in the log files. These results can easily be imported to a spreadsheet program such as MS Excel®. The averages and standard deviations can be graphed for further examination.

Any statistically significant variations in latency should be further investigated by exploring the detailed log files. If the variations can be traced back to one or more test messages, the message length as well as the test messages should be noted. Examine the log files to check if these test messages cause a variation in latency for all or many message lengths.

Run the test again for the select message lengths that had a high standard deviation. Compare the results with the previous run to verify the variation in latency.

[7] A statistically significant variation is a relative term. The tester should determine what an acceptable variation is. If the tester uses the GTI software tools to conduct the test, he/she should keep in mind that at best, the software only measures latency with accuracy up to ± 1 ms (see the related help files for further information).

C.2.2.1.5 Test Example

GTI tested the AGA 12 prototype retrofit cryptographic modules for the effect of message content on latency. The prototype CMs ran the ScadaSafe implementation[8] of the AGA 12, Part 2 protocol on the Arcom Viper® development boards. The software developed by GTI was used to help perform the test. GTI ran the test for messages ranging from 4 to 400 bytes in length and repeated each message 200 times. The log files generated by this test are not provided here. Instead, the results were plotted and two charts are attached along with a complete test form:

Standard deviation of latencies for chosen messages vs. Message length
Average latency vs. Message length.

C.2.2.1.5.1 Results

Test Name Effect of Message Content on Latency	Test Type (Bench / ~~Testbed~~ / ~~Field~~)

Date: 05/17/04
Company/Organization: GTI
Test Performed By: Aakash Shah | Position: Asst. Electronics Engineer

Cryptographic Module information

Make: Prototype | Model: --- | Type: ---
Firmware Version: ScadaSafe 0.6.7
Development board information*:*
Arcom Viper M64-F32 running AEL v3i5

Cryptographic protocol information

Module was AGA 12, Part 2 compliant
Encryption algorithm used: AES
Key length: 128
Authentication and integrity: 4 byte HMAC
AGA 12 Cipher Suite: 2 (PE Mode)

Test Configuration

[8] The ScadaSafe prototype is not optimized for performance. Therefore, performance data collected in this test may not reflect the performance commercial product.

Note the following information:

- Make, model and/or type of the different kinds of communications equipment

  Used standard RS 232 cable
- Use of custom connectors and/or Null modems

  Null modems on both connections to the PC
- Host controller information
  - Make, Model, Processor and Operating System

    Make: Dell

    Model: D600

    Processor: Intel Pentium 4.

    Operating System: Windows 2000 Professional
  - Real-time control system host software used

    GTI software tool
- Communications parameters: baud rates, start/stop/parity bits, full/half duplex, flow control, if negotiated, ever changed

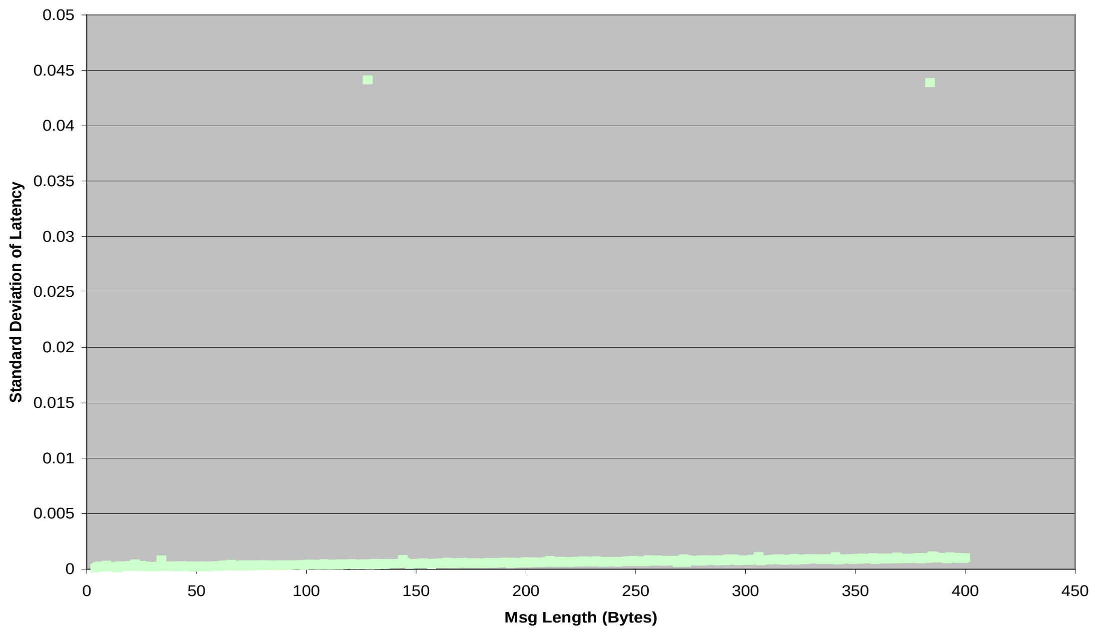
  9600 8n2, half duplex, no flow control
- Any specific operating modes

  None
- Note all relevant parameters used for the GTI software tool.
  - Start Message Length: 4 byte
  - End Message Length: 400 byte
  - Number of times each test message repeated: 200
  - Timeout: 5 sec
  - Time between consecutive messages: 4 ms

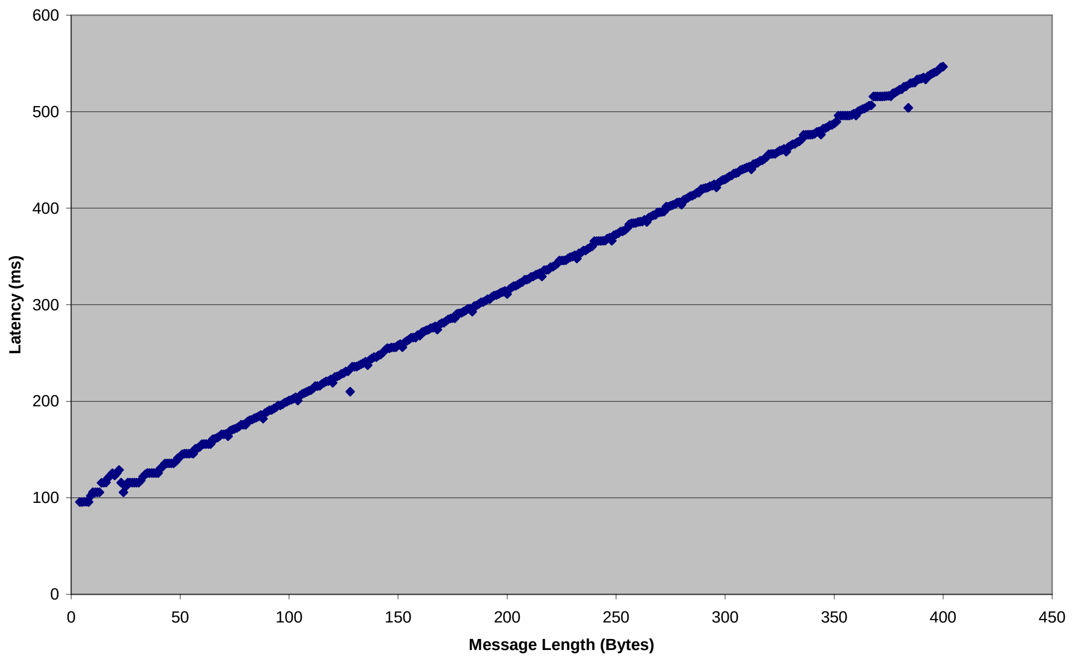
Results

Charts are attached.

Std. Deviation vs. Message Length



Average Latency vs. Message Length



Significant variations noted:

The standard deviation of the measured latencies for the 128 byte message as well as the 384 byte message had higher deviations than the rest of the messages. The detailed log file recorded the following latency averages for the seven different test messages:

128 byte message	384 byte message
0 : 0.237	0 : 0.532
1 : 0.236	1 : 0.529
2 : 0.236	2 : 0.529
3 : 0.236	3 : 0.529
4 : 0.235	4 : 0.529
5 : 0.148	5 : 0.444
6 : 0.142	6 : 0.436

The test was conducted again for the same message lengths and similar results were obtained.

Comments:

The CMs had a lower measured latency for the test patterns with the descending binary count as well as random values for the messages of length 128 and 384. It should be noted that the prototype CM is not optimized for performance and it may take varying times to perform the same calculation. Since both message lengths in question above are multiples of 16 (the payload for each block of encrypted data), it may be the case that the CM took more or less time to calculate the HMAC in the trailer.

C.2.2.1.6 Interpretation of Results

There are four types of relevant results possible for this test:

1) No significant variations for any test messages.
2) One or more test messages has a significantly higher/lower than average latency for all message lengths, i.e. outside the bounds of statistical uncertainty.
3) One or more test messages has a significantly higher/lower than average latency for a particular message length, i.e. outside the bounds of statistical uncertainty.
4) A combination of 2 and 3.

Results of type 1 may indicate no measured effect on latency due to message content. Results of type 2, 3 and 4 may imply some effect of message content on latency but do not offer specific conclusions regarding CM design and/or performance.

The tester should be aware of the resolution and accuracy of the system used to measure latency and should be aware of its impact on the experiment. These issues regarding the GTI/NIST software tool are discussed in the related help files. Knowing these values will help the tester parse out the statistically significant data.

One possible reason for a higher than average latency is the variable overhead in the cryptographic protocol (see C.2.1.1.2.1.2). It is possible that the ciphertext generated for a particular test message always contains the framing escape character, thereby, increasing the latency. This implies that the ciphertext is not pseudo-random – a key prerequisite for a good cryptographic algorithm. To determine if variable overhead is indeed the cause for the above average latency, the user can optionally use the block length probing software tool (see section C.2.1.1.8), select the appropriate test value and message length(s), and examine the resultant ciphertext for the occurrence of the framing character.

The performance of a CM cannot be gauged by the results of this test alone. However, any timing reports provided by the vendor can be compared to the measured latency values for correctness.

C.2.2.1.8 Test Results Sheet

Test Name	Test Type
Effect of Message Content on Latency	(Bench / ~~Testbed~~ / ~~Field~~)

Date:
Company/Organization:
Test Performed By: Position:

Cryptographic Module information

Make: Model: Type:
Firmware Version:
Other information:

Cryptographic Protocol information

Note all relevant information regarding the CM's cryptographic protocol including:

- Encryption algorithm used:
    - Key length:
- Authentication algorithm
- Integrity method
- Mode of operation

Test Configuration

Note the following information:
- Make, model and/or type of the different kinds of communications equipment
- Use of custom connectors and/or Null modems
- Host controller information
    - Make, Model, Processor and Operating System
    - Real-time control system host software used
- Communication parameters on various links
    - Baud rate
    - Data bits
    - Stop bits
    - Parity bits
    - Full/half duplex
    - Flow control
    - Are communications parameters negotiated or ever changed?
- Any specific operating modes of the real-time control system
- All relevant parameters used for the GTI software tool.
    - Start Message Length
    - End Message Length

- Number of times each test message repeated
- Timeout:
- Time between consecutive messages
- Test pattern(s) used
- Software version
- Other

Results

Noted significant variations:

Charts:

Notes

C.2.3 Throughput Tests

C.2.3.1 Introduction

The purpose, background, procedure and evaluation as well as interpretation of results of testing the throughput of a cryptographic system are described.

C.2.3.1.1 Purpose

The purpose of this test is to determine impact of cryptography on the throughput of a real-time process control system.

C.2.3.1.2 Background

Throughput testing is used to measure the maximum sustainable rate of real-time process control messages per hour (MPH). A large number of transaction requests will stress the CM's ability to buffer the incoming messages, encrypt the message, and buffer the encrypted message to be sent to the receiving CM [1].

Testing the throughput of a cryptographic system is the great way to assess its performance. The throughput tests have three steps – the lab test, test bed evaluation as well as field test. These steps are described in greater detail in the procedure that follows (C.2.3.1.4).

The following ideas on throughput testing have been reproduced from Annex H of AGA 12:

> Throughput in payload bits per second is expected to be dependent on message length, due to block padding and other overhead. Since it is desirable to treat the CM as a black box, ideally the throughput would be measured for every realistic message length. Measurements should be made at expected data rates to be used. If throughput testing cannot be automated, and the size of the largest block is known, it should be sufficient to measure throughput for messages up to three blocks long, with extrapolations based on the throughput of the middle block (since the first and last blocks may have special overheads). The delta time between a two and a three-block message (because the two block message has first and last blocks) should be measured. Units for reporting throughput should be bits per second and messages per second.
>
> Alternatively, reporting the inverse of the throughput (seconds per message) makes it easier to compute polling intervals, and also coincides with the definition of latency used here (see Appendix B).

To obtain accurate results it is advisable to test the throughput for a long period of time. The tester should realize the differences between a unidirectional throughput test and a bi-directional throughput test. In a unidirectional throughput test, messages are continuously sent one way from the encrypting CM to the decrypting CM. But, since the ciphertext message will always be greater than or equal to the cleartext message, continuous sustained throughput is not possible without a inserting a delay between messages. This is not the case with a bidirectional test, where the message is sent one way from the host to the slave and then back from the slave to the host.

AGA 12 discusses three different types of configurations for the throughput tests:

Baseline configuration
The baseline test configuration should not include CMs or any loading other than that introduced from application processors. This configuration establishes the maximum MPH over the communication path.

Baseline with CMs
Test configurations that include CMs will be used to measure the realized MPH over the same communication path.

Baseline with CMs and other loads
Test configurations that include CMs and other loading will be used to measure the realized MPH over the same communication path

The degradation can then be calculated as 1- (the realized MPH divided by the maximum MPH).

The throughput of a system is sensitive to numerous different factors. Some of these factors are listed below:

- Polling Frequency

  The polling frequency of the real-time process control host software or any other host emulation software (such as the GTI software) will directly affect the recorded throughput of a system. The GTI software is configurable and can be set to poll continuously. In actuality though, the host may poll at a certain polling frequency (e.g. once per second), leaving large silences of 100's of milliseconds on the communications line. In such cases, the added latency due to cryptography may be undetectable to the real-time process control operator. This is not the case if the host software, instead of polling at a certain frequency, waits a configurable amount of time before sending out the next consecutive query.

  The test procedures described below for the lab test as well as testbed evaluation require the GTI software tool to be configured to poll continuously. However, the GTI software tool may be configured to poll at a particular polling frequency.

- The measurement method

  The measurement technique can add overhead to each message, thus reducing the overall throughput of the system. Therefore, the measured throughput in payload bits/second for a small message will be lower than that of a larger message. The GTI software may add some minor overhead per message. Refer to software documentation for further detail.

- Message length
  The message length will directly affect the throughput. A long message that translates to multiple blocks of ciphertext will have a much higher throughput than a small message.

- Communications Media
  The communications medium (and any equipment associated with it) used to in the test configuration may also affect the recorded throughput. For example, the throughput in a dial-up modem network may be considerably lower than that in a wireless network due to connection time required in the dial-up modem network.

- Serial Port Configuration
  The baud rate, number of parity bits, stop bits as well as data bits directly affect the throughput.

- Test configurations
  A faster communications channel between the CM and the real-time process control system host/slave may provide higher throughput.

- Topology
  The network topology may also affect the recorded throughput.

- The real-time process control slave device
  In the test bed evaluation as well as field tests, the throughput may differ because of varying response times of different slaves. The slaves themselves may have varying response times for different messages.

- Cryptographic Module design
  The throughput of the real-time process control system will obviously depend on factors such as the additional latency introduced by a cryptographic module in encrypting and decrypting the data. But even architectural design choices may affect the throughput. For example, cryptographic modules designed to individually protect each modem versus ones designed to protect the link between a server and a bank of modems will have varying throughputs.

Because the throughput is dependent on so many variables, it is imperative that the tester maintain a consistent test environment throughout the throughput tests.

C.2.3.1.3 Test Equipment

- PC
- Three identical CMs to test
- Null Modem (if necessary)
- Serial Cable
- Real-time process control master software
- Three real-time process control system slave devices
- Line Analyzer Box
- GTI/NIST software tool

C.2.3.1.4 Procedure

C.2.3.1.4.1 General Procedure

The general idea behind the throughput tests is to continuously send messages through the real-time process control system for a predetermined period of time and record the number of messages that are communicated correctly for the different configurations specified by AGA 12. The degradation can then be calculated from the results.

C.2.3.1.4.2 Detailed Procedure

GTI has developed a software tool to help conduct the throughput tests. The following procedure uses this tool. Refer to the software documentation for further detail.

Start the test tool and select the appropriate configurations. Choose the amount of time to run the test as well as the various message lengths (or valid messages) to be tested.

As discussed in earlier, the user must determine the minimum delay needed between messages to conduct the unidirectional throughput test. This information may be available in the manufacturer's documentation. If this information is not available, the delay can be roughly determined by trial and error.

- Pick a test message with a length that is one byte more than the payload of the CM.
- Continuously send this test message through the cryptographic system. If the CM concludes the end of message by a set timeout, the delay between these messages should at least be the timeout value.
- If the CM fails or crashes, increase the delay between messages and run the above step again.
- Once sustained functioning is achieved, note the delay used. This is the delay that will be used in the following lab tests.

C.2.3.1.4.1 Lab Test

C.2.3.1.4.1.1 Baseline configuration

Setup the test equipment as shown in Configuration 1. Run the unidirectional throughput test for the predetermined time for each selected message length. Record the number of messages communicated correctly. Calculate the MPH as well as payload bits per second. Run the test again for the bidirectional case with no delay between messages.

C.2.3.1.4.1.2 Baseline with CMs

Setup the test equipment as shown in configuration 3. Run the unidirectional throughput test for the predetermined time for each selected message length. The delay between messages should be the delay determined above. Record the number of messages communicated correctly. Calculate the MPH as well as payload bits per second. Run the test again for the bidirectional case with the only delay between messages being the one needed by the CM to denote the end of message.

C.2.3.1.4.1.3 Degradation

Calculate degradation for each tested message length.

C.2.3.1.4.2 Test bed evaluation

The test-bed evaluation incorporates the slave devices into the lab test. On the host side the user has the choice to employ an actual real-time process control host or emulate the host by running the GTI software tools on a PC.2.3.

The tester should run this test for at least the following three topological configurations:

Point to Point (See Configurations 4 and 5)
Point to Multipoint (See Configurations 6 and 7)
Point to Multipoint in mixed-mode (See Configurations 6 and 8)

The tester may add other configurations based on his/her need.

To use the GTI software tools, valid polls as well as corresponding responses must be known or obtained by using the line analyzer box (see Configuration 11). The message should be carefully picked such that the slave response to the message is always the same length. If testing a point to multipoint topology, the slave polls and responses should be of the same length respectively. This can be easily accomplished by sending a standard read command for a predefined range of register addresses to each slave. The procedure below utilizes a PC along with the GTI software tools to emulate the real-time process control host and measure throughput.

If an actual real-time process control host is used, the same message(s) (for example, a continuous poll) should be transmitted repeatedly. This way the message(s) will be stored in the real-time process control host's cache, and will be retrieved speedily. The number of messages communicated successfully should be recorded.

C.2.3.1.4.2.1 Baseline configuration

Setup the test equipment in the baseline configuration associated with the topological configuration being tested (Configuration 4 or 6).

Input a valid message(s) to be tested as well as the corresponding slave response(s) into the throughput program. Run the throughput test for the predetermined time. The delay between messages should be the delay required by the slave(s) to function properly. Record the number of messages communicated correctly. Calculate the MPH as well as payload bits per second.

Repeat the above steps for each set of valid message(s) to be tested.

C.2.3.1.4.2.2 Baseline with CMs and other loads

Setup the test equipment in the baseline configuration with CMs associated with the topological configuration being tested (Configuration 5, 7 or 8)

Follow the same steps as the baseline configuration throughput test (C.2.3.1.4.2.1).

C.2.3.1.4.2.3 Degradation

Calculate degradation for each message tested.

C.2.3.1.4.3 Field tests

The field test configuration should be documented thoroughly. To get the best results, the same message(s) should be transmitted repeatedly. This can be done by continuously polling the slave(s). The length(s) of the message(s) being repeated as well as the corresponding response(s) should be obtained by using the line analyzer box (see Configuration 11). The number of messages communicated successfully should be recorded.

The tester should run this test for at least the following three topological configurations:

Point to Point
Point to Multipoint
Point to Multipoint in mixed-mode

C.2.3.1.4.3.1 Baseline configuration

Continuously transmit the known sequence of messages to the slaves for the predetermined time. Record the number of messages communicated correctly. Calculate the MPH as well as payload bits per second.

C.2.3.1.4.3.2 Baseline with CMs and other loads

Once the CMs are installed in the field and are operational, follow the same steps as discussed in section C.2.3.1.4.3.1.

C.2.3.1.4.3.3 Degradation

Calculate degradation.

C.2.3.1.5 Test Example

GTI conducted the throughput tests on the AGA 12 prototype retrofit cryptographic modules. The prototype CMs ran the ScadaSafe implementation of the AGA 12, Part 2 protocol on the Arcom Viper® development boards. The software developed by GTI was used to help perform the test. The test results are provided below[9].

C.2.3.1.5.1 Results

Test Name	Test Type
Throughput tests	(~~Bench~~ / Testbed / ~~Field~~)

Date: 05/28/04
Company/Organization: Gas Technology Institute
Test Performed by: Aakash Shah

Cryptographic Module information

ScadaSafe prototype running on an Arcom Viper M64 – F32 board.
ScadaSafe Version: 0.6.7

Cryptographic Protocol information

Algorithm used: AES
Key length: 128
Authentication: 4 byte HMAC
Mode: AGA 12 – Part 2 Cipher Suite 2 (PE Mode)

Divide the real-time process control network being tested into logical sections based on the communications mediums and speeds. Complete this test sheet for each one of these sections.

Test Configuration

Draw a detailed communications system diagram depicting the serial real-time process control network topology along with any LAN connections.

Note the following information:

- Approximate geographic distances between components.
  On Bench
- Communications protocols used on various links
  Modbus RTU

[9] The ScadaSafe prototype is not optimized for performance. Therefore, performance data collected in this test may not reflect the performance of a commercial product.

- Make, model and/or type of the different kinds of communications equipment
    - Acromag 913 MB I/O modules
- Use of custom connectors and/or Null modems
    - Null modem & custom connector between Host PC and CM
- Host controller OS and real-time process control host software
    - Win2k, GTI test software
- Communications parameters: baud rates, start/stop/parity bits, full/half duplex, flow control, if negotiated, ever changed
    - 9600 8n2, half duplex, no flow control
- Any specific operating modes
    - None
- Polling scheme (master poll, report-by-exception, etc.), timeouts, and polling rates (e.g. poll every second, as fast as possible, etc.)
    - For results in Table 1a:
        - Master Poll | 1 sec host timeout | as fast as possible
    - For results in Table 1b:
        - Master Poll | 1 sec host timeout | 7200 polls per hour
    - For results in Table 1c:
        - Master Poll | 1 sec host timeout | as fast as possible

- Types of polls and responses and average lengths. If the set of polls is recurring, record the poll/response lengths. If possible note the slave processing time for each poll.
    - For the first test in table 1a as well as in table 1b
        - 4 byte poll, 43 byte response | Avg. slave processing time 30 ms
        - The poll queries the slave for its model number, serial number etc.

    - For the second test in table 1a, the following poll sequence was tested.
        - Polling sequence of 4 polls
        - 4 byte poll, 43 byte response
        - 8 byte poll, 7 byte response
        - 8 byte poll, 25 byte response
        - 8 byte poll, 127 byte response

- Note all relevant parameters used for the GTI software tool.
    - Timeout: 1 s
    - Time between consecutive poll/resp cycles: 4 ms
    - Test Time: 15 hours

Results

Record results separately for each different topological configuration.

Test Message Length (bytes)	Slave Response Length (bytes)	Baud Rate	Baseline Configuration				Baseline with CMs and other loads				Degradation	Avg. RTU processing time per message (if known)
			Test Time	Total # of messages communicated	Errors / Timeouts	Throughput (MPH)	Test Time	Total # of messages communicate d	Errors / Timeouts	Throughput (MPH / Payload bits/s)		
4	43	9600 (8n2)	15 hrs	837945	0	55863	15	213840	0	14256	.74	30 ms
4 8 8 8	43 7 25 127	9600 (8n2)	1 hr	36876	0	36876	1 hr.	13476	0	13476	.63	Not known

Table 1a – Polling as fast as possible

Test Message Length (bytes)	Slave Response Length (bytes)	Baud Rate	Baseline Configuration				Baseline with CMs and other loads				Degradation	Avg. RTU processing time per message (if known)
			Test Time	Total # of messages communicated	Errors / Timeouts	Throughput (MPH)	Test Time	Total # of messages communicated	Errors / Timeouts	Through put (MPH / Payload bits/s)		
4	43	9600 (8n2)	15 hrs	108000	0	7200	15	108000	0	7200	0	30 ms

Table 1b – Polling frequency of 7200 messages per hour

C.2.3.1.6 Interpretation of Results

As explained in section C.2.3.1.2, the recorded throughput of a system is highly dependent on a number of variables. This makes it very hard to objectively compare test results from throughput tests conducted independently (In such cases it may be easier to compare the latency – refer to the related help files on how to use the GTI software to measure latency). This also means that the throughput test results from the field tests are far more important in gauging the impact of the CMs on the real-time process control system than any lab or the test-bed evaluations.

C.2.3.1.7 Definitions

Degradation: The deterioration in performance of a real-time control system network.

Throughput: The amount of real-time control system messages communicated successfully through the given network over a period of time. Throughput is measured in messages per hour (MPH), message sequences per hour (MSPH) or payload bits per second.

C.2.3.1.8 Test Results Sheet

<u>Test Form</u>

<u>Test Name</u>	<u>Test Type</u>
Throughput tests	(Bench / Testbed / Field)

Date:
Company/Organization:
Test Performed By: Position:

Cryptographic Module information

Make: Model: Type:
Firmware Version:
Other information:

Cryptographic Protocol information

Note all relevant information regarding the CM's cryptographic protocol including:

- Encryption algorithm used:
    - Key length:
- Authentication algorithm
- Integrity method
- Mode of operation

Divide the real-time process control network being tested into logical sections based on the communications mediums and speeds. Complete this test sheet for each one of these sections.

Test Configuration

Draw a detailed communications system diagram depicting the serial real-time process control network topology along with any LAN connections.

Note the following information:

- Approximate geographic distances between components
- Communications protocols used on various links
- Make, model and/or type of the different kinds of communications equipment
- Use of custom connectors and/or Null modems
- Host controller information
    - Make, Model, Processor and Operating System
    - Real-time control system host software used

- Slave device information (if applicable)
    - Make, model and/or type
- Communication parameters on various links
    - Baud rate
    - Data bits
    - Stop bits
    - Parity bits
    - Full/half duplex
    - Flow control
    - Are communications parameters negotiated or ever changed?
- Any specific operating modes of the real-time control system
- Polling scheme (master poll, report-by-exception, etc.), timeouts, and polling rates (e.g. poll every second, as fast as possible, etc.)
- Types of polls and responses and average lengths. If the set of polls is recurring, record the poll/response lengths. If possible note the slave processing time for each poll
- All relevant parameters used for the GTI software tool
    - Timeout
    - Test Time
    - Time between consecutive poll/response cycles
    - Any polls/responses entered
    - Software version
    - Other

Results

Record results separately for each different topological configuration.

Test Message Length (bytes)[10]	Slave Response Length (bytes)[11]	Baud Rate	Baseline Configuration				Baseline with CMs and other loads				Degradation	Avg. RTU processing time per message (if known)
			Test Time	Total # of messages communicated	Errors / Timeouts	Throughput (MPH)	Test Time	Total # of messages communicated	Errors / Timeouts	Throughput (MPH)		

Table 1 – Test Results

[10] If applicable, record message lengths of a sequence of polls instead of a single poll

[11] If applicable, record message lengths of a sequence of responses instead of a single response

Notes

C.2.4 Clock Synchronization Test

C.2.4.1 Introduction

The purpose, background, procedure and evaluation as well as interpretation of results of the clock synchronization test are described.

C.2.4.1.1 Purpose

The purpose of this test is to determine impact cryptography on clock synchronization.

C.2.4.1.2 Background

Addition of cryptography to a real-time process control system may cause problems with clock synchronization. The clock synchronization test assesses the functionality and accuracy of this feature after the installation of cryptography to the SCADA system.

AGA 12 has the following recommendations for the clock synchronization test:

> Clock synchronization test should first be performed without cryptographic modules. The test should be repeated with cryptographic modules in place, and the differences measured.
>
> One approach is to use the SCADA system write and read clock commands to measure clock synchronization. If more precision is required, then a GPS time signal is generally used. Another approach is to use IEEE 1588 to synchronize clocks over a local area network.

Clock synchronization is an important issue especially for electric real-time process control systems. The accuracy of the clock synchronization depends on the method used to synchronize time. For example, the DNP protocol measures the path delay and offsets the time by this delay to synchronize the clock. This can be a fairly precise way to synchronize time if all the messages transmitted back and forth are of the same length. In a real-time process control system with no cryptography, slight differences in the message length may only lead to minor differences in the clock synchronization using this method. But when retrofit CMs are installed in this system, the same method may not be as precise. Depending on the lengths of the messages used to determine the path delay, an extra block of data may be communicated. In addition, variable overhead due to the appearance of the framing characters in the ciphertext may further affect the precision of the clock synchronization.

The following procedure uses commercial protocol simulators to test the clock synchronization test. A single PC with two serial ports can be also used to emulate the master as well as the slave units of the real-time process control system. The tester can implement the clock synchronization algorithm used by the communication protocol in software. Then user can view both clocks being set simultaneously and calculate the time difference between the host and slave clocks.

C.2.4.1.3 Test Equipment
This test requires the following equipment:

- PC
- Commercial protocol simulator - Triangle Microworks DNP3 test harness
- Two identical CMs
- Serial Cables
- Null modem

C.2.4.1.4 Procedure

C.2.4.1.4.1 General Procedure
Compare the precision of the clock synchronization with and without SCMs on a communications channel using the timestamps provided by the protocol simulator.

C.2.4.1.4.2

Perform the following steps for both Configuration 0 and 3.

The communication protocol write and 'read clock commands will be used to test the clock synchronization.

Run the commercial protocol simulator. Ensure that the messages being communicated back and forth between the master and slave units can be viewed along with a timestamp.

Set the clock using the write clock command. Note the timestamps on the message being sent from the master as well as the message being received by the slave. The difference between the timestamps should be the transmission time for the messages.

Read the clock using the read clock command. The difference between the timestamp on the received message and the time reported by the message should either reflect the transmission time for the message or will be 0 if the received time is already offset for the transmission time.

Repeat the above test many times to get a better idea on the clock differences between the master and slave units.

Record the differences on the read and write timestamps and calculate the average and standard deviation.

C.2.4.1.5 Test Example

GTI conducted the clock synchronization test for the DNP3 protocol on the AGA 12 prototype retrofit cryptographic modules. The prototype CMs ran the ScadaSafe implementation of the AGA 12, Part 2 protocol on the Arcom Viper® development boards. The Triangle Microworks DNP3 test harness was used to conduct the test. The results are provided below as an example[12].

C.2.4.1.5.1 Results

Test Name	Test Type
Clock Synchronization Test	Bench

Date: 11/14/04
Company/Organization: Gas Technology Institute
Test Performed By: Aakash Shah   Position: Asst. Electronics Engineer

Equipment Used:

Cryptographic Module Information

Make: Prototype   Model: Type:
Other Information:
AGA 12, Part 2 compliant ScadaSafe implementation
ScadaSafe v0.6.7
ScadaSafe configuration:
  Timeout: 20 character times
  Prebuffer: 14 character times

Cryptographic Protocol information
AGA 12, Part 2 - SSPP protocol
Note all relevant information regarding the CM's cryptographic protocol including:

- Encryption algorithm used: AES
  - Key length: 128
- Authentication (and integrity) algorithm: 4 byte HMAC
- Mode of operation: PE mode

PC information

Make: Dell   Model: Latitude D800   Processor: Pentium 4 1.4 GHz
Memory: 256 MB
Other: Dual Serial port PCMCIA card
Operating System: WinXP

[12] The ScadaSafe prototype is not optimized for performance. Therefore, performance data collected in this test may not reflect the performance of a commercial product.

Communication Protocol used: DNP3

Protocol Simulator information:
Triangle Microworks DNP3 test harness
List any relevant information regarding the commercial protocol simulator:
The test harness has two modes of time syncs – with and without offset i.e. the time sync is either just a simple write or the time is offset based on the round trip message transmission time.

Serial port configuration

Baud rate: 9600
Stop bits: 2
Parity: n
Data bits: 8
Flow control: none

Describe the algorithm used by the communication protocol to synchronize the clocks:

> The round trip communications time for a message is measured. The time sync write from the master, offsets the time by the measured round trip time divided by 2.

Draw a table with the recorded differences between clocks:

> Note: The test harness did not provide the functionality to measure the time stamp difference on the read command.

Configuration 1		Configuration 3	
Read timestamp difference	Write timestamp difference	Read timestamp difference	Write timestamp difference
n/a	45	n/a	105
n/a	55	n/a	0
n/a	130	n/a	120
n/a	120	n/a	110
Avg: n/a	Avg: 87.5 Std. Deviation: 43.68	Avg: n/a	Avg: 83.8 Std. Deviation: 56.18

C.2.4.1.5 Interpretation of Results

The clock synchronization test is highly dependent on the algorithm used by the communications protocol to synchronize time. In the above procedure, the performance of the commercial protocol simulator may also affect the results. The tester should ensure that the clock difference between the master and slave units is acceptable for the real-time process control system in place.

C.2.4.1.6 Test Form

Test Name Clock Synchronization Test	Test subset (if applicable)

Date:
Company/Organization:
Test Performed By: Position:

Cryptographic Module information

Make: Model: Type:
Firmware Version:
Other:

Cryptographic Protocol information

Note all relevant information regarding the CM's cryptographic protocol including:

- Encryption algorithm used:
  - Key length:
- Authentication algorithm
- Integrity method
- Mode of operation

Test Configuration

Note the following information:

- Use of custom connectors and/or Null modems
- Host controller information
  - Make, Model, Processor and Operating System
  - Real-time control system host software used
- RTU information (if applicable)
  - Make, model and/or type
- Communication parameters on various links
  - Baud rate
  - Data bits
  - Stop bits
  - Parity bits
  - Full/half duplex
  - Flow control
  - Are communications parameters negotiated or ever changed?

- Communications protocol used
- Describe the algorithm used by the communication protocol to synchronize the clocks
- Protocol Simulator information
    - List any relevant information regarding the commercial protocol simulator

Results

Recorded differences between clocks:

Configuration 1		Configuration 3	
Read timestamp difference	Write timestamp difference	Read timestamp difference	Write timestamp difference
…	…	…	…
Avg:	Avg:	Avg:	Avg:

C.2.4 Susceptibility of Adverse Conditions

C.2.5.1 Introduction

The purpose and background of testing cryptographic module's susceptibility to adverse conditions are described.

C.2.5.1.1 Purpose

The purpose of this test is to determine impact of adverse environmental conditions on the cryptographic modules used to protect the real-time process control systems.

C.2.5.1.2 Background

Annex H of AGA 12 offers the following background on testing the cryptographic module's susceptibility to adverse conditions:

> The CM should be tested to determine its ability to withstand (and perhaps even function despite) specific environmental conditions, such as transients, discharges, RF radiation, or extremes of humidity or temperature. IEEE 1613 describes some such tests [2].

Depending on the expected operating environment, there may be a variety of environmental specifications that the CM must comply with. There are numerous environmental testing facilities that test to these specifications. The real-time control system operator should require that the vendor supply certifications from accredited testing facilities.

A procedure will not be provided for this test, as almost any formal environmental test would require costly equipment. However, reliability field tests should be conducted at sites that offer adverse environmental conditions.

C.2.5 Effect of Noise

C.2.6.1 Introduction

The purpose, background, procedure and evaluation as well as interpretation of results of testing the effect of noise on the cryptographic module's performance are described.

C.2.6.1.1 Purpose

The purpose of this test is to determine impact of noise on the cryptographic modules used to protect the real-time process control systems.

C.2.6.1.2 Background

The CMs may be exposed to a lot of noise in the field. This test studies the effects of noise on CM performance. The test will be conducted in the lab as well as on the test-bed. Noise will be simulated using commercial equipment in a controlled manner and CM performance will be tested.

The tester should understand that simulated noise can never be a substitute for actual noise encountered in the field. But, the CM reaction to simulated noise is nevertheless important in determining if the CMs can maintain sustained functioning in the field. The reliability tests conducted in the field should give a better indication on how well the CMs handle noise.

The procedure that follows simulates noise using a MicroSeven© mini-PBX simulator. The tester may use alternate tools to simulate noise.

C.2.6.1.3 Test equipment

- PC
- Two identical CMs to test
- Null Modem (if necessary)
- Serial Cable
- Real-time process control host and slave device
- 2 Modems
- MicroSeven© mini-PBX simulator or similar equipment to generate noise at varying levels
- GTI/NIST software tool

C.2.6.1.4 Procedure

C.2.6.1.4.1 General Procedure

For Configurations 9 and 10, record the baseline throughput for a variety of message lengths. Note any errors, timeouts or CM failures. Using the noise simulator systematically increase the noise level and record the throughput of the same messages for each case.

C.2.6.1.4.2 Detailed Procedure

For both Configurations 8 and 9:

Run the Throughput.exe program. Refer to the documentation associated with the program for further information. Configure the settings appropriately and ensure that message verification is checked for Configuration 8. Choose the time to run the test as well as the length of the test message. Just as the throughput test procedure, determine the delay needed between messages to ensure proper functionality from the CM.

Run the throughput test for a predetermined amount of time. Record the throughput. From the log file, determine the number of messages communicated incorrectly due to errors and timeouts. If the CM fails or crashes there may be other issues that need to be resolved first.

Using the software that accompanied the MicroSeven© mini-PBX simulator, generate low noise on the line.

Run the throughput test again for the same amount of time and record the throughput. From the log file, determine the number of messages communicated incorrectly. If the CM fails or crashes record the number of messages communicated correctly.

Increase the noise and run the above step again.

Once the maximum noise level has been tested, tabulate the results.

Calculate the degradation in throughput.

C.2.6.1.5 Test Example

GTI tested the effect of noise on the AGA 12 prototype retrofit cryptographic modules. The prototype CMs ran the ScadaSafe implementation of the AGA 12, Part 2 protocol on the Arcom Viper® development boards. The software developed by GTI was used to help perform the test. The MicroSeven LS200 PBX was used to generate the noise. The results are provided below as an example[13].

C.2.6.1.5.1 Results

Test Name	Test Type

[13] The ScadaSafe prototype is not optimized for performance. Therefore, performance data collected in this test may not reflect a commercial product.

Effect of Noise	(~~Bench~~ / Testbed / ~~Field~~)

Date: 01/06/05
Company/Organization: Gas Technology Institute
Test Performed by: Aakash Shah

Cryptographic Module information

ScadaSafe prototype running on an Arcom Viper M64 – F32 board.
ScadaSafe Version: 0.6.7

Cryptographic Protocol information
Algorithm used: AES
Key length: 128
Authentication: 4 byte HMAC
Mode: AGA 12 – Part 2 Cipher Suite 2 (PE Mode)

Test Configuration

Note the following information:

- Communications protocols used on various links
  Modbus RTU
- Make model and/or type of noise simulator
  MicroSeven LS200 PBX
  - Describe the process as well as any different modes used by the noise simulator to generate noise
    The LS 200 provides 3 modes that add noise to the line:
    - True line Impairment mode
    - Pseudo line Impairment mode
    - Non-pseudo line impairment mode.

    The Non-pseudo line impairment mode along with random noise and frequency attenuation was selected for this test.

- Make, model and/or type of the different kinds of communications equipment
  Acromag 913 MB I/O modules
- Use of custom connectors and/or Null modems
  Null modem & custom connector between Host PC and CM
- Host controller OS and real-time process control host software
  Win2k, GTI test software
- Communications parameters:
  - Baud rate: 9600
  - Data bits: 8
  - Stop bits: 2

    - Parity bits: 0
    - Full/half duplex: half duplex
    - Flow control: None
    - Are communications parameters negotiated or ever changed? N
- Any specific operating modes

  None
- Polling scheme (master poll, report-by-exception, etc.), timeouts, and polling rates (e.g. poll every second, as fast as possible, etc.)

  Master Poll | 1 sec host timeout | as fast as possible
- Types of polls and responses and average lengths. If the set of polls is recurring, record the poll/response lengths. If possible note the slave processing time for each poll.

  8 byte poll, 127 byte response | Avg. slave processing time 60 ms
  The poll queries the slave for its holding register contents.
- Note all relevant parameters used for the GTI software tool.

  Timeout: 5 s
  Time between consecutive poll/resp cycles: 4 ms
  Test Time: 3 hours / message

Results

Baseline case without CMs:

Test Message Length (Poll / Response): 8/127						
Noise Level	Timeouts	Errors	CM Failure (Y/N)? If yes, provide short description.	Other Equipment Failure (y/n) If yes, provide short description.	Retries (Classify retries by equipment e.g. Host, CM and comm. equipment)	Throughput / Number of Messages communicated before fatal failure
00	91	0	N	N	0	46236
06	250	0	N	N	0	41053
0C	395	0	N	N	0	33654
12	0	0	N	Yes, the modem could not connect due to noise	Host tried redialing 10 times	0

Baseline case with CMs:

Test Message Length (Poll / Response): 8/127							
Noise Level	Timeouts	Errors	CM Failure (Y/N)? If yes, provide short description.	Other Equipment Failure (y/n) If yes, provide short description.	Retries (Classify retries by equipment e.g. Host, CM and comm. equipment)	Throughput / Number of Messages communicated before fatal failure	Degradation
00	105	0	N	N	0	18144	.61
06	270	0	N	N	0	15370	.63
0C	562	0	N	N	0	11352	.66
12	0	0	N	Yes, the modem could not connect due to noise	Host tried redialing 10 times	0	0

C.2.6.1.6 Interpretation of Results

As mentioned earlier, simulated noise cannot substitute for real noise caused by environmental conditions. If the CM repeatedly fails to communicate the information correctly under low noise conditions, it may not be ready to test in the field.

The modems used in the test may also be the cause of CM failure in some cases. The modems in the point to point connection may have trouble staying connected over a noisy line. The test may be repeated optionally with different modems to check if this is the case.

The recorded throughputs under various noise conditions provide the tester with a more realistic degradation value.

C.2.6.1.6 Test Results Sheet

<u>Test Form</u>

<u>Test Name</u> Effect of Noise	<u>Test Type</u> (Bench / Testbed / ~~Field~~)

Date:
Company/Organization:
Test Performed By: Position:

Cryptographic Module information

Make: Model: Type:
Firmware Version:
Other information:

Cryptographic Protocol information

Note all relevant information regarding the CM's cryptographic protocol including:

- Encryption algorithm used:
    - Key length:
- Authentication algorithm
- Integrity method
- Mode of operation

Test Configuration

Note the following information:

- Communications protocol used
- Make model and/or type of noise simulator
    - Describe the process used by the noise simulator as well as any different modes used to generate noise
- Make, model and/or type of the different kinds of communications equipment
- Use of custom connectors and/or Null modems
- Host controller information
    - Make, Model, Processor and Operating System
    - Real-time control system host software used
- Slave device information (if applicable)
    - Make, model and/or type
- Communication parameters on various links
    - Baud rate
    - Data bits
    - Stop bits
    - Parity bits
    - Full/half duplex
    - Flow control
    - Are communications parameters negotiated or ever changed?
- Any specific operating modes of the real-time control system
- Note all relevant parameters used for the GTI software tool.
    - Timeout
    - Time between consecutive messages or poll/response cycles
    - Test Time
    - Software version
    - Other

Results

Complete the following table for each different test message length tested, for both baseline cases – with and without CMs..

Test Message Length (Poll / Response):							
Noise Level	Timeouts	Errors	CM Failure (Y/N)? If yes, provide short description.	Other Equipment Failure (y/n) If yes, provide short description.	Retries (Classify retries by equipment e.g. Host, CM and comm. equipment)	Throughput / Number of Messages communicated before fatal failure	Degradation for baseline case with CMs
Lowest Noise Level							
…							
…							
…							
…							
Highest Noise Level							

\

Table 1: Lab test for the effect of noise

C.3 Operability Tests

This section describes the operability tests in detail.

C.3.1 Bottleneck Identification

C.3.1.1 Introduction

The purpose, background, procedure and evaluation as well as interpretation of results of bottleneck identification for retrofit serial CMs are described.

C.3.1.1.1 Purpose

The purpose of this test is to identify whether a bottleneck exists in the cryptographic system.

C.3.1.1.2 Background

Annex H of AGA 12 offers the following background on bottleneck identification:

> The addition of cryptographic protection has the potential for creating a bottleneck in real-time process control communications. To determine whether a bottleneck may exist, refer to the manufacturer's specification of the maximum sustained throughput for each component through which the data must travel. It may be necessary to convert or interpret the specifications to derive a common measure (such as bits per second) for all of the components. If the full data stream must pass through a CM and it has the lowest rated throughput, it represents a theoretical bottleneck.
>
> In reality, a component is only a bottleneck if it impedes operation. A component may be capable of operating at a peak rate adequate to meet the requirements of the real-time process control system, or the real-time process control system may not exercise the full system throughput capability. For the cryptographic system, operating the real-time process control system under worst-case conditions both with and without the cryptographic system, and comparing the results can determine this.
>
> In some cases, when the cryptographic system creates a bottleneck, the problem can be alleviated using configuration options. For example, the interface between a CM and its associated computer (the plaintext port) may be capable of operating at a substantially higher speed than the communication link (on the ciphertext port). This type of asymmetric operation can dramatically reduce delays associated with filling and emptying the CM buffers.

In most cases, the CM will have the lowest rated throughput and therefore a theoretical bottleneck will exist in most cryptographic systems. The header, trailer, a cleartext to ciphertext ratio lower than 1 and encryption/decryption times are all factors that lead to a bottleneck.

C.3.1.1.3 Test Equipment

- PC
- GTI/NIST software tool or any similar tools
- Two identical CMs to test
- Null Modem (if necessary)
- Serial Cable
- Real-time process control host and slave device
- 2 Modems
- Line analyzer box

C.3.1.1.4 Procedure

C.3.1.1.4.1 General Procedure

Continuously send small messages to the cleartext port of the CM with as little delay between messages as possible.

Check to see if the CM crashes or fails eventually.

If the CM crashes, a theoretical bottleneck can be concluded.

Repeat test for a realistic field test configuration to ensure that the CM is indeed the theoretical bottleneck amongst all the equipment in the real-time process control network.

C.3.1.1.4.2 Detailed Procedure

A bottleneck can be easily identified by testing the throughput of the cryptographic system under peak load (see section C.3.3.1.4.1.2). The following steps discuss this procedure in detail:

Setup the equipment as shown in Configuration 3.

Run the GTI/NIST software tool. For more information on this tool refer to documentation associated with the software. Configure the settings for the program appropriately.

Pick the smallest realistic message length as well as the time to run the test.

The delay between messages should be the delay needed by the CM to determine the end of message.

Start the sending the small message continuously to the CM. If the CM crashes or fails, it is clear that the CM is a bottleneck. (To be sure that it is indeed the CM, it can be checked that the test runs successfully for Configuration 0).

The tester may repeat this test for the field test configuration using an actual message, to verify that the above conclusion holds for that particular field test condition.

To determine if the CM impedes operation, the tester should conduct the operational reliability tests discussed in C.3.3.1.4.2.1 and C.3.3.1.4.3.1.

C.3.1.1.5 Test Example

GTI conducted the bottleneck identification test on the AGA 12 prototype retrofit cryptographic modules. The prototype CMs ran the ScadaSafe implementation of the AGA 12, Part 2 protocol on the Arcom Viper® development boards. The software developed by GTI was used to help perform the test. The results are provided below as an example.

C.3.1.1.5.1 Results

Test Name	Test Type
Bottleneck Identification	(Bench / ~~Testbed~~ / ~~Field~~)

Date: 03/06/05
Company/Organization: Gas Technology Institute
Test Performed by: Aakash Shah

Cryptographic Module information

ScadaSafe prototype running on an Arcom Viper M64 – F32 board.
ScadaSafe Version: 0.6.7

Cryptographic Protocol information
Algorithm used: AES
Key length: 128
Authentication: 4 byte HMAC
Mode: AGA 12 – Part 2 Cipher Suite 2 (PE Mode)

Test Configuration

- Communications protocols used on various links
  Modbus RTU
- Make, model and/or type of the different kinds of communications equipment
  N/A

- Use of custom connectors and/or Null modems
  - Null modem & custom connector between Host PC and CM
- Host controller OS and real-time process control host software
  - Win2k, GTI test software
- Communications parameters:
  - Baud rate: 9600
  - Data bits: 8
  - Stop bits: 2
  - Parity bits: 0
  - Full/half duplex: half duplex
  - Flow control: None
  - Are communications parameters negotiated or ever changed? N
- Any specific operating modes
  - None
- Polling scheme (master poll, report-by-exception, etc.), timeouts, and polling rates (e.g. poll every second, as fast as possible, etc.)
  - Unidirectional Poll | 200 ms host timeout | as fast as possible

- Types of polls and responses and average lengths. If the set of polls is recurring, record the poll/response lengths. If possible note the slave processing time for each poll.
  - 8 byte test message
- Note all relevant parameters used for the GTI software tool.
  - Timeout: 200 ms
  - Time between consecutive poll/resp cycles: 4 ms
  - Test Time: ~3 hours
- Reliability test configuration and results
  - Communications protocol used
    - Modbus RTU
  - Slave device information
    - Make, model and/or type
      - Acromag 913MB I/O Module
  - Field test configuration (for C.3.3.1.4.3.1)
    - Field test not conducted
  - Was bottleneck detected under the stressed reliability test (Y/N)?
    - No
  - Did CM fail for either reliability test (Y/N):
    - No

Bottleneck Test Results

CM failure (y/n)

No

Notes:

The CM buffer was large enough to successfully pass through continuous unidirectional 4 byte messages over a 3 hour period. While the throughput of the system may have decreased considerably, the CM did not experience a failure. However, during this period the windows PC crashed, probably due to an overloaded serial port buffer.

C.3.1.1.6 Interpretation of Results

As mentioned before, realistically, a component is only a bottleneck if it impedes operation [1]. It is expected that the CM will be a theoretical bottleneck. But, if the host connected to the CM only sends a single message every day, the addition of cryptographic modules will not “impede operation”. On the other hand, if the CM fails under the operational reliability tests (C.3.3.1.4.2.1 and C.3.3.1.4.3.1) it may be a bottleneck.

C.3.1.1.7 Definitions

Bottleneck: A component of a control system network that limits performance on that network.

C.3.1.1.7 Test Results Sheet

Test Form

Test Name Bottleneck identification	Test subset (if applicable)

Date:
Company/Organization:
Test Performed By: Position:

Cryptographic Module information

Make: Model: Type:
Firmware Version:
Other information:

Cryptographic Protocol information

Note all relevant information regarding the CM's cryptographic protocol including:

- Encryption algorithm used:
  - Key length:
- Authentication algorithm
- Integrity method
- Mode of operation

Test Configuration

Note the following information:

- Use of custom connectors and/or Null modems
- Host controller information
  - Make, Model, Processor and Operating System
  - Real-time control system host software used
- Communication parameters on various links
  - Baud rate
  - Data bits
  - Stop bits
  - Parity bits
  - Full/half duplex
  - Flow control
  - Are communications parameters negotiated or ever changed?
- Any specific operating modes of the real-time control system
- All relevant parameters used for the GTI software tool.

    - Timeout
    - Time between consecutive messages
    - Software Version
- Reliability test configuration and results
    - Communications protocol used
    - Slave device information
        - Make, model and/or type
    - Field test configuration (for C.3.3.1.4.3.1)
    - Was bottleneck detected under the stressed reliability test (Y/N)?
    - Did CM fail for either operational reliability test (Y/N):
        - If yes - describe any relevant information regarding the CM failure:

Results

CM failure (y/n)

Notes

C.3.2 Regression Tests

C.3.2.1 Introduction

The purpose, background and procedure for regression tests of new cryptographic module versions are described.

C.3.2.1.1 Purpose

The purpose of this test is to run tests on a new cryptographic module version to ensure that it will not affect real-time control system once deployed.

C.3.2.1.2 Background

AGA 12 offers the following background on regression tests:

> Regression testing is not one test, but a series of tests that measure critical aspects of the CM under test. For each new release of CM software and hardware, regression testing ensures that the upgrade will function properly prior to deployment. A regression test plan identifies which new basic test objectives should be run against each new CM product release.
>
> CM regression testing can verify that a hardware or software upgrade does not impact performance, reliability, or functionality of the cryptographic system. Regression testing does not measure new features or capabilities. Such tests fall under functional testing discussed in Clause H.3.1.2.
>
> Use test data from past regression test as a baseline for the current regression test. If no current data exists, first run test against the current cryptographic system before testing the upgrade. Without a baseline against which to compare the CM upgrade, it cannot be determined that the cryptographic system has been improved or regressed.

C.3.2.1.3 Equipment

- Tools to conduct reliability, throughput, latency, jitter, clock synchronization and noise tests
- Three identical CMs to test
- Null Modem (if necessary)
- Serial Cables
- Real-time process control host and slave devices

C.3.2.1.4 Procedure

Before upgrading the CMs to the version under test, conduct the following tests to gather baseline data:

- Effect of Message Content on Latency (see C.2.2.1)
- Throughput (see C.2.3.1)
- Reliability (see C.3.3.1)
- Effect of noise (see C.2.6.1)
- Clock Synchronization (see C.2.4.1)
- Jitter (see C.1.2.1)

Upgrade the three CMs to the version under test. Repeat the test procedures above. Optionally, the interoperability tests between CMs with new and old versions may be conducted if applicable.

Compare the results from both tests and ensure that results are satisfactory.

C.3.2.1.5 Test Results

<u>Test Results Form</u>

<u>Test Name</u> Regression Tests	<u>Test Type</u> (Bench / Testbed / Field)

Date:
Company/Organization:

Test Performed by:	Position:

Cryptographic Module information

Make:	Model:	Type:

Old Firmware version:
New Firmware version:

Results

Attach results from all the performance, reliability and functionality tests conducted.

Note any significant improvements / deterioration in performance.

Note any failures in the functionality and reliability tests.

C.3.3 Reliability Tests

C.3.3.1 Introduction

The purpose, background, procedure as well as interpretation of results of reliability tests for retrofit serial cryptographic modules are described.

C.3.3.1.1 Purpose

The purpose of this test is to assess the reliability of the cryptographic modules used to secure a real-time process control system network.

C.3.3.1.2 Background

The following background on reliability testing is reproduced from AGA 12, Annex H:

> Reliability testing forces the CM, or the cryptographic system, under test to handle in a compressed time the activity it would normally experience over weeks, months, or years in operation. The testing may use accelerated loading techniques to apply and maintain high load on the CM for prolonged periods of time (30 hours or more). Reliability testing attempts to accelerate failure of the CM or cryptographic system caused by:
>
> > Cumulative errors: These are the result of repeating an operation multiple times in a fashion that results in an error.
> >
> > Timing errors: These errors are caused by two time-dependent operations that occur out of sequence or without proper delay.
> >
> > Statistical errors: It is virtually impossible to test and verify every possible path through the CM's code. However, statistically, over time, every path will be traversed, either because of an error condition or a seldom-invoked sequence of events. Reliability testing increases the probability that a statistical error will occur.
>
> Cryptographic system reliability testing measures how well the CMs maintain operation under various loads and feature configurations.
>
> Reliability testing provides three key measurements:
>
> > Operational reliability: Cryptographic system operation under maximum sustained load.
> >
> > Stressed reliability: Cryptographic system operation under peak load.
> >
> > Reliable recovery: Time to reestablish normal cryptographic system operation after non-fatal faults (e.g., adverse environmental conditions or loss of power supply).

The first measurement determines how reliable the cryptographic system is under a sustainable load where virtually all received messages are correctly forwarded to the destination. The second measurement determines how stable the cryptographic system is under peak loads. Operational reliability requires the cryptographic system to be stable for a long time at medium to heavy loads. Under stressed reliability, cryptographic systems can almost always be forced to fail; it is the mode of failure and recovery (e.g., fail-safe) that are important. Typical results could show:

• The cryptographic system cannot maintain the sustained load for long periods.

• The cryptographic system can maintain sustained loads, but fails under peak loads.

• The cryptographic system can maintain both sustained and peak loads.

• The cryptographic system encounters non-critical or recoverable errors under one or both loads.

• The cryptographic system encounters fatal errors under sustained loads.

C.3.3.1.3 Test Equipment

The following equipment is need for this test:

- PC
- GTI/NIST software tool or any similar tool(s)
- Three identical CMs to test
- Null Modem (if necessary)
- Serial Cable
- Real-time process control system host and slave device
- Breakout box

C.3.3.1.4 Procedure

This test has three steps to it:

- Lab test
- Test bed evaluation
- Field test

The general procedure to test the reliability of a cryptographic system is as follows:

- Determine the maximum sustainable load as well as the peak load for the CMs.
- Test the CMs for a prolonged period of time (30 hours or more) under the maximum sustainable load and then under the peak load. Document all instances of CM failure.
- If the cryptographic system fails, note the mode of failure and recovery.
- Test the cryptographic module under the maximum sustainable load. While the cryptographic system is under operation turn off the power. Turn the power back on and ensure reliable recovery. Also, note the time to reestablish normal cryptographic operation.

GTI has developed a software tool to run the lab as well as test bed portion of the reliability tests on retrofit serial cryptographic modules. More information on this software is available in the related README file.

C.3.3.1.4.1 Lab test

Setup the test equipment as shown in Configuration 3. Download and run the throughput.exe test program.
Pick a realistic message length to test.

C.3.3.1.4.1.1 Operational reliability test:

- Determine the minimum delay needed between continuous messages of a particular length to reach the maximum sustainable load for the CMs. The maximum sustainable load is just the maximum throughput of the system. The manufacturer's documentation should have information regarding the minimum delay required between messages and/or the maximum sustainable load (the minimum delay can be calculated from this value). If this information is not available, the delay can be roughly determined by trial and error.
    1. Pick a test message with a length that is one byte more than the payload of the CM.
    2. Continuously transmit this test message through the cryptographic system. If the CM concludes the end of message by a set timeout, the delay between these messages should at least be the timeout value.
    3. If the CM fails or crashes, increase the delay between messages and run the above step again.
    4. Once sustained functioning is achieved, note the delay used. This is the delay that will be used in the following lab tests.

- Pick a message to test. Input this message to the GTI software. Optionally, it may be easier use a test message generated by the software to test the CM.
- Run the throughput test for a prolonged period of time (30 hours or more).
- If the CM fails or crashes, note the delay used between messages as well as the number of messages communicated successfully prior to the failure.
- Note the process of system recovery after failure.

C.3.3.1.4.1.2 Stressed reliability test

- For the peak load, the delay between the messages should just be the silence required by the CM (if any) to denote the end of message.
- Run the throughput test for a prolonged period of time (30 hours or more) under the peak load.
- Under peak loads the CM can almost always be forced to fail. Note the number of messages communicated successfully prior to CM failure.
- Note the process of system recovery after CM failure.

C.3.3.1.4.1.3 Reliable recovery test

- Test the cryptographic module under the maximum sustainable load.
- While the cryptographic system is under operation turn off the power.
- Turn the power back on and ensure reliable recovery. Also, note the time it takes to reestablish normal cryptographic operation.

C.3.3.1.4.2 Test bed evaluation

The test-bed evaluation incorporates the slave devices into the lab test. On the host side the user has the choice to employ an actual real-time process control host or emulate the host by running the GTI software tools on a PC.

The tester should run this test for at least the following three topological configurations:
Point to Point
Point to Multipoint (See Configuration 7)
Point to Multipoint in mixed-mode (See Configuration 8)
The tester may add other configurations based on his/her need.

To use the GTI software tools, valid polls as well as corresponding responses must be known or obtained by using the line analyzer box (see Configuration 11). The procedure below utilizes a PC along with the GTI software tools to emulate the real-time process control host and measure throughput.

If an actual real-time process control host is used, the same message(s) (for example, a continuous poll) should be transmitted repeatedly. This way the message(s) will be stored in the real-time process control host's cache, and will be retrieved speedily. The number of messages communicated successfully should be recorded.

Setup the test equipment as shown in Configuration 5 for point to point, Configuration 7 for point to multi-point and Configuration 8 for point to multi-point in mixed-mode.

C.3.3.1.4.2.1 Operational Reliability test
- Pick a valid message to test. Input this message to the GTI software.
- Run the throughput test for a prolonged period of time (30 hours or more). The delay between the messages should just be the delay needed by the CMs to denote silence.
- If the CM fails or crashes, note the delay used between messages as well as the number of messages communicated successfully prior to the failure.
- Note the process of system recovery after failure.

C.3.3.1.4.1.3 Reliable recovery test

- Test the cryptographic module under the maximum sustainable load.
- While the cryptographic system is under operation turn off the power.

- Turn the power back on and ensure reliable recovery. Note the time to reestablish normal cryptographic operation.

C.3.3.1.4.3 Field Test

The field test configuration should be documented thoroughly. To get the best results, the same message(s) should be transmitted repeatedly. This can be done by continuously polling the slave(s). The length(s) of the message(s) being repeated as well as the corresponding response(s) should be obtained by using the line analyzer box (see Configuration 11). The number of messages communicated successfully should be recorded.

The tester should run this test for at least the following three topological configurations:

Point to Point
Point to Multipoint
Point to Multipoint in mixed-mode

C.3.3.1.4.3.1 Operational Reliability test

- Continuously poll the slave.
- If the CM fails or crashes, record all relevant information possible.
- Note the process of system recovery after failure.

C.3.3.1.4.3.2 Reliable recovery test

The process for the reliable recovery test is the same as described in C.3.3.1.4.1.3

C.3.3.1.5 Test Example

GTI conducted the reliability tests on the AGA 12 prototype retrofit cryptographic modules. The prototype CMs ran the ScadaSafe implementation of the AGA 12, Part 2 protocol on the Arcom Viper® development boards. The software developed by GTI was used to help perform the test. The results are provided below as an example.

Test Name	Test Type
Reliability tests	(~~Bench~~ / Testbed / ~~Field~~)

Date: 10/28/04
Company/Organization: Gas Technology Institute
Test Performed by: Aakash Shah

Cryptographic Module information

ScadaSafe prototype running on an Arcom Viper M64 – F32 board.
ScadaSafe Version: 0.6.7

Cryptographic Protocol information

Algorithm used: AES

Key length: 128

Authentication: 4 byte HMAC

Mode: AGA 12 – Part 2 Cipher Suite 2 (PE Mode)

Divide the real-time process control network being tested into logical sections based on the communications mediums and speeds. Complete this test sheet for each one of these sections.

Test Configuration

Draw a detailed communications system diagram depicting the serial real-time process control network topology along with any LAN connections.

Note the following information:

- Approximate geographic distances between components.
  On Bench
- Communications protocols used on various links
  Modbus RTU
- Make, model and/or type of the different kinds of communications equipment
  Acromag 913 MB I/O modules
- Use of custom connectors and/or Null modems
  Null modem & custom connector between Host PC and CM
- Host controller OS and real-time process control host software
  Win2k, GTI test software
- Communications parameters: baud rates, start/stop/parity bits, full/half duplex, flow control, if negotiated, ever changed
  9600 8n2, half duplex, no flow control
- Any specific operating modes
  None
- Polling scheme (master poll, report-by-exception, etc.), timeouts, and polling rates (e.g. poll every second, as fast as possible, etc.)
  Master Poll | 1 sec host timeout | as fast as possible
- Types of polls and responses and average lengths. If the set of polls is recurring, record the poll/response lengths. If possible note the slave processing time for each poll.
  4 byte poll, 43 byte response | Avg. slave processing time 30 ms
  The poll queries the slave for its model number, serial number etc.
- Note all relevant parameters used for the GTI software tool.
  Timeout: 1 s
  Test Time: 30 hours

Operational Reliability Test

Note: Since the system performed properly under stressed conditions the same results are presented for the operational reliability tests.

Number of messages communicated correctly under operational reliability test: 1675870
Number of errors in message transmission: 23
Time between consecutive poll/resp cycles: 4 ms
CM failure (Y/N): N

Stressed Reliability Test

Number of messages communicated correctly under stressed reliability test: 1675870
Number of errors in message transmission: 23
Time between consecutive poll/resp cycles: 4 ms
CM failure (Y/N): N

Reliable recovery test

Was CM able to recover after power failure (Y/N): Y
Time it took for CM to recover: 60 s
Describe the process used to get the CM functioning normally again: Since the prototype CM runs on Arcom Embedded Linux (*not* a real time operating system), the recovery process was essentially the time it took for the embedded board to reboot and restart the operating system and then run the ScadaSafe implementation.

C.3.3.1.6 Interpretation of Results

The main goal of these tests is to ensure the reliability of the CMs. The CMs should pass all the operational reliability tests. The number of messages transmitted successfully in the operational reliability tests is the throughput for the test period. The operational reliability tests as described above only test a single message or a sequence of messages. Different message lengths should be tested while conducting the throughput tests to ensure functionality.

As stated earlier, it is expected that the CMs will fail under the stressed reliability test. But knowing how many consecutive messages lead to a CM failure is important in gauging CM performance.

C.3.3.1.7 Definitions

Operational reliability test: This test evaluates cryptographic system operation under maximum sustained load

Stressed reliability test: This test evaluates cryptographic system operation under peak load.

C.3.3.1.8 Test Results Sheet

<u>Test Form</u>

<u>Test Name</u> Reliability Tests	<u>Test Type</u> (Bench / Testbed / Field)

Date:
Company/Organization:
Test Performed By: Position:

Cryptographic Module information

Make: Model: Type:
Firmware Version:
Other information:

Cryptographic Protocol information

Note all relevant information regarding the CM's cryptographic protocol including:

- Encryption algorithm used:
    - Key length:
- Authentication algorithm
- Integrity method
- Mode of operation

Test Configuration

Draw a detailed communications system diagram depicting the serial real-time process control network topology along with any LAN connections.

Note the following information:
- Approximate geographic distances between components
- Communications protocols used on various links
- Make, model and/or type of the different kinds of communications equipment
- Use of custom connectors and/or Null modems
- Host controller information
    - o Make, Model, Processor and Operating System
    - o Real-time control system host software used
- Slave device information (if applicable)
    - o Make, model and/or type
- Communication parameters on various links
    - o Baud rate
    - o Data bits
    - o Stop bits

    - Parity bits
    - Full/half duplex
    - Flow control
    - Are communications parameters negotiated or ever changed?
- Any specific operating modes of the real-time control system
- Polling scheme (master poll, report-by-exception, etc.), timeouts, and polling rates (e.g. poll every second, as fast as possible, etc.)
- Types of polls and responses and average lengths. If the set of polls is recurring, record the poll/response lengths. If possible note the slave processing time for each poll
- All relevant parameters used for the GTI software tool
    - Timeout
    - Test Time
    - Time between consecutive poll / response cycles
    - Message(s) repeated
    - Test Time
    - Typical response size (in bytes)
    - Software version
    - Other

Results

Operational Reliability Test

Number of messages communicated correctly under operational reliability test:
Number of errors in message transmission:
Host Software configuration:
CM failure (Y/N):
If *yes*,

Number of times test run:
Number of failures:
Describe the recovery process if failure or crash in each case and note the number of messages communicated correctly before each failure:

Stressed Reliability Test

Number of messages communicated correctly under stressed reliability test:
Number of errors in message transmission:
Host Software configuration:
CM failure (Y/N):
Describe the recovery process if failure:

Reliable recovery test

Was CM able to recover after power failure (Y/N):

Time it took for CM to recover:
Describe the process used to get the CM functioning normally again:

Notes

Appendix D AGA 12, Part 2 RFC

AGA
American Gas Association

Cryptographic Protection of SCADA Communications

Retrofitting Serial Communications

AGA Report No. 12-1

Request For Comments

This document presents the issues that the members of the AGA 12 Task Group identified as pertinent to the design and development of cryptographic protection mechanisms for the exposed serial communications that are part of a SCADA network. Its publication is with the intent that people will comment on the general background described herein, which are the underlying principles directing the design of the retrofit cryptographic modules.

The primary considerations in evaluating this RFC are its completeness and correctness. Specifically, we would like feedback on whether there are any other considerations that should be taken into account in the continuing development of the specification for retrofit cryptographic modules.

The details for retrofitting serial communications links with cryptographic protection, such as the hardware design, the general functionality, and the description of the communications protocols and key management procedures, will be finalized based in part on your submitted comments.

Please submit your comments to:

John A. Kinast
Gas Technology Institute
1700 S Mount Prospect Rd
Des Plaines IL 60018
847-768-0555
john.kinast@gastechnology.org

Also contact John should you have any questions about this RFC.

Thank you for your assistance tin ensuring that the eventual design meets the widest needs of the natural gas, electric, water, waste-water, and related industries.

AGA 12 Task Group

SCADA Serial Line Operational Constraints

The following is a description of the general SCADA equipment characteristics and operating environment for transferring SCADA messages over asynchronous serial links as described in AGA 12 Annex C. When SCADA communications are to be cryptographically protected the equipment characteristics and the operating environment combine to produce constraints of which system designers and operators must be aware.

This document is structured to be a list of the environmental realities (identified below by a letter and number in the form E*n*, where *n* is an integer designated to identify the particular environmental or equipment characteristic. Following each characteristic, the corresponding constraint(s) is(are) listed; these constraints are identified with an index number of the form C*n*.*m*, where the C indicates a constraint, *n* is an index that is the same as the index E*n* corresponding to an environmental characteristic, and *m* is an index to distinguish different constraints arising from environmental characteristic E*n*. Thus, each environmental characteristic E*n* has one or more constraint implications, C*n*.1, C*n*.2, etc. The acronym SCM designates a SCADA Cryptographic Module, the component of the system that provides cryptographic protection to the communication channel used by the SCADA system.

Terminology

The following terms have specific meaning within the context of the this document.

Shall: the word ***shall***, equivalent to "is required to," is used to indicate a mandatory requirement.

Should: the word ***should***, equivalent to "is recommended that," is used to indicate that a certain course of action is preferred, or a particular function is desired, but not required.

NOTE: As in a previous version, the numbering has not been updated to be able to identify where individual items originated. The numbering will be updated when finalized.

General

E1: SCADA system operators want to deny unauthorized access to their system, including access as a result of malicious or inadvertent insider operations.

C1.1: Without existing methods, a new method needs to be developed to protect SCADA systems from unauthorized uses. This includes providing confidentiality, integrity, and authentication.

C1.2: A system for managing the SCADA protection mechanisms should address protecting data-in-transit, data-at-rest, and authorized operators.

C1.3: The system for managing the SCADA protection mechanisms should be designed with an operating model of "separation of duties", i.e., no one individual shall have total control over the management system.

C1.4: The system for managing the SCADA protection mechanisms should be designed as a single system or interoperable modules to address all forms of SCADA protection, including configuration of cryptographic modules; provisioning cryptographic material for cryptographic modules; monitoring forensic and usage information from cryptographic modules; managing operators, their authorized roles and privileges; managing operator two-factor authentication hardware tokens; provisioning cryptographic materials for operators and their two-factor authentication hardware tokens; monitoring usage and actions taken by operators; and creating and managing cryptographic materials for protection of data-at-rest.

Business & Operational Issues

E2: Utilities have significant investment in SCADA equipment, in terms of capital investment (hardware and software) and training of operational personnel (SCADA operators, engineers, maintenance technicians, etc.).

C2.1: Protection methods shall not be developed that require major replacement of existing equipment or software/hardware modifications requiring the re-certification of the existing SCADA equipment in the near term.

C2.2: Integration of protection methods shall not be prohibitive in terms of cost, complexity, or operations.

E3: Operating under time & funding limitations, utility SCADA operators would like the option of selecting from multiple vendors equipment and techniques to protect their systems that are low cost, and easy to install and operate.

C3.1: SCADA protection products shall have a minimum level of interoperability, both between different products by the same manufacturer and products made by different manufacturers.

C3.2: Management systems of the SCADA protection mechanism should be designed to leverage and interoperate with existing security management systems, including directory structures, key management systems, and intrusion detections systems.

E4: Utility communications (including SCADA, EMS, DMS, and DER) can occur between multiple entities which interact for business and operational purposes, including market trading and ISO, RTO, or DER Aggregator relationships.

C4.1: Protection mechanisms shall support at least limited interoperability, both between different products by the same manufacturer, and products made by different manufacturers.

C4.2: Protection mechanism should address multi-domain issues, such as designated roles and responsibilities including owners, operators, field maintenance technicians, control center management/provisioning, data access privileges, etc., and interoperable credentials to allow access without creating vulnerabilities.

Equipment

E5: SCADA and similar control equipment are designed to have significant lifetimes.

C5.1: Protection methods shall not be developed that assume that replacement of SCADA equipment will happen in the near term, e.g., that new equipment with additional capabilities will replace existing equipment any time soon.

C5.2: Protection methods should be designed to provide for compatibility between near-term and long-term solutions.

E6: The processors traditionally used in remote SCADA units (e.g., RTUs, IEDs) have limited capabilities and low processor speeds compared to current desktop computers, and have little additional memory for programmable functionality.

C6.1: Because testing with typical hardware (such as the effort in a previous GTI project) has shown, in general, that adequate cryptographic functionality cannot be provided by the limited capabilities of legacy remote SCADA processors, protection methods should be implemented as a retrofitted add-on device.

C6.2: Existing SCADA systems should be able to operate without knowledge of any add-on protecting devices.

C6.3: Control communication with external devices shall not be interrupted, e.g., modem commands should be detected and sent "in-the-clear".

E7: SCADA communications occur over wired lines and wireless channels that may have various levels of protection afforded by existing measures. Some are protected within buildings and fences, some are exposed through/by third parties such as the telephone company, while others are relatively unprotected.

C7.1: Potential attackers can access exposed communications links. Such attacks include listening to or altering SCADA messages, generating new SCADA messages, or blocking the flow of data.

E8: SCADA systems operate in harsh environmental conditions.

C8.1: Protection mechanisms should operate in traditional SCADA environments, including both substation environments and control center environments.

E9: SCADA equipment is typically housed in substations, protective enclosures, or control center equipment rooms.

C9.1: Protection mechanisms should be designed, at a minimum, to exhibit evidence if the mechanism is tampered with.

C9.2: Protection mechanisms may rely upon physical protection (i.e., limited operator access) for some of its security requirements.

Channel topology

E10: The majority of remote SCADA communication is over leased lines, dial-up connections, or radio links.

C10.1: The communication links to field devices (e.g., asynchronous serial) do not provide either virtual or physical concurrent connections to multiple entities. If an SCM is placed on a particular communication link (e.g., in-line), the SCM cannot contact another entity to obtain information or services (e.g., to acquire an encryption key or to verify the status of a partner's credentials).

E11: In some cases, secondary, or backup, communication links are present, e.g., dial-up connections are often used as backup for leased lines.

C11.1: Alternative connections may require different types of SCADA message protection and may require protection equipment to be able to differentiate between the primary and secondary channels.

E12: Many SCADA networks are designed for the possibility of multiple SCADA hosts that communicate with SCADA field units, e.g., primary and backup center.

C12.1: Protection mechanism management should be designed to operate in one or more control centers, for disaster recovery and distributed management purposes.

C12.2: Protection methods shall not interfere with the ability of a utility to switch between the primary and the backup SCADA host(s), e.g., support for "cold" backup control centers (normally off, and must be started before use).

C12.3: SCMs should support multiple addresses so backup control centers can monitor traffic between the primary control center and field devices, e.g., support for "hot" (actively monitoring or aware of all message traffic) or "warm" (active but not monitoring message traffic) backup control centers.

C12.4: Protection mechanism should support both local and remote (in-band) management (including configuration, forensics, and key management).

Channel characteristics

E13: Most communication is low speed serial (300-2400 bps, with newer units using 9600 and 19,200 bps), and uses either byte- or bit-oriented protocols, with a trend toward byte-oriented in new equipment.

C13.1: Little additional bandwidth is available in which to perform functions related to the cryptographic protection (e.g., session establishment).

C13.2: Protection mechanisms shall support both hardware and software flow control to minimize the potential for data loss.

E14: In many instances, SCADA message times are short compared to the time to open and close communication channels. Many SCADA operators have stated that they thought higher baud rates require more time for modems to negotiate than the entire time it takes to issue a SCADA command and receive a reply using modems operating at the lower baud rates.

C14.1: SCADA systems are designed for frequent (near real-time) status updates. Incorporation of cryptographic functions should not reduce the reading frequency of an existing system below an acceptable level (nominally 20%).

E15: The serial connection ensures that the order of messages received matches the order of the sent messages. There is no packet re-ordering or multiple delivery path issue as may be present on packet networks.

C15.1: Any protection method shall preserve message ordering to accommodate the existing equipment and software.

E16: SCADA units can communicate over shared or unshared channels, i.e., a SCADA host may use a particular channel to communicate with one or more field units.

C16.1: Protection methods shall be selected/designed to be able to properly direct protected SCADA messages to the appropriate SCMs, for those instances in which the exposed channel is shared amongst multiple SCADA units (also described as a multi-drop, multi-point, or daisy-chain configuration). Note: SCADA units connected by radio communications can be seen as operating in this fashion.

C16.2: SCMs shall be able to associate multiple SCADA units with target SCMs, for those instances in which only one exposed channel connects multiple SCADA units with the host.

E17: SCADA units on a shared channel may have SCMs installed on only some of the RTUs as needed (determined by priority/security), the SCMs may be gradually deployed in a staged fashion, or some may be out-of-service for maintenance.

C17.1: Protected equipment may need to share a communications channel with unprotected equipment, without causing interference between protected and unprotected equipment.

C17.2: Protection methods need to be able to differentiate between SCADA units that are protected and those that are not, and send the appropriate encrypted or unencrypted message.

C17.3: Protection methods need to be designed so that protected SCADA messages cannot be inadvertently recognized as unprotected SCADA messages.

E18: SCADA units on a shared channel (e.g., daisy-chained) may have multiple segments of the line externally exposed between field units, requiring protection mechanisms at each connection point along the line.

C18.1 Protection mechanisms shall be designed to permit recognition of SCADA addresses to properly identify SCADA units, and to associate SCMs with SCADA units to direct the protected messages.

E19: Communications may be implemented as store-and-forward, where one SCADA unit receives a message as an intermediary, then re-transmits it to the intended unit.

C19.1: Protection methods need to be designed to not interfere with the store-and-forward function as implemented by some SCADA protocols.

SCADA messages

E20: Short message lengths are common for many commands and responses; most messages are in the range of 16-32 bytes. Almost all messages are less than 256 bytes.

C20.1: Implementation of methods to protect messages needs to ensure that transmission is not significantly delayed by the additional information required for protection, e.g., adding 32 bytes or more of overhead to an 8 byte message should be avoided.

E21: Responses to queries may be as short as the query, or they may be significantly longer. In some instances a single query can prompt a response of multiple messages (e.g., a request for historic information or range of data).

C21.1: SCADA message protection shall not be based the assumption of 1:1 message flows, i.e., one response for one query.

SCADA error handling

E22: Noise can interfere with integrity of messages (especially on radio links). Typically integrity is checked using a checksum or CRC as part of the SCADA message.

C22.1: Protection methods shall be designed to not interfere with existing SCADA protocol integrity checks.

C22.2: Protection methods should not be potentially weakened by the presence of SCADA integrity checks.

E23: SCADA systems are designed to handle damaged or un-received messages due to noise and other interference with timeouts and retries in the SCADA application.

C23.1: Protection methods should not interfere with normal problem identification and handling by the SCADA application, e.g., implementing secure transport involving retries may interfere with the natural SCADA retrial.

C23.2 Protection methods should not add additional traffic on the line for error handling that may interfere with the normal operation (i.e., poll cycle or scan rate) of the SCADA system

SCADA protocols

E24: There are estimated to be between 100 and 250 SCADA communications protocols in the field.

C24.1: Independence of the SCADA message format is desirable. Incorporation of a method to recognize (parse) all or most SCADA protocols is not considered necessary for protecting the SCADA messages. Protection methods that require minimal or no understanding of the SCADA protocol are desired.

E25: SCADA protocols use various methods to identify messages and message lengths. This can include message length values embedded in the SCADA message, special values used as markers (e.g., End-Of-Text), and/or timing values (e.g., period of silence indicates message end).

C25.1: Protection methods need to be able to handle, at a minimum, the three identified methods of recognizing SCADA start-of-message and end-of-message to be able to properly encapsulate messages.

C25.2: Overhead latency should be kept to a minimum so that time-based end-of-message detection is not adversely impacted (i.e., a long SCADA message broken into multiple parts by the SCM is decrypted and reassembled in a reasonable amount of time such that the SCADA protocol is not fooled into believing it has received multiple messages, rendering the original SCADA message unusable).

E26: Multiple SCADA protocols can be transmitted on the same channel. The differences between protocols permit units to recognize their own messages while ignoring the others with the different protocol, usually treating them as noise.

C26.1: Protection methods need to be designed so that protected SCADA messages are not inadvertently recognized as any of the existing SCADA protocols on a channel.

C26.2: Protection methods may need to be configurable to differentiate between (multiple) SCADA protocols that exist on the same channel.

E27: Most SCADA messages are repetitive (e.g., status requests) or are very similar (e.g., status responses).

C27.1: Protection methods shall to be designed to recognize that few unique messages are sent. If a unit sends the same (unprotected) message repeatedly, the protected form of the message shall be different.

C27.2 Protection mechanisms shall protect against message replay.

E28: Although most SCADA systems operate on the poll-response model, some SCADA field units can initiate a message to the host (i.e., report by exception).

C28.1: Protection methods should not interfere with the basic operation of each SCADA unit to initiate messages, if the SCADA protocol permits such function.

C28.2: SCMs at either end of the link should be able to initiate (request) a cryptographic session.

C28.3 Protection mechanisms shall protect against message spoofing.

SCADA broadcast/multicast capabilities

E29: Some SCADA protocols provide for a broadcast (all units) or multicast (range of units) address.

C29.1: Protection methods should be able to accommodate broadcast or multicast transmission to multiple SCADA field units. (Note: SCADA protocols that do not implement this feature natively will not have it provided by the SCMs.)

C29.2: Broadcast/multicast messages that are encrypted should be intelligible to all protected SCADA units on the SCADA system that are intended to receive them without the necessity of sending the broad/multi cast message to each unit separately.

C29.3: Broadcast/multicast messages shall be sent in the clear to unprotected SCADA units.

C29.4: SCMs should support multiple broadcast addresses or subscription groups.

C29.5: Protection methods should be designed to minimize the timing differences in the availability of the SCADA message between encrypted broadcast messages and unencrypted broadcast messages; important in "set time" functions.

C29.6: Protection methods should be designed recognizing that sending both plaintext and ciphertext of the same SCADA message makes analysis to derive the encryption key "easier". For example, separate keys should be used for broadcast message versus unicast message, and broadcast and multicast keys should be changed frequently.

SCADA field device maintenance ports

E30: Many SCADA field devices support dial-up maintenance ports, which permit remote access for authorized technicians to modify settings, programming, etc., as required for support functions.

C30.1: Protection mechanism shall recognize there may be a potentially large number of technicians authorized to perform maintenance on field devices.

C30.2: Protection mechanism shall identify technicians uniquely due to the potentially transient nature of technicians (e.g., temporary employees, contractors, vendors, etc), and for auditing purposes.

C30.3: Protection mechanism shall recognize the technicians may be authorized to perform maintenance on specific devices, or perform maintenance for a limited period of time.

C30.4: Protection mechanism may not be able to establish a concurrent connection to a centralized host to offload authentication and authorization processes, i.e., authentication and authorization may need to be performed locally by the protection mechanism.

E31: SCADA maintenance port interaction is defined by the speed of the technician, instead of the real-time needs specified for the SCADA host.

C31.1: Communication timing periods can be relaxed in comparison to those for the primary control and reporting channel needs.

E32: Typical SCADA maintenance port interaction is via dialup mechanisms.

C32.1: Protection mechanism should provide support for multiple types of dialup topologies, including: field devices with external modems; field devices with internal modems; a single modem interacting with a single field device; and a single modem interacting with multiple field devices.

C32.2: Protection mechanism should not interfere with the ability of the field device to dial out, such as to report an alarm or alert condition.

C32.3: Protection mechanism should not significantly interfere with the field device’s existing maintenance mechanisms or procedures.