

DNP3Sec: Distributed Network Protocol Version 3 (DNP3) Security Framework

Abstract. Distributed Network Protocol Version 3 (DNP3) is an open and optimized protocol developed for the Supervisory Control and Data Acquisition (SCADA) Systems supporting the utilities industries. The DNP3 enables the Master Station to request data from Substations using pre-defined control function commands and Substations to respond by transmitting the requested data. DNP3 was never designed with security mechanisms in mind and therefore the protocol itself lacks any form of authentication or encryption. Discussion so far has been centered on two solutions to provide security for SCADA: cryptographic technologies placed at each end of the communication medium, or security enhancements placed directly in the protocol. This paper recommends a new Distributed Network Protocol Version 3 Security (DNP3Sec) framework to enable confidentiality, integrity, and authenticity placed directly in the DNP3. Such framework requires some modifications in the data structure of the DNP3 Data Link layer. Our main goal is to address the threats related to confidentiality, integrity, and authenticity in the DNP3 as part of SCADA architecture, with a minimum performance impact on the communication link; and without requiring modification to the much more expensive Master Station and Substation devices and the applications supporting them.

Keywords: SCADA, DNP3, DNP3Sec

1 Introduction

SCADA systems are used extensively throughout the utilities industries to monitor and control processes that are deployed in many sites. SCADA depends on many networks to support communications between its components, including microwave, PSTN, satellite, frame relay, wireless networks, and private fiber networks. The utilities have recently begun to leverage the Internet as a communication network to support SCADA systems. The Internet is a cost-effective communication tool to handle utilities application traffic as SCADA communication becomes more complex and requires more bandwidth. However, the Internet has opened SCADA, and the systems they support, to new vulnerabilities. The vulnerabilities of SCADA systems are widely recognized and the improving of their security has become an important priority for the utilities [3], [28], [23]. [31], [33], [34], [35]. For example, approximately one-third of electric power utility

control communications traffic is carried on public networks [12]. According to a Newton-Evans report, nearly 40% of the utilities responding either currently use or plan to use the Internet for SCADA [24].

Distributed Network Protocol Version 3 (DNP3) is used by SCADA systems to communicate between the Master host and the Slave units. This infrastructure is open, and effective authentication or encryption mechanisms does not exist. Although the utilities have increased their attention on improving the security and reliability of the SCADA systems in recent years, many owners and operators do not yet have the technology, tools, capabilities, and/or resources needed to secure their systems. Discussion so far has been centered on two solutions to provide security for SCADA: Encryption/decryption technologies placed at each end of the communication media, or security enhancements placed directly in the protocol [5]. To solve the problem of providing a solution placed at each end of the communication media, American Gas Association (AGA 12) [1] developed a standard for "Cryptographic Protection of SCADA Communications." Our paper recommends a new security framework to enable confidentiality, integrity, and authenticity directly in the DNP3. Such a framework requires some modifications in the data structure of the DNP Data Link layer.

The main goal behind our approach is to address the threats related to confidentiality, integrity, and authenticity in the SCADA Systems using the DNP3 with a minimum performance impact on the communication link, and without requiring modification to the much more expensive Master Station and Substation units and the applications supporting them. Our framework is built along the lines of the IPSec standards [16], [17], [18], [26], but has some unique features to maintain the specifications and the requirements of the DNP3 and SCADA architecture.

The remainder of the paper is arranged as follows. Section 2 summarizes related work. Section 3 describes the DNPsec framework by proposing security enhancements placed directly in DNP3. Section 4 concludes our paper. We added 4 appendices to the paper to help in providing background on SCADA and DNP. Appendix A provides a brief overview of SCADA systems. Appendix B provides a brief overview of DNP3. Appendix C provides threats analysis for SCADA systems using DNP3. And finally, Appendix D provides an analysis of our approach.

2 Related Work

Few publications are available on SCADA security, such as the American Gas Association Report No. 12 (AGA 12) [1]. AGA 12 recommends practices designed to protect SCADA's Master-Slave serial communication links from a variety of

active/passive cyber attacks. One of these standards is AGA 12-1, Cryptographic Protection of SCADA Communications. The solution protects against hijacking or modifying the communication channel.

AGA 12 requires the installation of multi-channel SCADA Cryptographic Modules (SCM) on a communications channel between the SCADA unit (e.g., host, RTU, IED) and the modem. A SCM receives and transmits SCADA messages on two communication ports: plaintext port and ciphertext port. The plaintext port is used to receive and transmit plaintext messages from a SCADA unit to a SCM, and the ciphertext port is used to transmit and receive ciphertext messages from a SCM to its peer.

SCM immediately begins transmitting a ciphertext message header to its peer as soon as it receives the first SCADA message characters. When it receives enough characters to fill its cipher block, it encrypts and transmits a block of ciphertext. When it finished transmitting all message blocks, it transmits a trailer that includes a Message Authentication Code (MAC).

At the receiving SCM, an incoming ciphertext message header signals the start of a new message. Each time enough characters are received on the ciphertext port to fill a cipher block, the SCM decrypts the block and immediately begins forwarding the decrypted characters via its plaintext port to the receiving SCADA unit. When the trailer of the ciphertext message is received, the SCM computes and checks the MAC. By this time, the decrypted SCADA message may have already been forwarded in its entirety to the receiving SCADA unit. If the authentication check fails, it is too late to prevent forwarding the unauthentic message. Thus the authentication code only alerts the SCM to a possible failure of data integrity [37]. Such solution is limited and expensive. The standard does not protect an attack from a compromised field site or control center. In addition, SCADA owners need to install AGA 12 compliance multi-channel SCADA Cryptographic Module (SCM) and Key Management Appliance in the SCADA Control Center; and SCM and Maintenance Cryptographic Module attached to every Remote Terminal Unit (RTU). Moreover, AGA 12 is still in the early stages from a system implementation standpoint. Key management is a key component of the standards and is still in the development stage.

In another research, Graham and Patel [25] examined three security enhancements in SCADA communications to reduce the vulnerability of cyber attacks to include: (1) solutions that wrap the DNP3 protocols without making changes to the protocols, (2) solutions that alter the DNP3 protocols fundamentally, and (3) enhancements to the DNP3 application. One of the research directions they identified is to secure the DNP3 protocol which is the focus of this paper. They provided high level description of possible solutions to protect SCADA communications and analysis for existing solutions such as DNP3 over IPSec or DNP over SSL/TLS. The main purpose of the paper is to identify the

possible solutions to secure SCADA messages for further research work to model and proof these solutions. The discussion about providing security for the DNP protocol is theoretical and describes the features of the proposed protocol at a very high level.

3 Supervisory Control And Data Acquisition (SCADA) Systems

See appendix A.

4 Distributed Network Protocol Version 3

See appendix B.

5 Threats Analysis for SCADA Systems

See appendix C.

6 Distributed Network Protocol Version 3 Security (DNPSec) Framework

DNPSec authentication and integrity framework capabilities will verify the frame origin, assure that the frame sent is the frame received, assure that the network headers have not changed since the frame was sent, and give anti-replay protection. DNPSec confidentiality framework capability will encrypt frames to protect against eavesdropping and hide frame source by applying encryption methods. Some modifications to the DNP3 LPDU or frame structure are required to provide these capabilities.

CRC is a common technique used in DNP3 for detecting data transmission errors. CRCs occupy 34 bytes out of 292 bytes of the DNP3 LPDU for integrity. These bytes will be utilized in a different way in the DNPSec framework.

There are two main components of the DNPSec. The first is the DNPSec structure to construct the frame and transfer data in secure mode between the Master and the Slave. The second is the key exchange established during the installation and connection setup between the Master and the Slave.

DNPSec structure

DNPSec consists of five fields (as shown in Figure 6-1) the new header, the key sequence number, the original LH header, the payload data, and the authentication data.

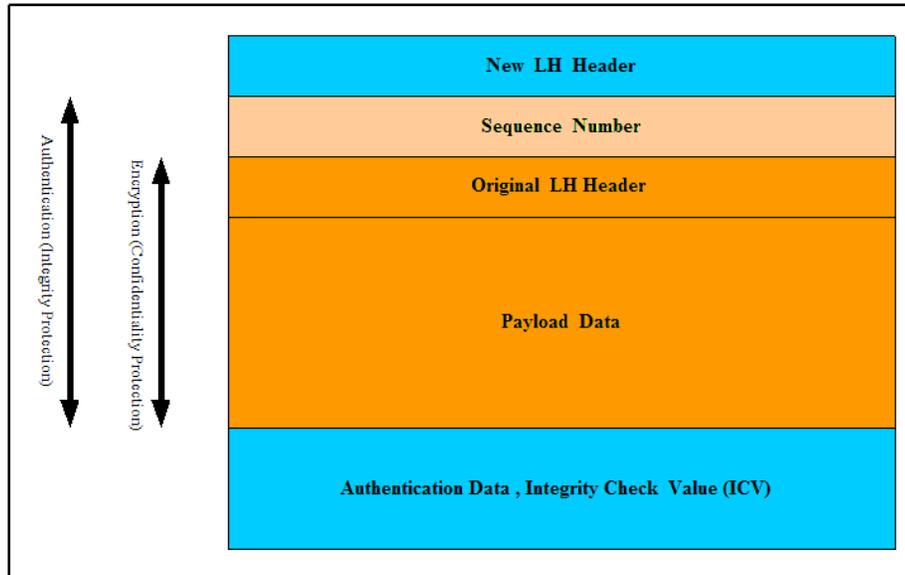


Figure 6-1: Protocol Structure - DNPSec

The **new header** is an unsigned 4 bytes field containing the destination address (DA) which occupies 2 bytes; the MH flag bit to recognize if the message is coming from the Primary Master host (0) or from the Secondary Master Host (1); the SK flag bit to indicate to the Slave if the message contains the new session key (1) or to decrypt the message using the session key in the S-keydb (0); and 14 bits reserved.

The **key sequence number** is an unsigned 4 bytes field containing a counter value that increases by one for each message sent by the Master. Each time a Master sends a message it increments the counter by one and places the value in the key sequence number field. Thus, the first value to be used is 1. The Master must not allow the key sequence number to cycle past $2^{32} - 1$ back to zero. If the limit of $2^{32} - 1$ is reached, the Master should terminate the session key and send a new key to the Slave using the value zero in the frame sequence number. This kind of functionality will guarantee that the Master and the Slave will continue establishing a new session key even if the connection is always open. Moreover, the security policy should indicate that a new session key must be established between the Master and the Slave in case the Slave used the same session

key for a certain time period. DNPsec uses the variable key-session-life-time to keep track of the life span of the session key.

The **original LH header** (DNP3 data link header without the 2 CRCs) and the payload data is protected by encryption and composed of 264 bytes field containing , 8 link protocol data unit header bytes, 250 Transport Protocol Data Unit bytes, and 6 padding dummy bytes. 264 bytes is a multiple of 4 bytes, which provides alignment of 4 bytes boundary and provides boundaries of 64 bits to support the encryption algorithms. For example, Data Encryption Standard (DES) specifies that the plaintext is 64 bits in length and the key is 56 bits in length. Longer plaintext are processed in 64-bit blocks.

The **authentication data** field containing an Integrity Check Value (ICV) computed over the key sequence number, the original LH header, and the payload data fields. ICV provides integrity services and is provided by a specific message authentication algorithm (MAC) such as, HMAC-MD5-96 or HMAC-SHA-1-96. The integrity algorithm specification must specify the length of the ICV and the comparison rules and processing steps for validation. DNPsec required 20 bytes for the authentication data field. To simplify the key management process, we are recommending to use the Master/Slave encryption/decryption session key to calculate the authentication data.

The DNPsec fields are as follows:

0 – 3	New Header (4 bytes)
DA: 0-1	Destination Address (2 bytes)
MH: 2(bit 0)	0: Primary Master Host, 1: Secondary Master Host
SK: 2(bit 1)	0: Fetch the database for the session key, 1: The frame contains a KSN value from the Master.
2(bits 2-7)-3	Reserved (2 bytes)
4 – 7	Key Sequence Number (4 bytes)
8 – 15	Original LH Header (8 bytes)
8 – 9	Sync (2 bytes)
10 - 10	Length (1 byte)
11 – 11	Link Control (1 byte)
12 – 13	Destination Address (2 bytes)
14 – 15	Source Address
16 – 271	Payload data (256 bytes)
16 – 265	TPDU data
266 – 271	Padding dummy data
272 – 291	Authentication Data (20 bytes)

Key management

The key management operations in DNPSec are very simple to accommodate the static nature of the SCADA environment. They occur during the configuration of the Primary Master host, the Secondary Master host, and the Slaves to establish the initial connection between them; after the re-initialization of the Key Sequence Number (KSN) to generate and distribute a new key to the hosts; and after the timeout of the usage of the session key.

The Master host generates and manages a secure database “M_Keydb” for the shared session keys with the Slaves (see Figure 6-2). The database consists of four fields: the Slave address used as an index key to the database, the shared session key, the time stamp used to limit the usage of the shared key for a certain pre-defined time period, and the Key Sequence Number. The Master calls “M_GenKey” to generate a unique session key when the old session key expired. “M_PutKey” is the function used to insert the new session key into the database. And the M_PutKSN is the function used to insert the new KSN into the database.

The Slave needs to maintain two session keys, one for communicating with the Primary Master host and the other for the Secondary Master host (see Figure 6-2). It manages a secure database “S_Keydb” for the shared session keys with the Master hosts. The database consists of three fields and two records: (0, Primary Master Session Key, Key Sequence Number and 1, Secondary Master Session Key, Key Sequence Number). The simple “S_PutKey” is the function used to update the database with a new session key and the “S_PutKSN is the function used to update the database with a new KSN.

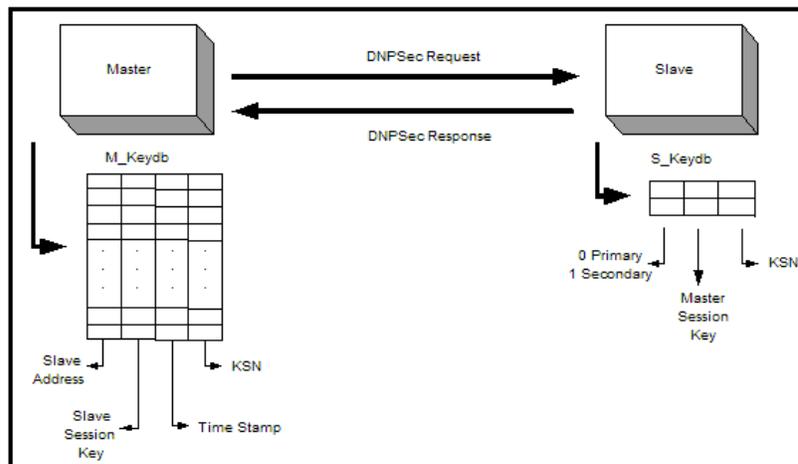


Figure 6-2: DNPSec Request / Response link communications

See appendix E for our approach to implement the DNPSec protocol.

7 Discussion

See appendix D for our discussion.

8 Conclusion

DNP3 was not designed with security capabilities in mind. The SCADA vendors can build such capabilities by utilizing the DNPSec framework with a minimum time and cost without a major impact on the systems components and the application supporting them.

This paper has discussed what DNPSec framework is, the components that constitute DNPSec, how DNPSec works, and provided analysis to our approach. The framework enables confidentiality, integrity, and authenticity in the DNP3. Such a framework requires some enhancements in the data structure of the DNP3 Data Link layer, without requiring modification to the Master Station and Substation devices and the applications supporting them. Confidentiality and integrity are achieved by encrypting frames between the Master and the Slaves using a common session key assigned at the setup time of the SCADA components. A new session key is established when the frame sequence number reaches the value $2^{32} - 1$ or when the time period for the use of the session key is expired. Authentication is achieved by applying authentication techniques to assure that the sender of the frame is what it claims to be. Proof of concept by testing or simulation could be a future topic worth investigation.

Our ongoing work address performance issues are related to the implementation of DNPSec.

Appendix A: Supervisory Control And Data Acquisition (SCADA) Systems

SCADA systems are real-time process controls that enable a user from a single Master station to collect and analyze data received from one or more Slave units, send limited control instructions, monitor and control equipment status, and open and shut valves or motors. It is used to control power generation, electricity transmission & distribution, electric utilities, natural gas utilities, oil & gas production facilities, water utilities, and gas stations/gas pumps.

A SCADA system consists of four major components: (1) the Field Instrumentations that measures the environmental conditions (read temperatures, pressures, flows, voltages, currents, frequencies, or other physical quantities) and controls valves, circuit breakers, or other devices that influence the physical processes, (2) the Remote Stations (Remote Terminal Unit), (3) the Control Center, and (4) the Communication links.

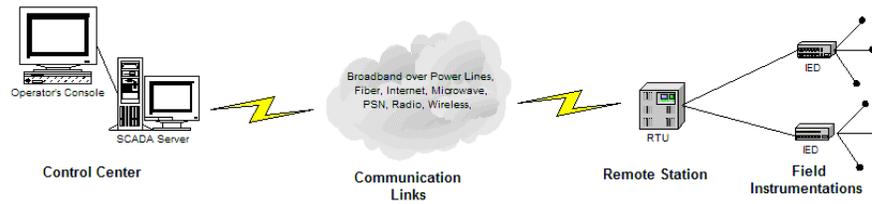


Figure 3-1: SCADA system architecture

A general architecture of SCADA components is shown in Figure 3-1. The RTUs located in the remote station poll many direct-wired field instruments. The RTUs are used for reporting data and controlling the remote devices. The RTUs collect and concentrate the data from the field instruments for transfer to the Master Station. The SCADA server using Human Machine Interface located in the control center polls the RTUs at a user defined polling rate over several communication links (Fiber, Radio, Modem, Microwave, Telephone, Wireless, Powerline Carrier, and Internet). The SCADA server is programmed to collect and reformat data from the remote sites and detect alarm messages reported by the RTUs. Alarm messages are reported when a status of an end point changed unexpectedly.

Several protocols are used to transmit messages between SCADA components. The following is a list of some of these protocols:

- Distributed Network Protocol Version 3, to be discussed in section 4
- Highway Addressable Remote Transducer (HART) Protocol [11],

- A global standard for smart instrument communication
 - Designed to support the traditional 4-20mA analog signaling
 - Support three layers stack: Application, Data Link, and Physical
- Inter Control Center Protocol (ICCP) [13],
 - Also known as TASE.2
 - Developed to allow real-time data, schedule, and control command exchanges to occur between two or more utility control centers, regional control centers, and power pools.
- International Electrotechnical Commission (IEC) 61850 [14],
 - Intra-substation communications
- Modicon Bus (Modbus), Modbus-Plus [21]
 - An application layer messaging protocol, positioned at level 7 of the OSI model, which provides client/server communication between devices connected on different types of buses and network.

SCADA messages comprise commands, responses, acknowledgments, negative acknowledgments, keep-alive messages, etc. We will discuss these further in the next section.

Appendix B: Distributed Network Protocol Version 3

DNP3 supports two kinds of data: static data and event data. Static data is called class 0 data. Event data can have three different classes or priorities: 1 (high priority), 2 (medium priority) and 3 (low priority). Also, it supports several data types, e.g. “binary input” and “analog input”, and the corresponding events, e.g. “binary input change” and “analog change”.

DNP3 protocol uses two message sets. The Master set, contains the valid commands for Master initiation a request (polling) or issuing command confirmation, and the Slave set, contains the valid commands to provide a response or initiation unsolicited messages. Messages can be sent between the DNP3 Master (Control Center) and the Slave (RTU) by one of the following communication operating modes [8]:

- Quiescent Operation. In this mode the Master does not poll the Slave. The Slave can send unsolicited report-by exception messages and the Master can send application layer confirmations to the Slave. During quiescent periods the device can be placed in an idle state.
- Unsolicited Report-by-Exception Operation. The communication is basically, unsolicited but the Master occasionally sends integrity polls for Class 0 data to verify that its database is up to date.
- Polled Report-by-Exception Operation. The Master regularly polls for event data and occasionally for Class 0 data.
- Static Report-by-Exception Operation. The Master polls only for Class 0 data or the specific data it requires.

The Master can address individual Slaves, or can initiate a broadcast message to all Slaves. Slaves return a message (response) to requests that are addressed to them individually. The DNP3 protocol establishes the format for the Master’s request message by placing it into the Slave (or broadcast) address, a function code defining the requested action, any data to be sent, and an error-checking field. The Slave’s response message is also constructed using DNP3 protocol. It contains fields confirming the action taken (if requested), any data to be returned, and an error-checking field. If an error occurred in receipt of the message, or if the Slave is unable to perform the requested action, the Slave will construct an error message and send it as its response.

Figure 4-1 shows the communication process between a Master and a Slave. The figure illustrates a Master initiates a request of data from a Slave, this could be a poll for current data. Also, the figure illustrates the communication sequence between a Master and a Slave with message direction shown between them.

The request message is contained in the application layer information within the message. A confirmation (acknowledge) response is required to this message. The Slave station sends an ACK message to the Master.

Since the last transaction contained an application level request for the transmission of data, the Slave station then performs the action requested and initiates a communication with the requested data.

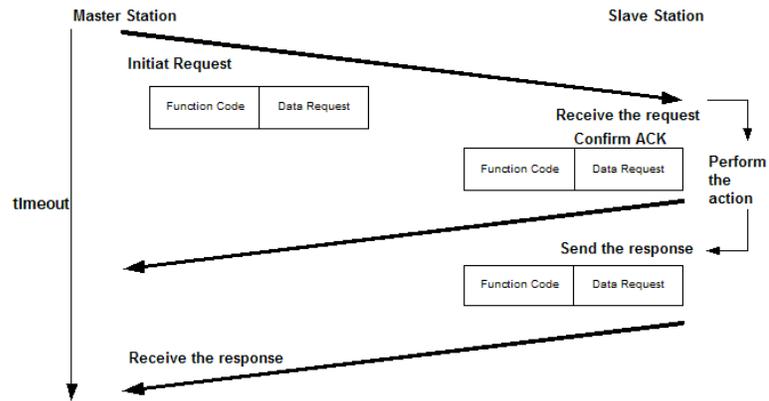


Figure 4-1: Master/Slave Communication Process

The DNP3 format structure is explained in more detail in the following paragraphs.

The DNP3 stack has three layers: the application layer, the data link layer, and the physical or adaptation layer. To support advanced Remote Terminal Unit (RTU) functions and messages larger than the maximum frame length, DNP3 added a transport pseudo-layer to be used with the Data Link layer (See figure 4.2).

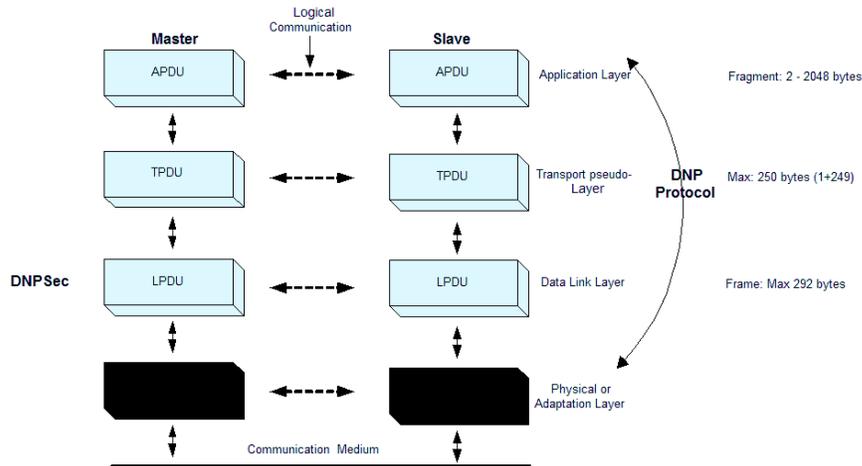


Figure 4-2: DNP3 Protocol Stack [32]

The *application layer* specification describes the message format, attributes, services, and procedures for the application layer. The application layer listens and builds messages based on the need for or the availability of user application data. When the data to be transmitted is too large for a single application layer message, multiple independent application layer messages may be built and transmitted sequentially. Each application layer is referred to as a fragment and a message may either be a single-fragment message or a multi-fragment message. Once messages are built, the application layer presents a buffer containing an application layer fragment to the pseudo-transport layer for segmentation.

The application layer initially forms the data into blocks of application service data units (ASDU). The application messages contain both function codes, and data objects. Function codes define what the meaning and purpose of a message. The data objects define the structure and interpretation of the data itself. The ASDU is made up of one or more object header and data object fields with up to a maximum size of 2048 byte.

The application layer then creates the application protocol data units (APDU) by adding a message header to the ASDU. The header has 2 bytes or 4 bytes depending on if the message is a request or a response, respectively. The application header is referred to as the application protocol control information (APCI). The response header includes an additional two-byte field designated internal indications (IIN). When a request cannot be processed due to formatting errors or the requested data is not available, the IIN is always returned with the appropriate bits set. The other two components of the header are the application control (AC) and the function code (FC). The AC is used to control the communication flow. It has three bits or flags, plus a sequence number. The flags are used to indicate if the fragment is one of a multi-fragment message, if an application layer confirmation is requested for the

fragment, and if the fragment was unsolicited. The application layer sequence number allows the receiving application layer to detect fragments that are out of sequence, or dropped fragments.

The FC is the second byte of the APCI. It follows the AC byte in both the request header and the response header. The FC indicates the purpose, or requested operation, of the message. The header function code applies to all object headers, and therefore all data within the message fragment. For example, function code 7 represents the immediate freeze function. When such function code is coded in the control function code, the Master is asking the Slave to copy the specified objects to a freeze buffer and respond with status of the operation. The rest of the function codes could be found in the DNP Users Group documentations.

The *pseudo-transport layer* specification describes the segment format, attributes, services, and functions for the pseudo-transport layer. The main function of the transport layer is to disassemble application layer fragments into data link layer sized data units (segments) for transmission and to reassemble these units into the original application fragment on reception. It listens and builds segments based on the need for or the availability of the fragment data. It breaks the application protocol data units down into multiple data segments called transport protocol data units (TPDU). The maximum size of the TPDU is 250 bytes. It consists of 1 byte transport header followed by a maximum of 249 bytes of data. The transport header indicates if the segment is the first segment of the fragment using the FIN field (FIN = 0 indicates more segments follow, FIN = 1 indicates the final segment in a series of segments), the last segment of a fragment using the FIR field (FIR = 0 indicates this is not the first segment in a series of segments, FIR = 1 indicates this is the first segment in a series of segments), or a single segment. The transport header also includes a rolling segment sequence number to verify that segments are received in the correct order and guards against duplicated or missing segments. It has a range of 0 to 63. Sequence numbers increment by one count modulo 64, for each segment in a series of segments that hold an application layer fragment. After sequence number 63, the next value is 0.

The *data link layer* manages the logical link between the Master and the Slave and it improves the physical channel error characteristics. It adds a 10 bytes header block (block 0) followed by optional data blocks to constitute the link protocol data unit (LPDU) or frame of up to 292 bytes in length. If more than 16 user data bytes follow block 0, each block must contain 16 bytes of data except the last block. Each data block has 2 CRC bytes appended to it, See Figure 4-3.

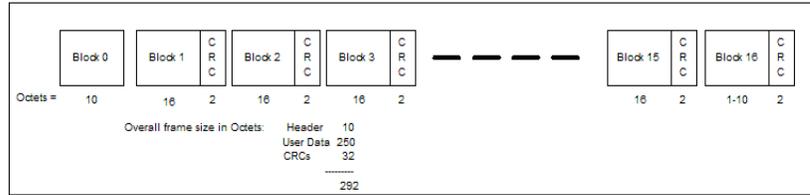


Figure 4-3: DNP3 User Data Frame

The header block fields, Figure 4-4, consists of two “sync” bytes that help the receivers determine where the frame begins; one “length” byte that specifies the number of bytes in the remainder of the frame, not including Cyclic Redundancy Check (CRC) check bytes; one “link control” byte used between sending and receiving link layers to coordinate their activities; two “destination address” bytes; two “source address” bytes; and two bytes for cyclic redundancy check. The data section is commonly called the payload and contains the data passed down from the pseudo-transport layer.

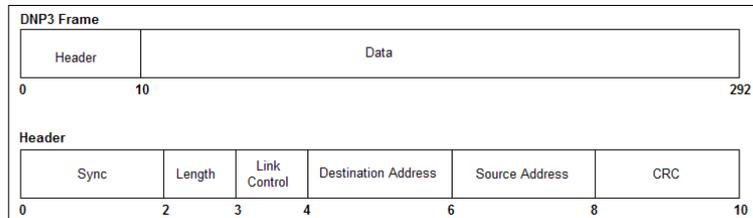


Figure 4-4: DNP3 Frame data structure

The data link control function code identifies the purpose of the message and indicates what function is required to be performed. It contains the direction of the frame, type of frame and flow control information. Figure 4-5 defines the fields of the control byte. Possible data link control byte values include: ACK, NACK, link needs reset, link is reset, request data link confirm (ACK) of frame, request link status, and link status reply. When a data link confirmation is requested, the receiver must respond with an ACK data link frame if the frame is received and passes CRC checks. If a data link confirmation is not requested, no data link response is required.

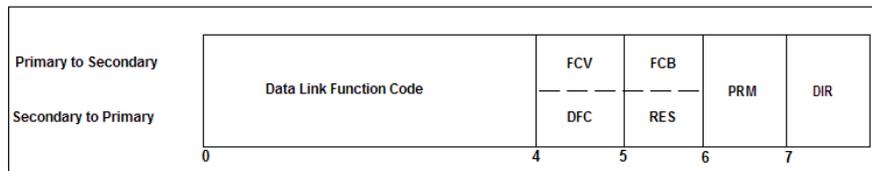


Figure 4-5: DNP3 Data Link Control Byte

DNP3 link layer message has two addresses: source address indicates where the message is coming from and destination address indicates where the message is going

to. DNP3 allows for addresses from 0 – 65534 for individual device identification, with the address 65535 defined as an all Slave units address to send broadcast commands such as freeze counters to all units on a communication channel. The main purpose of the DNP3 data link layer is to convert the TPDU into one frame (LSDU) to transfer the frame across the physical link, and to provide indications of other events such as link status.

Finally, the **physical layer** converts each frame into a bit stream and sends it over to a physical media. In another configuration, the TCP/IP protocols are used to provide transport of the DNP3 messages over the network, such configuration is called adaptation layer.

The following example (Figure 4-6) illustrates the three layer functions. Application layer byte numbers are shown inside the elements.

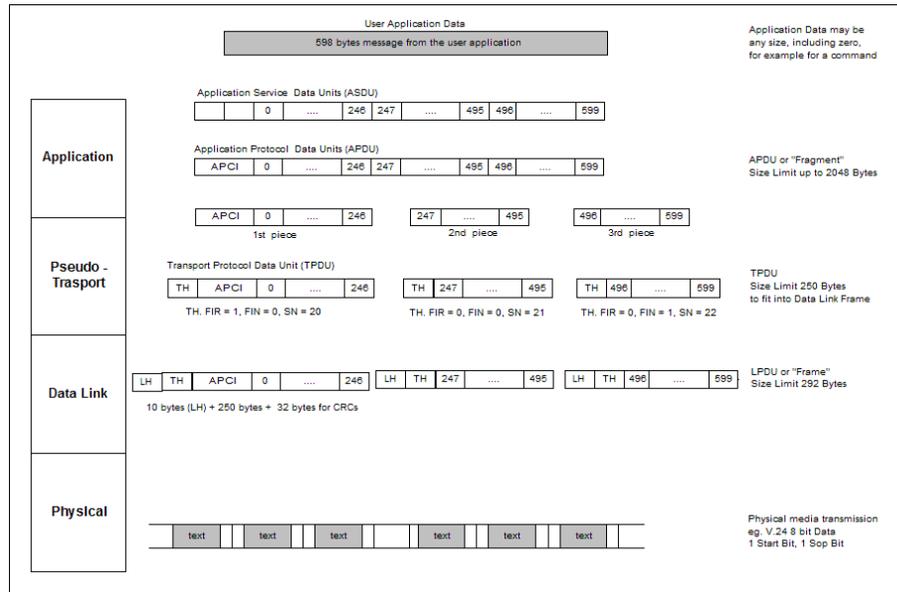


Figure 4-6: DNP3 data structure

1. The user application function:
 - a. The user application generates a message with the appropriate function code and data objects to poll the right information from the remote station. The message size is 600 bytes. The application layer receives the message from the user application function and reformats the message to fragments.
2. The application layer functions:

- a. Since the message size is less than 2048 bytes, the application layer will form one fragment.
 - b. It adds 2 bytes header to the message since this is a polling message, one byte for the application control and the second byte for the function code.
 - c. The values in the application control byte [FIR FIN CON SEQ- NUM] could be as follows:[1 1 0 01001], the code indicates that the message is one fragment (1 1), no confirmation required (0), with 9 (01001) as the sequence number (solicited request) value
 - d. The function code value could be as follows: [0000001], this function requests from the substation to respond with requested data objects.
 - e. The application layer transmits the fragment to the pseudo-transport layer for assembly. Now the size of the fragment is 602 bytes.
3. The pseudo-transport layer functions: The pseudo-transport layer breaks the fragment to segments. Each segment will consist of 1 byte transport header and 249 bytes of data.
- a. Accordingly, 3 segments will be assembled as follows:
 Segment 1: [(0 1 010100) + 249 bytes of data] the header indicates that more segments to follow (0), it is the first segment (1), and 20 (010100) is the sequence number

 Segment 2: [(0 0 010101) + 249 bytes of data] the header indicates that more segments to follow (0), it is not the first segment (0), and 21 (010101) is the sequence number

 Segment 3: [(1 0 010110) + 104 bytes of data] the header indicates that this is the final segment (1), it is not the first segment (0), and 22 (010110) is the sequence number.
 - b. The pseudo-transport layer transmits the three segments to the data link layer to form the frames by adding the link header to each segment.
4. The data link layer functions: The data link layer builds the contents of the three frames by adding 10 bytes data link header (block 0) and 282 user data.
- a. The frame format will be as follows: [(start:2) (length:1) (control:1) (dest:2) (source:2) (CRC:2) (USER DATA):282].
 - b. An example, frame 1 will have the following data:
 [Start:0X0564, Length:255, Control:111001001, Destination:0x05, Source:0x00 CRC:0x12 User-Data:282-bytes]
 .
 - c. After creating the frames, the data link transmits the frames to the physical layer.

5. The physical layer: Converts the frames to a bit stream and sends over the communication link.

Appendix C: Threats Analysis for SCADA Systems

Security was not part of the design of the DNP3. DNP3 does not include authentication or encryption technologies in its structure. Actually, DNP3 recommends through its documentation not to use authentication capabilities when the Slave respond to any command [c].

Attack Scenario

The attack scenario in this section describes several ways the Malicious Intruder (Intruder) uses to compromise the security of SCADA systems and networks.

The Intruder could use protocol analyzer tools such as “Ethereal” or other well known techniques to intercept the DNP3 frames. As a result, the Intruder grabs unencrypted (plaintext) frames from a DNP3 SCADA system network application. By doing so, the Intruder will capture the address of the source and destination systems.

The Intruder could use the unencrypted data frames contain control and settings information in subsequent attacks on either the SCADA system or the Intelligent Equipment Devices (IEDs). Such attacks could take the form of shutting off the MTU software, shutting down the MTU computer, or the RTU stops functioning. In addition, the Intruder could change the settings on the IED, controller, or SCADA system such that the equipment either (a) fails to operate when it should, causing bus, line, or transformer damage, or (b) operates when it shouldn't, causing service interruption [29].

In our scenario, a Intruder, after managing to get between the Master and the Slave, intercepts the transmission of the frames and implement his/her attack in two phases:

Plan the attack: An important feature of DNP3 is the ability for the Slave to generate unsolicited report by exception (RBE) event and send it to the Master. Unsolicited message (alarming) generation for event reporting is configurable by the Master Station through the usage of the configuration functions in the application function code. The Intruder understands the structure of the DNP3 protocol and plans his attack by following these steps:

1. The Master initiates a connection with the Slave
2. Unknown to both the Master and the Slave, the Intruder is waiting to intercept their connection

3. The Intruder receives Master's request for a connection (authentication capability is not implemented in DNP3, so the Intruder does not have to authenticate himself to the Slave)
4. The source address (#0), the destination address (#245), the function codes, and the data objects are available in clear text

Disable DNP3 unsolicited messaging(alarming) by attacking one or more Slave units: The Intruder implements his/her attack by following these steps:

5. The Intruder then initiates a connection with the Slave posing as Master
6. The Intruder sends a message to Slave unit #245 with code function code 21 (disable unsolicited messages).
7. Slave unit #245 receives the message and disables unsolicited messages function. At this point the Slave will not be able to send any alarming messages to the Master in case there is a failure or abnormal operation at the Slave unit.
8. The Intruder sends another message with code function code #18 to Slave #245. Code #18 gives instructions to the Slave to stop running the application specified in the message.
9. A simultaneous attack on other Slave units will disable all operations on a communication channel (DNP3 has the capability to send a broadcast message to all Slave units by using address #65535). This could interrupt the utilities services at that region, like shutting down the electricity services.
10. At this stage, the Master Terminal Unit (MTU) in the Master Station reports that the application is running normally, while the Remote Terminal Unit (RTU) in the Slave Stations receives tampered frames.

The best way to protect a communications network is the correct and conscious use of cryptographic and authentication suites at the DNP3 Data Link layer in both the master (client) and the slave (server) ends.

Appendix D: Discussion

Analysis of our approach

Reliability and time to delivery of DNP3 frames are very important requirements for SCADA/DNP3 Systems. These requirements are vital to market acceptance of a particular DNP3 security implementation. Reliability, as per DNP Group, is provided by Cyclic Redundancy Code (CRC) function in the Data Link Layer. CRC is non-cryptographic mechanism for detecting transmission errors. DNPsec added more efficient reliability and security capabilities by introducing cryptographic and authentication capabilities in the DNPsec framework. Such capabilities introduced new challenges related to time to delivery of the frames. But we will show soon that these challenges are not significant and our framework maintains a good balance between security, reliability and time to deliver since most of the frames in the SCADA systems are not real-time sensitive.

As we described in section 4, DNP3 provides several different means of retrieving data. These methods for retrieving data require different means of efficiency, quiescent and unsolicited report-by-exception operation requires real-time efficiency. The time of retrieving data from the Slave or the time the Slave needs to send unsolicited messages to the Master should not be significantly delayed by the implementation of DNPsec. More detailed performance analysis related to the implementation of DNPsec needs to be conducted.

Several performance studies on the effect of cryptography on the set-up time and the delivery of the messages from one end to the other indicate that the delay is not significant based on the advanced technologies in the communication networks, processing power at the end systems, and the cryptographic algorithms [10], [22], [30].

In a study by Kim and Montgomery [19] they examined the dynamic behavior and relative performance characteristics of large scale VPN environments based upon IPsec and IKE. The results of their study are summarized in the following table:

Operation, based on 128 bit key	DES	3-DES
Encryption Speed (Kbit/s)	10508 kbit/sec	4178 kbit/sec
Decryption Speed	10519 kbit/sec	4173 kbit/sec

(Kbit/s)		
-----------------	--	--

Based on the performance information above, we will calculate the worst case scenario to measure the time of delivery for the unsolicited message from the Slave to the Master, which required real-time delivery. Although, the numbers are far from exact, they should be usable as a first approximation. The total time to deliver such message is the sum of the encryption speed (ES), the decryption speed (DS), encryption key set up (EK), decryption key set up (DK), and the transmission time (TT).

$$\text{Unsolicited delivery time} = \text{ES} + \text{DS} + \text{EK} + \text{DK} + \text{TT}$$

We assume that the size of the DNPsec message is 292 bytes, Triple DES is the algorithm of choice with 112 bit key, the network bandwidth is 1.5 Mbps, and the performance speed is measured in kbit/s. The EK and DK are not applicable in our case since we are assuming that we are using manual distribution of the session keys during the installation of SCADA components. The table below shows the performance of each operation:

Operation	Performance	Time
Encryption Speed	4178 Kbit/sec	.00007 sec
Decryption Speed	4173 Kbits/sec	.00007 sec
Transmission Time	1.5 Mbit/sec	.0002 sec

As a result the unsolicited delivery time is equal to .00034 sec. Even if we double this number to accommodate for the authentication calculation time, we believe that this is a very minimum time to have an effect on the delivery time of the unsolicited messages in the SCADA systems. Accordingly, adding the operations above to include cryptographic and authentication operations will not affect the efficiency and the speed of delivery of DNP3 messages.

Attack Analysis

Let's re-visit the man-in-the middle or the eavesdropping attack [36] described in section 5, DNPsec protects the frames between the Master and the Slaves as follows: only the Master and the Slave can read the content of the frames exchanged. A message sent from Master to Slave cannot be changed in transit (assuring integrity of the data exchanged). Master and Slave need to authenticate each other (Slave can make sure the Master is truly Master (rather than the Intruder).

DNPsec uses the fact that during installation and setup of the SCADA components a copy of the session keys stored in the Master and all Slaves to ensure protection against man-in-the-middle-attacks. When the session key expires (frame sequence

number reaches $2^{32} - 1$ or the time period using the same key reached), the Master sends a new session key to the Slave encrypted with the previous session key, and the attacker cannot recover the session key without the previous session key.

Appendix E provides SCADA / DNP3 over IP comparison.

Appendix E: SCADA / DNP3 over IP

Several SCADA vendors have successfully implemented SSL/TLS (Security Sockets Layer /Transport Layer Security) in their applications. The implementation is provided by wrapping DNP3 with SSL/TLS protocols in the transport layer level. For example, Bow Networks eLAN SSL/TLS module is currently available with DNP3 to provide secure communications in SCADA architecture [2]. Also, California Independent System Operator (ISO) in their “Remote Intelligent Gateway (RIG) Technical Specification” recommends the usage of SSL/TLS to their members [4].

SCADA/DNP3 security can also be provided by wrapping DNP3 with Internet Protocol Security (IPSec) in the Network layer. For example, Bow Networks eLAN VPN is currently available to support DNP3 in secure communications. The eLAN VPN is a stand alone application which provides a secure tunnel between two sites at the IP layer. The eLAN VPN solution is based on IPSec [2].

A summary of the advantages and disadvantages of the proposed solution, DNP3 frame with IPSec, and DNP3 frame with SSL/TLS is given in Table 7-1.

Table 7-1: Advantages and Disadvantages of DNP3Sec (proposed solution), DNP3/IPSec, and DNP3/SSL/TLS architectures.

SCADA/DNP3 Security Solutions	Advantages	Disadvantages
Wrapping DNP3 frame with SSL/TLS	<ul style="list-style-type: none"> ▪ The IEC Technical Committee has accepted SSL/TLS as part of a security standard for their communication protocol [15] ▪ Freely available for all common OS ▪ Relatively mature 	<ul style="list-style-type: none"> ▪ Run only on a reliable transport protocol (TCP and not for UDP) ▪ High performance cost ▪ No non-repudiation services ▪ Can't protect data before it is sent or after it arrives its destination ▪ Implementation of the protocol required understanding of the application, OS, and its specific system calls. ▪ CA are rather expensive and not really compatible with each other
Wrapping	<ul style="list-style-type: none"> ▪ Protection against DOS 	<ul style="list-style-type: none"> ▪ Very complex and hard to

DNP3 frame with IPSec	<ul style="list-style-type: none"> ▪ Implemented by Operating Systems, Routers, ...etc. ▪ Transparent to applications (below transport layer) ▪ No need to upgrade applications 	<ul style="list-style-type: none"> ▪ implement [9] ▪ Higher performance cost ▪ All devices shall support TCP and UDP communications on port number 20000
DNPSec	<ul style="list-style-type: none"> ▪ End-to-End security at the application level to support any communication link ▪ Protocol is simple eliminating the complexity of the key exchange and management issues ▪ Implement it once for all communication networks 	<ul style="list-style-type: none"> ▪ Required some modification to the DNP3 Data Link Layer ▪ Theoretical approach, needs to proof the concept (in going work)

Appendix E: Implementing the proposed protocol

1. During the installation of SCADA Systems, a system administrator configures all SCADA units using authentication and cryptographic algorithms as per the Company's policy. Also, a system administrator manually configures each Master/Slave with common session keys. This could be a good solution for the SCADA systems since these systems are relatively static. The Slave is only going to be exchanging data with its predefined Master (Primary or Secondary) and the same is applied to the Master. The database administrator creates M-keydb table with four fields (Slave address, Master-Slave session key, time-stamp, Key-Sequence-Number) in the Primary Master host and the Secondary Master host. The number of records will be equal to the number of Slaves associated with the Master. The following is an example of initial values of the M-keydb table:

00001	123.....567	052405:2400	00001
00002	236.....678	052405:2400	00001
00003	789.....340	052405-2400	00001
00004	666.....098	052405-2400	00001
.....	

2. The database administrator creates S-keydb table in each Slave unit with three fields (Master address, Master-Slave session key, Key-Sequence-Number). The following is an example of initial values of the S-keydb table for Slave with address 00003:

0	789.....340	00001
1	654.....002	00001

3. Initially the Key-Sequence-Number will have the value 1. Each time the Master sends a message to a Slave it will increase the Key-Sequence-Number by one. The Slave will not have the privilege to change the Key-Sequence-Number value. This will help in maintaining the value of the Key-Sequence-Number at the Master level and will give control to the Master to effectively decide when to generate and send a new session key to the Slave.
4. **Polling process (Send Request):** the Master will follow these steps:
 - a. Fetch the Slave record from the M_keydb. The Slave address will be the key for the record.
 - b. If $(\text{current-system-time} - (\text{time-stamp}) < \text{key-session-life-time})$
 100: {If $(2^{32} - 1) > \text{KSN} > 0$
 {
 Do (* all frames belong to the same message *)

```

data
    {DNP3 using DNPSec build the frames from the message;
    Encrypt the frame's original LH Header and the Payload

        using the Master-Slave session key;
    Compute the authentication data over the frame's KSN,
        Original LH Header, and the Payload Data;
    Send frame;
    } (* End Do *)
    Increment Key-Sequence-Number by 1;
    M_PutKSN (Slave-Address, new KSN value);
}
Else (* key session expired – exceeds number of transmissions *)
{Call M_GenKey(new-key);
Send a message to the Slave with the new key,
KSN = 0, and requesting confirmation message;
Wait for confirmation message;
M_PutKey (Slave-Address, new key);
M_PutKSN (Slave-Address, KSN=1);
Go-To 100;
}
Else (* key session expired – time out *)
{Call M_GenKey(new-key);
Send a message to the Slave with the new key,
KSN = 0, and requesting confirmation message;
Wait for confirmation message;
M_PutKey (Slave-Address, new key);
M_PutTime (Slave-Address, current-system-time);
M_PutKSN (Slave-Address, KSN=1);
Go-To 100;
}
5. Polling Process (Receive response): the Master will follow these rules:
a. Fetch the Slave record from the M-keydb. The Slave address will be
the key for the record.
b. While receiving all frames of same message
{Decrypt the frame using the Master-Slave session key;
Calculate and validate the authentication data;
Process the data;
}
If (current-system-time) – (time-stamp) < key-session-life-time
{If  $(2^{32} - 1) > KSN > 0$ 
{Increment KSN by 1;
M_PutKSN (Slave-Address, new KSN value);
Send a message to the Slave with the
KSN new value and SK = 1;
}
Else (* key session expired – exceeds number of transmissions *)

```

- ```

 { Call M_GenKey(new-key);
 Send a message to the Slave with the new key,
 KSN = 0, and requesting confirmation message;
 Wait for confirmation message;
 M_PutKey (Slave-Address, new key);
 M_PutKSN (Slave-Address, KSN=1);
 }
 Else (* key session expired – time out *)
 { Call M_GenKey(new-key);
 Send a message to the Slave with the new key,
 KSN = 0, and requesting confirmation message;
 Wait for confirmation message;
 M_PutKey (Slave-Address, new key);
 M_PutTime (Slave-Address, current-system-time);
 M_PutKSN (Slave-Address, KSN=1);
 }
}

```
6. **Polling Process (Accepting/processing request):** the Slave will follow these steps:
- a. If (Key-Sequence-Number <> 0)
 

```

 { S_GetKey(MH, current-key); (* get the key from S_Keydb *)
 Decrypt all frames with the same sequence number;
 Calculate and validate the authentication data;
 Process the request;
 S_PutKSN(MH, KSN);
 }
 Else (* Master sending new key or KSN to the Slave *)
 { If (KSN =0) and (SK = 0) (* new key *)
 S_GetKey(MH, current-key);
 Decrypt the frame using the current key;
 Calculate and validate the authentication value;
 S_PutKey(MH, first 64 bits of the payload); (* assuming the key
size is 64 bits *)
 S_PutKSN(MH, KSN=1);
 Send confirmation message to Master;
 }
 Else (* new KSN from the Master *)
 S_PutKSN(MH, KSN);
 }

```
7. **Polling Process (Responding Process):** the Slave will follow these steps:
- a. S\_GetKey(MH, current-key)
 

```

 S_GetKSN(MH, KSN)
 Build the frame using the application data and the information above;
 Encrypt the original header and the payload data;
 Calculate the authentication value;
 Build the frame;
 Send the frame;

```

To enable the DNPSec framework in SCADA systems, SCADA owners need to establish a security policy to identify the cryptographic, authentication algorithms, and other parameters that will be used during the data communication between the Master and the Slaves. The SCADA owners need to coordinate with the application, hardware, and software vendors to enable such algorithms in their products.

## References

1. American Gas Association (AGA), Draft 4, AGA Report 12, November 2004, Cryptographic Protection of SCADA Communications Part 1: Background, Policies and Test Plan, <http://www.gtiservices.org/security/AGA12Draft4r1.pdf>
2. Bow Networks, Inc. products  
[http://www.bownetworks.com/datasheet\\_ELANSuite\\_security.asp](http://www.bownetworks.com/datasheet_ELANSuite_security.asp)
3. Brian Broyles and Frank Kling, "Is there anything new under the SCADA sun?", December 2003 [http://www.rigzone.com/news/insight/insight.asp?i\\_id=32](http://www.rigzone.com/news/insight/insight.asp?i_id=32)
4. California ISO Remote Intelligent Gateway (RIG) Technical Specification <http://www.caiso.com/docs/2002/10/21/2002102115313210338.pdf>, pp. 19
5. Jim Coats, President Triangle MicroWorks, Inc. DNP3 Protocol AGA/GTI SCADA Security Meeting August 19, 2002 / Washington DC
6. S.Deering, R.Hinden, "Internet Protocol version 6 Specification," IETF RFC 2460, November 1998
7. DNP3 Application Note AN2003-001, <http://www.dnp.org>
8. DNP3 Users Group, <http://www.dnp.org>
9. Ferguson, Niels, and Bruce Schneier, "A Cryptographic Evaluation of IPsec," Feb 1999 Available from <http://www.faqs.org/rfcs/2402.html>
10. Gordon Clarke, Deon Reynders, "Practical Modern SCADA Protocols, May 2003 <http://cs.gmu.edu/~menasce/papers/IEEE-IC-SecurityPerformance-May-2003.pdf>
11. Highway Addressable Remote Transducer (HART) Protocol, <http://www.hartcomm.org>
12. Information Assurance Task Force of the National Telecommunications Advisory Committee. "Electric Power Risk Assessment." Alpha Communications Integration. December 2004  
<http://www.aci.net/kalliste/electric.htm>
13. Inter Control Center Protocol, Electric Power Research Institute, <http://www.epri.com>
14. International Electrotechnical Commission (IEC) 61850  
<http://www.ucausersgroup.org>
15. The International Electrotechnical Commission (IEC) "Power system control and associated communications – Data and communication security."  
<https://domino.iec.ch/webstore.nsf/artnum/030578>
16. S.Kent, R.Atkinson, "Security Architecture for the Internet Protocol," IETF RFC 2401, November 1998
17. S.Kent, R.Atkinson, "IP Authentication Header." IETF RFC 2402, November 1998
18. S.Kent, R.Atkinson, "IP Encapsulation Security Payload (ESP)," IETF RFC 2403, November 1998
19. Kim and Montgomery, Behavioral and Performance Characteristics of IPsec/IKE in Large-Scale VPNs, <http://w3.antd.nist.gov/pubs/cnis-perf-vpns-ikev1.pdf>.

- Proceedings of the IASTED International Conference on Communication, Network, and Information Security, pp. 231-236, December 2003
20. Munir Majdalawieh, Parisi-Presicce, Ravi Sandhu, "RBAC for SCADA", 2004, [http://www.mediarus.com/scada/rbac\\_4\\_scada\\_122904.pdf](http://www.mediarus.com/scada/rbac_4_scada_122904.pdf)
  21. Modicon Bus (Modbus), Modbus-Plus, <http://www.modbus.org>
  22. Erich Nahum, Sean O'Malley, Hilarie Orman, and Richard Schroeppel, "Towards High Performance Cryptographic Software" <ftp://ftp.cs.arizona.edu/reports/1995/TR95-03.ps>
  23. National Science Foundation (NSF) workshop, SCADA/IT for Critical Infrastructure Protection Workshop, Minneapolis, MN. 20-21 October 2003 <http://www.adventiumlabs.org/NSF-SCADA-IT-Workshop/>
  24. Newton-Evans Research Company, "The World Market for Substation Automation and Integration Programs in Electric Utilities: 2002-2005"
  25. Sandip C. Patel and James H. Graham "Security Considerations in DNP3 SCADA Systems" . 17<sup>th</sup> International Conference on Computer Applications in Industry and Engineering. November 17-19, 2004. <http://www.louisville.edu/speed/cecs/facilities/ISLab/tech%20papers/ISRL-04-01.pdf>
  26. J.Postel, "Internet Protocol" IETF RFC 791, September 1981
  27. The Register, "Hacker jailed for revenge sewage attacks" [http://www.theregister.co.uk/2001/10/31/hacker\\_jailed\\_for\\_revenge\\_sewage](http://www.theregister.co.uk/2001/10/31/hacker_jailed_for_revenge_sewage)
  28. Riptech, Inc. January 2001 "Understanding SCADA System Security Vulnerabilities", <http://www.iwar.org.uk/rerources/utilities/SCADAWhitepaperfinal1.pdf>
  29. Safeguarding IEDs, Substations, and SCADA Systems Against Electronic Intrusions by Paul Oman, Edmund O. Schweitzer, III, and Jeff Roberts - Schweitzer Engineering Laboratories, Inc. Pullman, WA USA
  30. Schneier, Bruce, Performance Comparison of the AES Submissions, <http://www.schneier.com/paper-aes-performance.pdf>
  31. Sauyer, Joe St, Ph.D. University of Oregon. SCADA Security, <http://darkwing.uoregon.edu/~joe/scada/>
  32. Transporting DNP V3.00 over Local and Wide Area Networks, <http://www.dnp.org>
  33. U.S. Department of Energy, "21 Steps to Improve the Cyber Security of SCADA Networks," <http://www.ea.doe.gov/pdfs/21stepsbooklet.pdf>
  34. U.S. General Accounting Office, "Critical Infrastructure Protection: Challenges and Efforts to Secure Control Systems" <http://www.gao.gov/new.items/d04354.pdf>
  35. William F. Young, Jason E. Stamp, John D. Dillinger, and Mark Rumsey. "Communication Vulnerabilities and Mitigation in Wind Power SCADA Systems" <http://www.sandia.gov/wind/other/031649c.pdf>
  36. The World Spy, Definition of man-in-the-middle, Webpage 2002-03-26, Retrieved 2002-09, <http://www.wordspy.com/words/maninthemiddleattack.asp>.
  37. Andrew Wright, John Kinast, and Joe McCarty, "Low-Latency Cryptographic Protection for SCADA Communications", <http://scadasafe.sourceforge.net/security.pdf>

