

## Journal of Cybersecurity Education, Research and Practice

---

Volume 2016  
Number 2 *Two*

Article 2

---

December 2016


# Towards an In-depth Understanding of Deep Packet Inspection Using a Suite of Industrial Control Systems Protocol Packets

Guillermo A. Francia III  
*Jacksonville State University*, [gfrancia@jsu.edu](mailto:gfrancia@jsu.edu)

Xavier P. Francia  
*Jacksonville State University*, [x.francia3@gmail.com](mailto:x.francia3@gmail.com)

Anthony M. Pruitt  
*Jacksonville State University*, [apruitt3@stu.jsu.edu](mailto:apruitt3@stu.jsu.edu)

Follow this and additional works at: <https://digitalcommons.kennesaw.edu/jcerp>

 Part of the [Information Security Commons](#), [Management Information Systems Commons](#), and the [Technology and Innovation Commons](#)

---

### Recommended Citation

Francia, Guillermo A. III; Francia, Xavier P.; and Pruitt, Anthony M. (2016) "Towards an In-depth Understanding of Deep Packet Inspection Using a Suite of Industrial Control Systems Protocol Packets," *Journal of Cybersecurity Education, Research and Practice*: Vol. 2016 : No. 2 , Article 2.

Available at: <https://digitalcommons.kennesaw.edu/jcerp/vol2016/iss2/2>

This Article is brought to you for free and open access by DigitalCommons@Kennesaw State University. It has been accepted for inclusion in Journal of Cybersecurity Education, Research and Practice by an authorized editor of DigitalCommons@Kennesaw State University. For more information, please contact [digitalcommons@kennesaw.edu](mailto:digitalcommons@kennesaw.edu).

---

# Towards an In-depth Understanding of Deep Packet Inspection Using a Suite of Industrial Control Systems Protocol Packets

## **Abstract**

Industrial control systems (ICS) are increasingly at risk and vulnerable to internal and external threats. These systems are integral part of our nation's critical infrastructures. Consequently, a successful cyberattack on one of these could present disastrous consequences to human life and property as well. It is imperative that cybersecurity professionals gain a good understanding of these systems particularly in the area of communication protocols. Traditional Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) are made to encapsulate some of these ICS protocols which may enable malicious payload to get through the network firewall and thus, gain entry into the network. This paper describes technical details on various ICS protocols and a suite of ICS protocol packets for the purpose of providing digital forensic materials for laboratory exercises toward a better understanding of the inner workings of ICS communications. Further, these artifacts can be useful in devising deep packet inspection (DPI) strategies that can be implemented in network firewalls, in expanding challenge materials for cyber competitions, and in attribution, vulnerability assessment, and penetration testing research in ICS security. We also present software tools that are available for free download on the Internet that could be used to generate simulated ICS and Supervisory Control and Data Acquisition (SCADA) communication packets for research and pedagogical purposes. Finally, we conclude the paper by presenting possible research avenues that can be pursued as extensions to this seminal work on ICS security. Prominent among these possible extensions is the expansion of the ICS packet suite to include those protocols in the wireless domain such as Wi-Fi (802.11), Bluetooth, Zigbee, and other protocols that utilizes proprietary Radio Frequency.

## **Keywords**

Industrial Control Systems, Deep Packet Inspection, SCADA, DNP3, Modbus, Ethernet/IP, Common Industrial Protocols, Packet Capture

## **Cover Page Footnote**

This work is supported in part by a Center for Academic Excellence (CAE) Cyber Security Research Program grant (Grant Award Number H98230-15-1-0270) from the National Security Agency (NSA) and a National Science Foundation (NSF) grant (Award No. 1515636). Opinions expressed are those of the authors and not necessarily of the granting agencies.

## INTRODUCTION

Industrial Control Systems (ICS) are increasingly at risk and vulnerable to internal and external threats. These systems are integral to our nation's critical infrastructures. Consequently, a successful cyberattack on one of these systems could present disastrous consequences to human life and property as well. It is imperative that cybersecurity professionals gain a good understanding of these systems particularly in the area of communication protocols. Traditional Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) are made to encapsulate some of these ICS protocols which may enable malicious payload to get through the network firewall and thus, gain entry into the network. The contribution of this paper is to describe a suite of ICS protocol packets for the purpose of providing digital forensic materials for laboratory exercises toward a better understanding of the inner working of ICS communications. These artifacts can also be useful in devising deep packet inspection (DPI) strategies and tools that can be implemented in network firewalls. Lastly, we hope that this collection of captured ICS packets could further enhance cybersecurity education and training particularly in ICS packet analysis and attribution, vulnerability analysis, penetration testing, and challenge materials for cyber competitions.

The rest of the paper is organized into three parts. First, we present a literature review of ICS protocols and technologies, deep packet inspection, and ICS packet generation and analysis. Second, we provide a technical overview and packet structure description of the prominent ICS protocols. Furthermore, we examine the details of a representative sample packet for each protocol and highlighted the essential components that define each protocol. Finally, we provide concluding remarks and present possible research avenues that can be pursued as extensions to this work.

## BACKGROUND

Industrial control system protocols range from wired to wireless. Wired protocols include Ethernet Industrial Protocol (Ethernet/IP), Common Industrial Protocol (CIP), Modbus, Modbus Transmission Control Protocol (Modbus/TCP), Distributed Network Protocol version3 (DNP3), Process Field Bus (Profibus), DeviceNet, Controller Area Network (CAN, 2013), and Ethernet for Control Automation Technology (EtherCAT). The wireless variety includes WirelessHART, 802.15 (Bluetooth), 802.16 (Broadband) and IEEE 802.15.4 (Zigbee) (Francia & Francia, 2013). With the ever increasing risk that ICS are being subjected to, it is imperative that cybersecurity professionals gain a good understanding of the communication protocols with which these systems operate and the threats that exist in securing them.

## **Deep Packet Inspection**

Packet filters found in network devices are the first line of defense against malicious packets. Deep packet inspection is a small part of the filtering techniques that are adopted by security providers in their commercial products. A non-exhaustive list of techniques is presented in (Franz & Pothamsetty, 2004) and they include the following:

- Layer 2 filtering using intelligent switches/bridges;
- Access Control Lists on Layer 3 and 4 using routers;
- Stateful Firewall filtering;
- Application Proxy filtering; and
- Deploying DPI and Intrusion Prevention Systems.

Obviously there is a great need for developing filters that are creatively constructed using DPI and intelligent mechanisms. These filters can then be applied on fields and values in control systems. These fields might include register read and write commands, controlled objects, and service requests (Byres, 2016).

Deep Packet Inspection is the process of allowing packet inspecting devices, such as firewalls and Intrusion Prevention Systems (IPS), to perform an in-depth analysis of packet contents. This in-depth analysis is much broader than common technologies in that it combines protocol anomaly detection and signature scanning to realize its potential (Ramos, 2009). For more comprehensive literature reviews on various tools and techniques for the development of DPI systems, the astute reader is referred to the work of (Antonello, et al., 2012).

## **ICS Protocol Packet Generation and Sources**

The collection of ICS protocol packet captures consists of 30 files representing 12 different ICS protocols that include, among others, CIP, DNP3, IEC 60870, EtherCAT, Ethernet/IP, Modbus, Hart, and BACnet. While a number of these files were generated in our Critical Infrastructure Security and Assessment Laboratory (CISAL) (Francia, Bekhouche, & Marbut, 2011), additional files came from multiple sources on the Internet and were compiled by Jason Smith on GitHub (Smith, 2016). Additionally, IEC 60870 packets were generated using two simulation tools: QTester104 and WinPP104. QTester104 is an implementation of the IEC60870-5-104 protocol for substation data acquisition and control over TCP/IP network (Ricardolo, 2016). WinPP104 is another implementation of the IEC 60870-5-104 protocol for listening and simulation of tele-control center and substation (Fink, 2016).

## ICS Protocol Packet Capture and Analysis

ICS packet capture and analysis can be accomplished using a freely available tool called Wireshark (Wireshark 2.0.5, 2016). This network packet analyzer tool provides a mechanism to drill down into each packet. This is a very convenient tool to use especially with ICS packets since the control information is, most likely, encapsulated by the traditional protocol frames such as TCP and UDP. Furthermore, the Wireshark website provides packet capture samples (Wireshark Sample Captures, 2016) on a number of popular ICS protocol packets.

## ICS PROTOCOLS

The following subsections will provide technical details on various ICS protocols with emphasis on their specific packet structure. The purpose of providing these details is to enable the reader to understand and interpret the information that is being uncovered when doing a deep packet inspection using Wireshark. However, due to space constraints, we simply provide what we believe is essential to get started in ICS packet analysis. It is still incumbent for the reader to follow the references cited within to get a deeper understanding of each protocol's technical details.

### Distributed Network Protocol 3 (DNP3)

The Distributed Network Protocol Version 3 (DNP3) is a protocol standard to define communications between Remote Terminal Units (RTU), master stations, and Intelligent Electronic Devices (IEDs). It was originally a proprietary model developed by Harris Controls Division and designed for Supervisory Control and Data Acquisition (SCADA) systems. DNP3 is a master/slave control system protocol and is an accepted standard by the electric, oil & gas, waste/water, and security industries (Clarke, Reynders, & Wright, 2004). There are numerous system architectural configurations for DNP3. A typical multi-drop configuration that includes one master station and multiple outstation devices is shown in Figure 1.

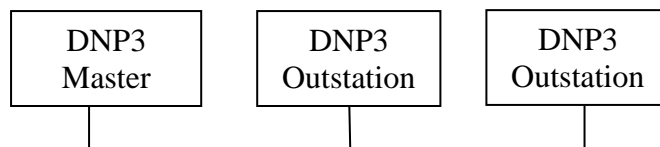


Figure 1. A DNP3 Multi-drop System Architecture

DNP3 is a four-layer subset of the OSI 7 layer model. The layers are the application, data link, physical, and pseudo-transport layers. The pseudo-transport layer includes routing, flow control of data packets, and transport functions such

as error-correction and assembly/disassembly of packets (Clarke, Reynders, & Wright, 2004). The data link header contains two Start bytes (0x0564) that help the receiver determine where the frame begins. The Length specifies number of octets of the remainder of the frame excluding the Cyclic Redundancy Check (CRC) section. The Data Link Control octet is used for the sending and receiving link layers for coordination (Curtis, 2005). The DNP3 data link frame is depicted in Figure 2.

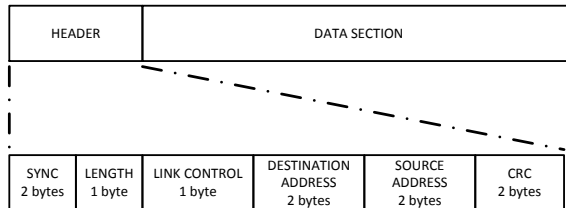


Figure 2. The DNP3 Data Link Frame

The 2-octet Destination Address specifies which DNP3 device should process the data, and the 2-byte Source Address identifies the DNP3 device sender. With the 2-byte addressing scheme, there are over 65,500 available addresses in which every DNP3 device is required to have a unique address for sending and receiving messages to and from each other. Three destination addresses are reserved by DNP3 as broadcast addresses. The data payload is divided into blocks with each block containing a pair of CRC octets for every 16 data octets except for the last block.

The pseudo-transport layer has the responsibility of breaking long application layer messages into smaller packets sized for the link layer to transmit, and, when receiving, to reassemble frames into longer application layer messages (Curtis, 2005). Note that the data link layer can only handle a maximum of 250 data octets.

The application layer fragments a message depending on the receiver's buffer size which ranges between 2048 to 4096 bytes. Consequently, a fragment of size 2048 must be broken into 9 frames by the transport layer before passing on to the data link layer.

Included in our ICS protocol packet suite is a DNP3 packet capture file with a write activity. The packet capture file is contributed by B. Wilkerson and can be found in the Wireshark sample capture repository (Wireshark Sample Captures, 2016). Figure 4 illustrates Wireshark's Packet List pane which displays the DNP3 packet. Figure 5 provides the details of the DNP3 packet. Particularly of interest are the Start bytes (0x0564), the source address (0x0003), the destination address (0x0004), and the function code (0x02).

1	0.000000	127.0.0.1	127.0.0.1	TCP	54	37712 → 20000 [SYN] Seq=0 Win=32768 Len=0
2	0.000056	127.0.0.1	127.0.0.1	TCP	54	20000 → 37712 [SYN, ACK] Seq=0 Ack=1 Win=32768 Len=0
3	0.000078	127.0.0.1	127.0.0.1	TCP	54	37712 → 20000 [ACK] Seq=1 Ack=1 Win=32768 Len=0
4	0.000174	127.0.0.1	127.0.0.1	DNP 3.0	79	from 4 to 3, Write, Time and Date
5	0.000185	127.0.0.1	127.0.0.1	TCP	54	20000 → 37712 [ACK] Seq=1 Ack=26 Win=32768 Len=0

Figure 4. Wireshark Packet List Pane with the DNP3 Pcap File

```

> Internet Protocol Version 4, Src: 127.0.0.1, Dst: 127.0.0.1
> Transmission Control Protocol, Src Port: 37712 (37712), Dst Port: 20000 (20000), Seq: 1, Ack: 1, Len: 25
> Distributed Network Protocol 3.0
  > Data Link Layer, Len: 18, From: 4, To: 3, DIR, PRM, Unconfirmed User Data
    Start Bytes: 0x0564
    Length: 18
    Control: 0xc4 (DIR, PRM, Unconfirmed User Data)
      1... .... = Direction: Set
      .1. .... = Primary: Set
      ..0 .... = Frame Count Bit: Not set
      ...0 .... = Frame Count Valid: Not set
      .... 0100 = Control Function Code: Unconfirmed User Data (4)
    Destination: 3
    Source: 4
    CRC: 0x2d15 [correct]
  > Transport Control: 0xc1, Final, First(FIR, FIN, Sequence 1)
    1... .... = Final: Set
    .1. .... = First: Set
    ..00 0001 = Sequence: 1
  > Application data chunks
    Application Chunk 0 Len: 13 CRC 0x63c8
  > Application Layer: (FIR, FIN, Sequence 1, Write)
    > Application Control: 0xc1, First, Final(FIR, FIN, Sequence 1)
      1... .... = First: Set
      .1. .... = Final: Set
      ..0 .... = Confirm: Not set
      ...0 .... = Unsolicited: Not set
      .... 0001 = Sequence: 1
    Function Code: Write (0x02)
  > WRITE Request Data Objects
    > Object(s): Time and Date (Obj:50, Var:01) (0x3201), 1 point
      > Qualifier Field, Prefix: None, Code: 8-bit Single Field Quantity
        ..00 .... = Index Prefix: None (0)
        .... 0111 = Qualifier Code: 8-bit Single Field Quantity (7)
      > Number of Items: 1
      > Point Number 0
      > Timestamp: Aug 25, 2006 15:56:00.890000000 UTC

```

Figure 5. Wireshark Packet Details Pane with the DNP3 Packet Information

## Modbus

Modbus, the most widely deployed ICS protocol, is an application layer messaging protocol. Transactions can either be a query/response type, where only a single slave is addressed, or a broadcast/no response type where all slaves are addressed. The three major implementations of Modbus include:

- Modbus Plus - a high speed token passing network;
- Asynchronous serial communication; and
- Modbus TCP – TCP/IP over Ethernet encapsulation.

The Modbus protocol specifies the Protocol Data Unit (PDU), which is comprised by the function code and the data field, and is independent of the underlying communication layers. A summary of the most important function codes is shown in Table 1. Additional fields such as the address and error-check fields augment the PDU to complete the Application Data Unit (ADU) (Modbus.org, 2012). The generic Modbus frame is depicted in Figure 6.

FUNCTION CODE	DESCRIPTION
0x01	Read Coils
0x02	Read Discrete Inputs
0x03	Read Holding Registers
0x04	Read Input Registers
0x05	Write Single Coil
0x06	Write Single Register
0x07	Read Exception Status
0x08	Diagnostics
0x0F	Write Multiple Coils
0x10	Write Multiple Registers
0x11	Report Server ID
0x14	Read File Record
0x15	Write File Record
0x16	Mask Write Register
0x17	Read/Write Multiple Registers

*Table 1. Modbus Function Code Descriptions*



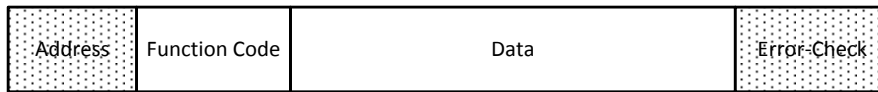


Figure 6. A Generic Modbus Frame

The single octet Address field identifies the controller/device to which the request/response is being directed. The single octet Function Code can be any code from one of these three categories: Public, User-defined, and Reserve function codes. The Data field contains the information that is being requested, the exception code, or the information, such as the addresses and number of registers, which are being passed to the server. The Error-Check field contains the Longitudinal Redundancy Check (LRC) information for the ASCII mode and for the RTU mode, the Cyclic Redundancy Check (CRC) information.

The Modbus organization, Modbus.org, extended the Modbus protocol to work over the Transmission Control Protocol (TCP) by encapsulating the Modbus PDU with the Modbus TCP ADU (Thomas, 2008). This protocol is registered to utilize port 502 and is realized by augmenting the standard Modbus PDU with a Modbus Application Protocol (MBAP) header as shown in Figure 7.

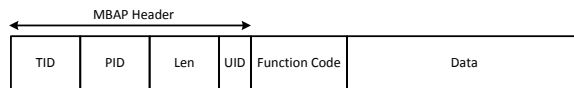


Figure 7. Modbus TCP ADU

The MBAP Header is made up the following: a two-octet Transaction Identifier (TID), a two-octet Protocol Identifier (PID), a two-octet total length (Len) in bytes of the remaining fields, and an octet indicating the Unit Identifier (UID) which identifies the remote slave connected by a serial line (Modbus.org, 2012).

Our ICS protocol packet suite includes multiple Modbus packet captures utilizing several of the functions codes shown in Table 1. These packet capture files were generated in our Critical Infrastructure Security and Assessment Laboratory (CISAL) and Control System Toolkit. These equipment are shown in Figures 8 and 9, respectively. While Figure 10 shows a listing of the captured packets, Figure 11 provides the details of the Modbus/TCP packet capture. The detailed report reveals the port number (502) that is intrinsic to a Modbus/TCP implementation and the function code that is carried by the transaction.



Figure 8. The CISAL Laboratory Toolkit



Figure 9. The Control System Toolkit

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.11.100	192.168.11.101	Modbus/TCP	66	Query: Trans: 488; Unit: 1, Func: 3: Read Holding Registers
2	0.000001	192.168.11.101	192.168.11.100	TCP	60	502 → 49618 [ACK] Seq=1 Ack=13 Win=8180 Len=0
3	0.001382	192.168.11.101	192.168.11.100	Modbus/TCP	67	Response: Trans: 488; Unit: 1, Func: 3: Read Holding Registers
4	0.062285	192.168.11.100	192.168.11.101	TCP	60	49618 → 502 [ACK] Seq=13 Ack=12 Win=20260 Len=0
5	0.515390	192.168.11.100	192.168.11.101	Modbus/TCP	66	Query: Trans: 489; Unit: 1, Func: 3: Read Holding Registers
6	0.515391	192.168.11.101	192.168.11.100	TCP	60	502 → 49618 [ACK] Seq=12 Ack=25 Win=8180 Len=0
7	0.516973	192.168.11.101	192.168.11.100	Modbus/TCP	67	Response: Trans: 489; Unit: 1, Func: 3: Read Holding Registers
8	0.577944	192.168.11.100	192.168.11.101	TCP	60	49618 → 502 [ACK] Seq=25 Ack=23 Win=20249 Len=0
9	1.031289	192.168.11.100	192.168.11.101	Modbus/TCP	66	Query: Trans: 490; Unit: 1, Func: 3: Read Holding Registers
10	1.031310	192.168.11.101	192.168.11.100	TCP	60	502 → 49618 [ACK] Seq=23 Ack=27 Win=8180 Len=0

Figure 10. Wireshark Packet List Pane with the Modbus/TCP Pcap File

```

> Ethernet II, Src: Dell_a0:57:2a (b8:2a:72:a0:57:2a), Dst: HostEngi_90:1a:d5 (00:e0:62:90:1a:d5)
> Internet Protocol Version 4, Src: 192.168.11.100, Dst: 192.168.11.101
v Transmission Control Protocol, Src Port: 49618 (49618), Dst Port: 502 (502), Seq: 1, Ack: 1, Len: 12
  Source Port: 49618
  Destination Port: 502
  [Stream index: 0]
  [TCP Segment Len: 12]
  Sequence number: 1 (relative sequence number)
  [Next sequence number: 13 (relative sequence number)]
  Acknowledgment number: 1 (relative ack number)
  Header Length: 20 bytes
  > Flags: 0x018 (PSH, ACK)
  Window size value: 20271
  [Calculated window size: 20271]
  [Window size scaling factor: -1 (unknown)]
  > Checksum: 0xee9d [validation disabled]
  Urgent pointer: 0
  > [SEQ/ACK analysis]
  [PDU Size: 12]
v Modbus/TCP
  Transaction Identifier: 488
  Protocol Identifier: 0
  Length: 6
  Unit Identifier: 1
v Modbus
  Function Code: Read Holding Registers (3)
  Reference Number: 0
  Word Count: 1
    
```

Figure 11. Wireshark Packet Details Pane with the Modbus/TCP Packet Information

## IEC 60870.5

IEC 60870.5 is an open standard created by the International Electromechanical Commission (IEC) for the transmission of SCADA telemetry control and data (Clarke, Reynders, & Wright, 2004). Of particular interest in this paper is IEC 60870-5-104, a companion standard which defines the transport of controls and data of a SCADA system over TCP/IP networks.

A portion of the generic protocol frame structure of the IEC 60870-5-104 companion standard, termed as the Application Protocol Data Unit (APDU), is shown in Figure 12. The Application Protocol Control Information (APCI) is comprised of the first 6 octets and the remainder of the APDU contains the Application Service Data Unit (ASDU). The APCI control field can be either one of the following types: Information, Supervisory, or Unnumbered. These are very similar to the control fields used in High-level Data Link Control (HDLC), a synchronous data link layer protocol developed by the International Organization for Standardization (ISO) (Tanenbaum & Wetherall, 2010). The Control Field structure for the Information type is shown in Figure 13. The two least significant bits indicate the type APCI control field, i.e. 10 for supervisory, 11 for unnumbered, and 0- for information.

Start (0x68)	Length	Control 1	Control 2	Control 3	Control 4	ASDU
--------------	--------	-----------	-----------	-----------	-----------	------

Figure 12. Application Protocol Data Unit (APDU) Generic Structure

The ASDU structure includes the data unit identifier and the data payload of one or more information objects. The data unit identifier defines the specific type of data, the address determines the identity of data, and the Cause of Transmission (COT). The COT is used to interpret the information received at the destination. The 8-bit wide code types, shown in Table 2, identify the 58 groups that are currently defined. Furthermore, information code references, defined in IEC 60870-5-5, are provided in the ASDU. For instance, code type number 9 has reference code *M\_ME\_NA\_I* indicating “Measured value, Normalized value.” Detailed descriptions of all type group and reference codes are found in (Clarke, Reynders, & Wright, 2004).

Send Sequence Number	0	Control Field 1
Send Sequence Number		Control Field 2
Receive Sequence Number	0	Control Field 3
Receive Sequence Number		Control Field 4

Figure 13. Control Field Information Type Structure

CODE TYPE RANGE	GROUP
1-21, 30-40	Process information in monitor direction
45-51	Process information in control direction
70	System information in monitor direction
100-106	System information in control direction
110-113	Parameter in control direction
120-126	File Transfer

Table 2. Code Type Groups

Just like Modbus/TCP is tied to specific port (504), IEC 60870-5-104 is assigned TCP port number 2404. Thus, deciphering and identifying ICS packets, at least for these two protocols, would be much easier by focusing on the port numbers that are being used for communications.

Our ICS protocol packet suite includes several IEC 60870-5-104 packet capture files. Figure 14 depicts the ACPI and ASDU components of the packet structure. Figure 15 provides the details of transmission such as control type, code, transmit and receive sequence numbers, and reference descriptions. A careful scrutiny of the information provided would reveal the following information: ASDU type code 50 with reference C\_SE\_NC\_1 (“Set point command, short floating point number”), the APCI is Information type with Transmission sequence number 74 and Receive sequence number 16, the cause of transmission (COT) is 3 (Spontaneous), the Information Object Address (IOA) of 12, and a floating point value of 9.87.

No.	Time	Source	Destination	Protocol	Length	Info
81	55.363229	10.20.102.1	10.20.100.108	104apci	60	<- S (72)
82	55.384742	10.20.100.108	10.20.102.1	TCP	60	2404 -> 46413 [ACK] Seq=2219 Ack=319 Win=1826 Len=0
83	56.385114	10.20.100.108	10.20.102.1	104apci	60	-> S (15)
84	56.584637	10.20.102.1	10.20.100.108	TCP	54	46413 -> 2404 [ACK] Seq=319 Ack=2225 Win=65205 Len=0
85	57.362293	10.20.102.1	10.20.100.108	104asdu	74	<- I (15,72) ASDU=10 C_SE_NC_1 Act IOA=12
86	57.387127	10.20.100.108	10.20.102.1	104asdu	74	-> I (72,16) ASDU=10 C_SE_NC_1 ActCon IOA=12
87	57.584703	10.20.102.1	10.20.100.108	TCP	54	46413 -> 2404 [ACK] Seq=339 Ack=2245 Win=65185 Len=0
88	57.585190	10.20.100.108	10.20.102.1	104asdu	101	-> I (73,16) ASDU=10 C_SE_NC_1 ActTerm IOA=12   -> I (74,16) ASDU=10 M_...
89	57.784710	10.20.102.1	10.20.100.108	TCP	54	46413 -> 2404 [ACK] Seq=339 Ack=2292 Win=65138 Len=0
90	65.504778	10.20.102.1	10.20.100.108	104apci	60	<- S (75)
91	65.665405	10.20.100.108	10.20.102.1	TCP	60	2404 -> 46413 [ACK] Seq=2292 Ack=345 Win=1800 Len=0
92	67.418145	10.20.100.108	10.20.102.1	104apci	60	-> S (16)
93	67.611763	10.20.102.1	10.20.100.108	TCP	54	46413 -> 2404 [ACK] Seq=345 Ack=2298 Win=65132 Len=0
94	85.957323	10.20.100.108	10.20.102.1	104apci	60	-> U (TESTFR act)
95	85.975055	10.20.102.1	10.20.100.108	104apci	60	<- U (TESTFR con)
96	86.207771	10.20.100.108	10.20.102.1	TCP	60	2404 -> 46413 [ACK] Seq=2304 Ack=351 Win=1794 Len=0
97	106.008479	10.20.100.108	10.20.102.1	104apci	60	-> U (TESTFR act)
98	106.020138	10.20.102.1	10.20.100.108	104apci	60	<- U (TESTFR con)

Figure 14. IEC 60870-5-104 Packet List

```

v IEC 60870-5-104-Asdu: ASDU=10 C_SE_NC_1 ActTerm IOA=12 'set point command, short floating point number'
  TypeId: C_SE_NC_1 (50)
  0... .... = SQ: False
  .000 0001 = NumIx: 1
  ..00 1010 = CauseTx: ActTerm (10)
  .0... .... = Negative: False
  0... .... = Test: False
  OA: 0
  Addr: 10
  v IOA: 12
    IOA: 12
    Value: 9.87
    > QOS: 0x00
v IEC 60870-5-104-Apci: -> I (74,16)
  START
  ApduLen: 25
  .... ..0 = Type: I (0x00)
  Tx: 74
  Rx: 16
v IEC 60870-5-104-Asdu: ASDU=10 M_ME_TF_1 Spont IOA=12 'measured value, short floating point number with time tag CP56Time2a'
  TypeId: M_ME_TF_1 (36)
  0... .... = SQ: False
  .000 0001 = NumIx: 1
  ..00 0011 = CauseTx: Spont (3)
  .0... .... = Negative: False
  0... .... = Test: False
  OA: 0
  Addr: 10
  v IOA: 12
    IOA: 12
    Value: 9.87
    > QOS: 0x00
    > CP56Time: Jul 4, 2013 08:24:14.307000000 Central Daylight Time
    
```

Figure 15. A Snapshot of IEC 60870-5-104 Packet Details

## EtherNet/IP

EtherNet/Industrial Protocol (EtherNet/IP) a member of the Common Industrial Protocol (CIP) provides the network tools to deploy standard Ethernet technology to enable industrial automation applications and enterprise connectivity for data and control. This protocol runs over TCP/IP or UDP/IP. TCP/IP uses the reserved port **0xAF12 (44818)** for transmitting explicit messages while UDP/IP uses the reserved port **0x08AE (2222)** for transmitting I/O messages. A typical Ethernet frame with the encapsulated EtherNet/IP data and the encapsulated packet format are shown in Figures 16 and 17, respectively (Schiffer, 2016).

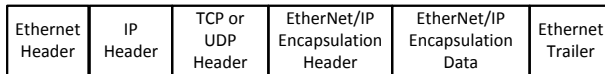


Figure 16. Ethernet Frame with Encapsulated EtherNet/IP

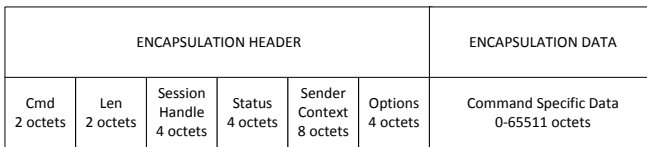


Figure 17. Encapsulation Packet Format

The two-octet command (Cmd) field represents various types of commands such as broadcast, session opening and closing, and receiving and sending data for connected and unconnected messaging (Francia & Francia, 2013).

```

> Frame 2: 138 bytes on wire (1104 bits), 138 bytes captured (1104 bits) on interface 0
> Ethernet II, Src: Rockwell_b5:63:bf (00:00:bc:b5:63:bf), Dst: Rockwell_c4:0c:bd (00:00:bc:c4:0c:bd)
> Internet Protocol Version 4, Src: 10.1.1.17, Dst: 10.1.1.10
> User Datagram Protocol, Src Port: 2222 (2222), Dst Port: 2222 (2222)
  Source Port: 2222
  Destination Port: 2222
  Length: 104
  Checksum: 0xb496 [validation disabled]
  [Stream index: 1]
  EtherNet/IP (Industrial Protocol)
    Item Count: 2
      Type ID: Sequenced Address Item (0x0002)
      Type ID: Connected Data Item (0x00b1)
    
```

Figure 18. Packet Details of Ethernet/IP over UDP

```

Transmission Control Protocol, Src Port: 44818 (44818), Dst Port: 4941 (4941), Seq: 8888, Ack: 1279, Len: 526
  Source Port: 44818
  Destination Port: 4941
  [Stream index: 0]
  [TCP Segment Len: 526]
  Sequence number: 8888 (relative sequence number)
  [Next sequence number: 9414 (relative sequence number)]
  Acknowledgment number: 1279 (relative ack number)
  Header Length: 20 bytes
  > Flags: 0x018 (PSH, ACK)
  Window size value: 4096
  [Calculated window size: 4096]
  [Window size scaling factor: -1 (unknown)]
  > Checksum: 0xf261 [validation disabled]
  Urgent pointer: 0
  > [SEQ/ACK analysis]
  [PDU Size: 526]
Ethernet/IP (Industrial Protocol), Session: 0x12020300, Send Unit Data
  Encapsulation Header
    Command: Send Unit Data (0x0070)
    Length: 502
    Session Handle: 0x12020300
    Status: Success (0x00000000)
    Sender Context: 0000000000000000
    Options: 0x00000000
  Command Specific Data
    Interface Handle: CIP (0x00000000)
    Timeout: 0
  Item Count: 2
    > Type ID: Connected Address Item (0x00a1)
    > Type ID: Connected Data Item (0x00b1)

```

Figure 19. Packet Details of Ethernet/IP over TCP

## EtherCAT

EtherCAT stands for Ethernet for Control Automation Technology. It is a control system protocol that is mainly used to satisfy very high performance requirements in the manufacturing environment. For a typical 1000 distributed points the update time is approximately 30 microseconds (Digital Bond, 2016). The format of an EtherCAT UDP frame is shown in Figure 20. The EtherCAT telegram consist of one or more datagrams, each serving a particular memory area of the logical process images of up to 4 GB in size. The EtherCAT telegram header consists of a length field, a reserve field, and a type field which indicates the nature of the data carried by the EtherCAT telegram (EtherCAT Technology Group, 2016). ICS communication messages are transported and encapsulated on EtherCAT as a UDP payload using port **0x88A4 (34990)**. EtherCAT is highly susceptible to Denial of Service (DoS) and spoofing attacks due to lack of authentication (Knapp & Langill, December 2014).

Our ICS protocol packet suite includes an EtherCAT packet capture file downloaded from the Wireshark sample capture file repository (Wireshark Sample Captures, 2016). Details of the contents of the EtherCAT telegram are shown in Figure 21. Points of interest on this EtherCAT packet snapshot are the following: the Ethernet frame type: EtherCAT (0x88a4), the Broadcast Write command (BWR) with data value 1100 as the first datagram command, the Broadcast Read (BRD) starting at slave address 0x0130, and the Auto Increment Physical Read (APRD) datagram command.

Ethernet Header	IP Header	UDP Header	EtherCAT Header	EtherCAT Command Header + Data +WC	EtherCAT Command	Ethernet CRC
14 octets	20 octets	2 octets	2 octets	10 + n + 2 octets	10 + m + 2 octets	4 octets

Figure 20. EtherCAT Telegram Structure with UDP (Digital Bond, 2016)

```

> Frame 3: 114 bytes on wire (912 bits), 114 bytes captured (912 bits)
< Ethernet II, Src: Oracle_23:98:cf (00:14:4f:23:98:cf), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
  > Destination: Broadcast (ff:ff:ff:ff:ff:ff)
  > Source: Oracle_23:98:cf (00:14:4f:23:98:cf)
  Type: EtherCAT frame (0x88a4)
< EtherCAT frame header
  .... .000 0110 0010 = Length: 0x0062
  ... 0... .. = Reserved: Valid (0x0000)
  0001 .... .. = Type: EtherCAT command (0x0001)
< EtherCAT datagram(s): 7 Cmds, SumLen 14, 'BWR'..
< EtherCAT datagram: Cmd: 'BWR' (8), Len: 2, Adp 0x0, Ado 0x120, Cnt 0
  < Header
    Cmd      : 8 (Broadcast Write)
    Index: 0x03
    Slave Addr: 0x0000
    Offset Addr: 0x0120
    > Length : 2 (0x2) - No Roundtrip - More Follows...
    Interrupt: 0x0000
    Data: 1100
    Working Cnt: 0
  < EtherCAT datagram: Cmd: 'BRD' (7), Len: 2, Adp 0x0, Ado 0x130, Cnt 0
    < Header
      Cmd      : 7 (Broadcast Read)
      Index: 0x04
      Slave Addr: 0x0000
      Offset Addr: 0x0130
      > Length : 2 (0x2) - No Roundtrip - More Follows...
      Interrupt: 0x0000
      Data: 0000
      Working Cnt: 0
    < EtherCAT datagram: Cmd: 'APRD' (1), Len: 2, Adp 0x0, Ado 0x130, Cnt 0
      < Header
        Cmd      : 1 (Auto Increment Physical Read)
        Index: 0x05
        Slave Addr: 0x0000
        Offset Addr: 0x0130
        > Length : 2 (0x2) - No Roundtrip - More Follows...
        Interrupt: 0x0000
        Data: 0000
        Working Cnt: 0
  
```

Figure 21. Packet Details of EtherCAT over UDP

## Profibus

Process Fieldbus (Profibus) is an open fieldbus serial network standard, and is mainly used for real-time control applications. The protocol operates on the application, data link, and physical layers of the OSI model. Profibus comes in three forms: Profibus Process Automation (PA), Decentralized Peripherals (DP), and Profibus Fieldbus Message Specification (FMS) (Krutz, 2006).



The *Profibus DP* telegram header (11 bytes) and data field (variable length—maximum of 244 bytes) is depicted in Figure 22. The header consists of the start delimiter (SD), the net data length (LE), the length repeated (LEr), destination address (DA), source address (SA), function code (FC), the destination service access point (DSAP), the source service access point (SSAP), the frame checking sequence (FCS), and the end delimiter (ED) (Acromag Incorporated, 2002).

SD 1 octet	LE 1 octet	LEr 1 octet	DA 1 octet	SA 1 octet	FC 1 octet	DSAP 1 octet	SSAP 1 octet	DU var	FCS 1 octet	ED 1 octet
---------------	---------------	----------------	---------------	---------------	---------------	-----------------	-----------------	-----------	----------------	---------------

Figure 22. *Profibus-DP (Message) Telegram Structure*

## ControlNet

ControlNet is a token-passing bus control network protocol that is based on the IEEE 802.4 standard. The nodes in the token bus network are configured into a ring topology, and in particular, in ControlNet, each node knows the address of the preceding and succeeding nodes (Lian, Moyne, & Tilbury, 2001). The ControlNet protocol became a part of the Common Industrial Protocol (CIP) family of protocols in 1997.

The Medium Access Control (MAC) frame format transmitted on ControlNet is shown in Figure 23. The transmitted data, which could be as many as 510 bytes, is carried by a series of link packets (LPackets). A link packet can either be a Fixed Tag or a Generic Tag. The Fixed Tag LPacketets are used for Unconnected Messaging and network administration while the Generic Tag LPacketets are used for Connected Messaging. The link data field occupies 506 bytes in the fixed tag and 505 bytes in the generic tag (Schiffer, 2016).

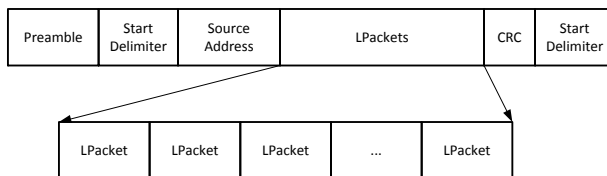


Figure 23. *ControlNet MAC Frame Format (Schiffer, 2016)*

## CONCLUSION AND FUTURE PLANS

This paper presented a review of industrial control protocols and the need for resources towards an in-depth understanding of ICS protocols. We cannot emphasize enough the fact that a solid understanding of the ICS protocols and architectures is essential to securing our critical infrastructures. We believe that this small contribution will provide indispensable materials in three areas of learning and research on ICS: digital forensics, cyber competitions, and deep packet inspection.

The challenge for the authors will be in the expansion of the collection of ICS packet samples to cover the protocols that are just evolving such as those found in the Internet of Things (IoT) and the automotive controls systems. Additional future plans include:

- Development of an intelligent attribution system utilizing the packet signature; and
- Expansion of the ICS packet samples to include those that utilize the wireless network protocols.

## ACKNOWLEDGEMENTS

This work is supported in part by a Center for Academic Excellence (CAE) Cyber Security Research Program grant (Grant Award Number H98230-15-1-0270) from the National Security Agency (NSA) and a National Science Foundation (NSF) grant (Award No. 1515636). Opinions expressed are those of the authors and not necessarily of the granting agencies.

## REFERENCES

- Acromag Incorporated. (2002). *Department of Engineering and Information Technology*. Retrieved from University of Catania: <http://www.diit.unict.it/users/scava/dispense/II/Profibus.pdf>
- Antonello, R., Fernandes, S., Kamienski, C., Sadok, D., Kelner, J., Godor, I., . . . Westholm, T. (2012). Deep Packet Inspection Tools and Techniques in Commodity Platforms: Challenges and Trends. *Journal of Network and Computer Applications*(35), 1863-1878.
- Byres, E. (2016). *Understanding Deep Packet Inspection for SCADA Security*. Retrieved from Tofino Security-Belden: <http://info.belden.com/dpi-tk-lp>
- CAN. (2013). *CAN Specification 2.0, Part B*. Retrieved 3 8, 2013, from CAN in Automation: <http://www.can-cia.org/fileadmin/cia/specifications/CAN20B.pdf>

- Clarke, G., Reynders, D., & Wright, E. (2004). *Practical Modern SCADA Protocols: DNP3, IEC 60870.5 and Related Systems*. Burlington, MA: IDC Technologies, Elsevier Ltd.
- Curtis, K. (2005). *A DNP3 Protocol Primer*. The DNP Users Group.
- Digital Bond. (2016, August). *EtherCAT*. Retrieved from Digital Bond: <http://www.digitalbond.com/scadapedia/ethercat/>
- EtherCAT Technology Group. (2016, August). *EtherCAT-the Ethernet Fieldbus*. Retrieved from EtherCat Technology Group: <https://www.ethercat.org/en/technology.html#3.1>
- Fink, R. (2016, August). *Telecontrol Protocol*. Retrieved from Reinhard Fink Engineering Office: <http://www.ppfink.de/>
- Francia, G. A., & Francia, X. P. (2013). Dissecting Industrial Control Systems Protocol for Deep Packet Inspection. *17th Annual Colloquium for Information Systems Security Education (CISSE)*, (pp. 31-36). Mobile, AL.
- Francia, G., Bekhouche, N., & Marbut, T. (2011). Design and Implementation of a Critical Infrastructure Security and Assessment Laboratory. *2011 International Conference on Security and Management (SAM2011)* (pp. 72-77). Las Vegas, NV: CSREA Press.
- Franz, M., & Pothamsetty, V. (2004). *ModbusFW Deep Packet Inspection for Industrial Ethernet*. Retrieved from Ciscoc Systems Critical Infrastructure Assurance Group (CIAG): <http://blogfranz.googlecode.com/files/franz-niscc-modbusfw-may04.pdf>
- Knapp, E. D., & Langill, J. T. (December 2014). *Industrial Network Security: Securing Critical Infrastructure Networks for Smart Grid, SCADA*. Syngress.
- Krutz, R. L. (2006). *Securing SCADA Systems*. Indianapolis: Wiley.
- Lian, F., Moyne, J., & Tilbury, D. (2001, February). Performance Evaluation of Control Networks: Ethernet, ControlNet, and DeviceNet. *IEEE Control Systems Magazine*, pp. 66-83.
- Modbus.org. (2012, April 26). *Modbus Application Protocol v1.1b3*. Retrieved from Modbus.org: [http://www.modbus.org/docs/Modbus\\_Application\\_Protocol\\_V1\\_1b3.pdf](http://www.modbus.org/docs/Modbus_Application_Protocol_V1_1b3.pdf)
- Modbus.org. (2012). *Modbus Messaging on TCP/IP Implementation Guide*. Retrieved from Modbus.org: [http://www.modbus.org/docs/Modbus\\_Messaging\\_Implementation\\_Guide\\_V1\\_0b.pdf](http://www.modbus.org/docs/Modbus_Messaging_Implementation_Guide_V1_0b.pdf)
- Ramos, A. (2009). Deep Packet Inspection Technologies. In H. Tipton, & M. Krause, *Information Security Management Handbook* (6th ed., Vol. 3). New York: Auerbach Pub.
- Ricardolo. (2016, August). *QTester104*. Retrieved from SourceForge: <https://sourceforge.net/projects/qttester104/>
- Schiffer, V. (2016, February). *The Common Industrial Protocol (CIP) and the Family of CIP Networks*. Retrieved from ODVA: [https://www.odva.org/Portals/0/Library/Publications\\_Numbered/PUB00123R1\\_Common-Industrial\\_Protocol\\_and\\_Family\\_of\\_CIP\\_Networks.pdf](https://www.odva.org/Portals/0/Library/Publications_Numbered/PUB00123R1_Common-Industrial_Protocol_and_Family_of_CIP_Networks.pdf)
- Smith, J. (2016, August). *A Collection of ICS/SCADA PCAPs*. Retrieved from Github-Automayt/ICS-pcap: <https://github.com/automayt/ICS-pcap>
- Tanenbaum, A. S., & Wetherall, D. J. (2010). *Computer Networks* (5th ed.). Pearson.

Thomas, G. (2008). *Introduction to Modbus Serial and Modbus TCP*. Retrieved from The Extension—A Technical Supplement to Control Network. Contemporary Control Systems, Inc.: <http://www.ccontrols.com/pdf/Extv9n5.pdf>

*Wireshark 2.0.5*. (2016, August). Retrieved from Wireshark.org: <https://www.wireshark.org/>

*Wireshark Sample Captures*. (2016, August). Retrieved from Wireshark: <https://wiki.wireshark.org/SampleCaptures#DNP3>