

ICS012 ICS Lab 4: Deep Packet Analysis with Wireshark

Lab Objective

The objective of this lab is to use Wireshark for deep packet inspection.

In this lab, you will learn to:

- Analyze HTTP, FTP, and Modbus packets with Wireshark.

Lab Environment

This lab requires Wireshark and capture files.

Lab Duration

30 minutes

Lab Tasks

Analyze packets using Wireshark.

Background

Wireshark is a Free and open source packet analyzer. It is used for network troubleshooting, analysis, software and communications protocol development, and education. Wireshark can be used to perform deep packet analysis of data on a network.

Lab Scenario

The packets captured on a network can be analyzed to determine patterns in information. For this lab you will perform packet analysis on HTTP, FTP, and Modbus protocols.

Lab Procedure-Using Wireshark

There are two different types of inspection when dealing with packets: shallow packet inspection and deep packet inspection. Each type of inspection is used for various tasks and has their advantages and disadvantages.

Deep packet inspection (DPI) allows the Application, Presentation, Session, and Transport layers in the OSI model to read the payload of the packet. The reading of the payload can be beneficial in instances where viruses and malware may hide in the payload of a packet.

In this lab, we will examine a sample .pcap files that were obtained through monitoring Modbus, FTP, and HTTP communications.

PART I

1. Launch Wireshark. Click File-Open and select the provided HTTP.pcap file. Fig 1

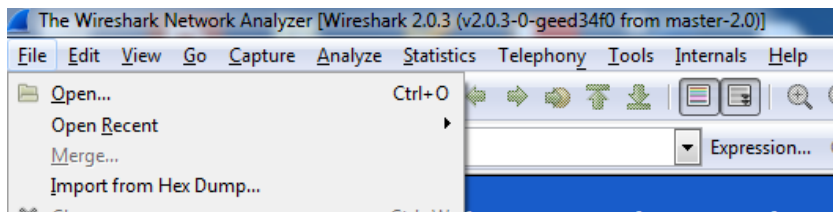


Fig. 1

2. Packets captured previously will be displayed in Packets window. Fig. 2

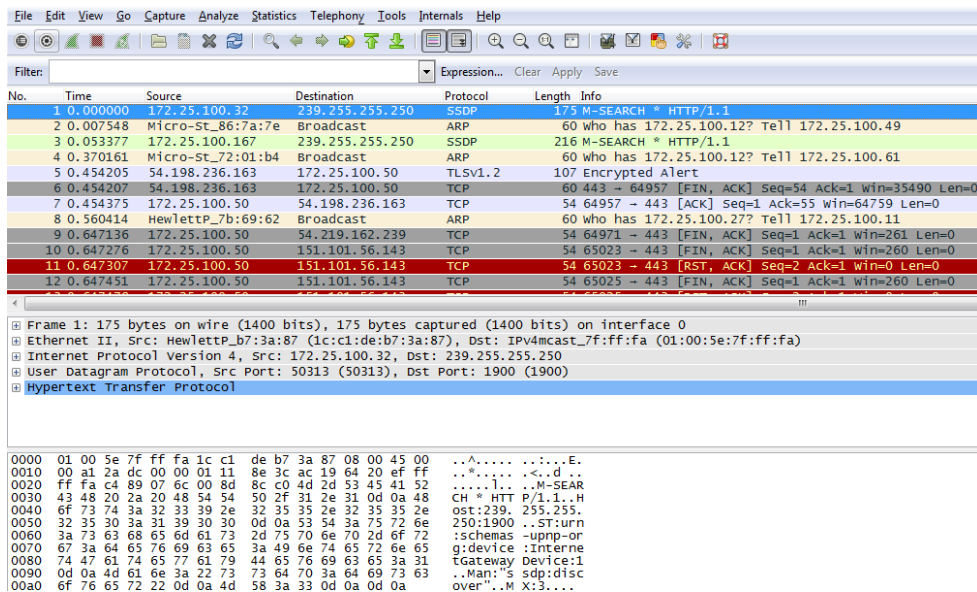


Fig. 2

3. Set a filter to display all HTTP packet details. Fig. 3

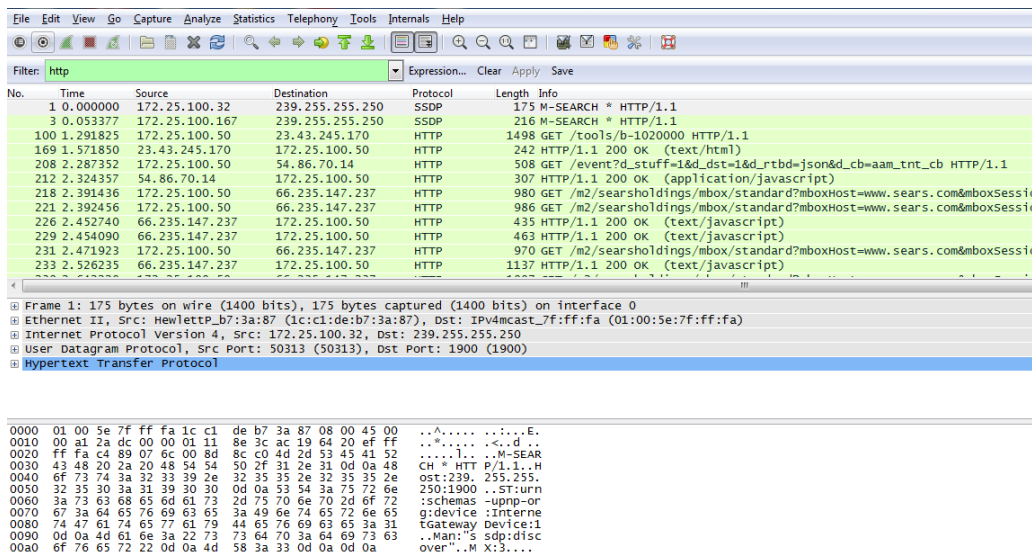


Fig. 3

4. The Hypertext Transfer Protocol (**HTTP**) is an application protocol for distributed, collaborative, hypermedia information systems. **HTTP** is the foundation of data communication for the World Wide Web. Hypertext is structured text that uses logical links (hyperlinks) between nodes containing text.
 - In viewing the HTTP packets listed, can you determine which Website the packets originate? _____

The contents of the HTTP frame offer details on three main commands. Fig. 4

- GET- Requests data from a specified resource
- PUT - Uploads a representation of the specified URI
- POST- Submits data to be processed to a specified resource

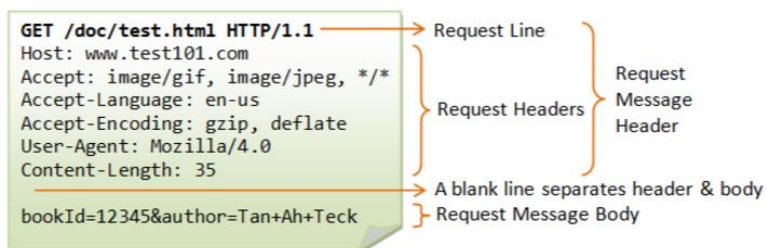


Fig. 4

5. Select packet number **328** from the packet pane to view the details of the HTTP **GET** command.

- From what host does the HTTP data originate? _____ Fig. 5

300	3.363429	00.253.147.237	172.25.100.50	HTTP	939	HTTP/1.1 200 OK (text/javascript)
303	3.373458	52.204.28.104	172.25.100.50	HTTP	872	HTTP/1.1 302 Found (text/html)
328	3.647720	172.25.100.50	23.43.245.170	HTTP	146	GET /crsp/api/cart/v1/itemcount HTTP/1.1
335	3.725365	172.25.100.50	54.86.70.14	HTTP	162	GET /event?d_nsId=0&d_ld=ts%3D1464963709279&c_pageName=Vertical%20
337	3.753448	23.43.245.170	172.25.100.50	HTTP	741	HTTP/1.1 200 OK (application/json)
341	3.761754	54.86.70.14	172.25.100.50	HTTP	910	HTTP/1.1 200 OK (application/javascript)
345	3.857227	172.25.100.50	23.43.245.170	HTTP	292	GET /crsp/api/cart/v1/itemcount HTTP/1.1
353	3.911187	172.25.100.50	52.204.28.104	HTTP	922	GET /video/embed/v5/691366/7f1ef9bac3b4b69f2652fe8db9d32189?key=59b

Frame 328: 146 bytes on wire (1168 bits), 146 bytes captured (1168 bits) on interface 0	
Ethernet II, Src: FujitsuL_bc:d2:e2 (2c:d4:44:bc:d2:e2), Dst: HewlettP_87:13:c0 (d8:9d:67:87:13:c0)	
Internet Protocol Version 4, Src: 172.25.100.50, Dst: 23.43.245.170	
Transmission Control Protocol, Src Port: 65096 (65096), Dst Port: 80 (80), Seq: 8711, Ack: 25508, Len: 92	
[3 Reassembled TCP Segments (3004 bytes): #326(1456), #327(1456), #328(92)]	
Hypertext Transfer Protocol	
GET /crsp/api/cart/v1/itemcount HTTP/1.1\r\n	
Host: www.sears.com\r\n	
Connection: keep-alive\r\n	
Accept: application/json, text/javascript, */*; q=0.01\r\n	
X-Requested-with: XMLHttpRequest\r\n	
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/50.0.2661.102 Safari/537.36\r\n	
Referer: http://www.sears.com/tools/b-1020000\r\n	
Accept-Encoding: gzip, deflate, sdch\r\n	
Accept-Language: en-US,en;q=0.8\r\n	
[truncated]Cookie: irp=c02f3643-ea8e-49f3-b5cd-f4123867deb2 jp2530kjbiyufwfh9w2UBFKyJ5ng5Ajo%2B4w%2F5a4HFjw%3D G 136842534042200027_4097\r\n	

Fig. 5

4. Packet inspection scenario:

For this exercise, we will perform a packet inspection to determine what type of images are being transferred across the local network.

Select packet number **564** from the packet pane and expand the JPEG Interchange Format details. Fig. 6

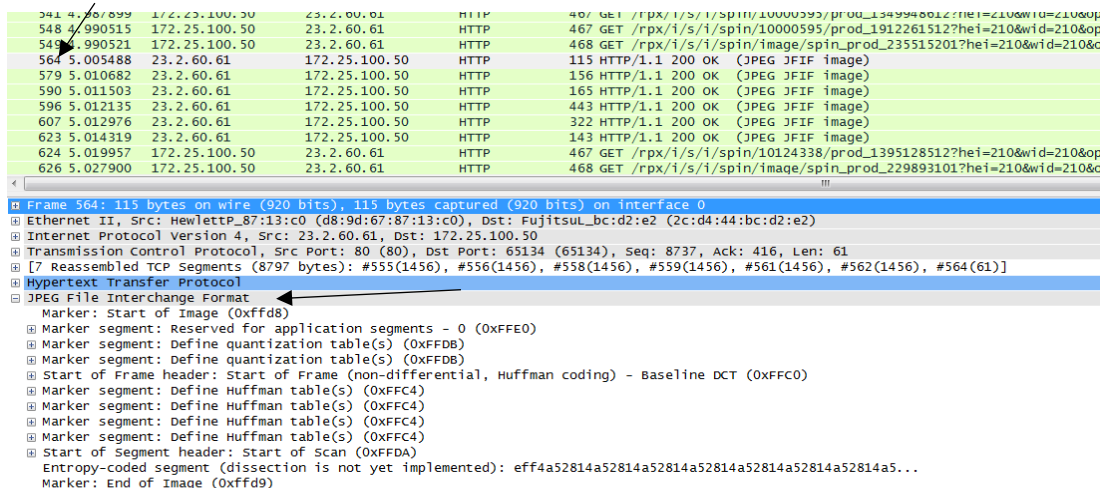


Fig 6

The goal of this packet inspection scenario is to extract the .jpeg file to determine what type of graphic it actually is.

5. Extract the .jpeg file:

Step 1: On the menu bar towards the top of the Wireshark program click on "FILE", go down to "Export Objects", next click on "HTTP" **Fig. 7**

Step 2: Click Packet Number **564** on the HTTP object list. **Fig. 7a**

Step 3: Click "Save As", choose a name for a folder where these files will be saved, click "SAVE"

Step 4: Open up your Internet Browser (Chrome, Firefox, etc.) and place the browser on the opposite side of your screen.

Step 5: Left-Click the file and drag over to the center of your Internet Browser then release.

Step 6: You should now see the graphic.

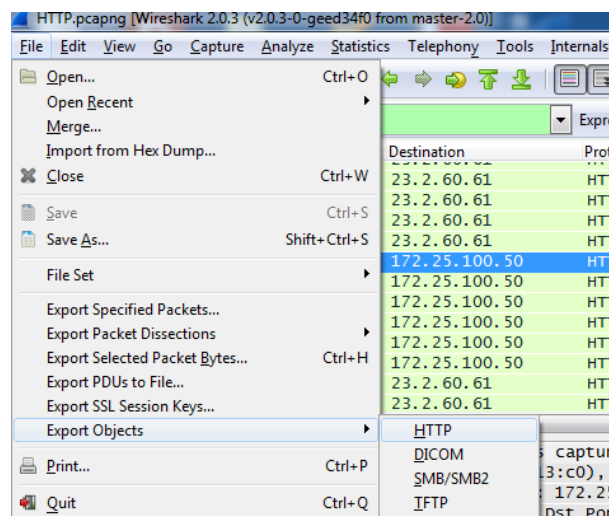


Fig. 7

Packet num	Hostname	Content Type	Size	Filename
483	wwwcdn.expotv.com	text/javascript	17 kB	player.js
487	sears.demdex.net	application/javascript	2330 bytes	event?d_nsid=0&d_id=_ts%3D1464963
493	om.sears.com	image/gif	43 bytes	s41135793572569?AQB=1&ndh=1&t=:
506	s.thebrighttag.com	application/x-www-form-urlencoded	2004 bytes	tag
513	s.thebrighttag.com	text/javascript	22 kB	tag
522	client.expotv.com	application/json	1831 bytes	7f1ef9bac3b4b69f2652fe8db9d32189
564	c.shld.net	image/jpeg	8549 bytes	spin_prod_1058520712?hei=210&wid=
579	c.shld.net	image/jpeg	10 kB	prod_1565359712?hei=210&wid=210&
590	c.shld.net	image/jpeg	8599 bytes	prod_1349948612?hei=210&wid=210&
596	c.shld.net	image/jpeg	5965 bytes	spin_prod_949833212?hei=210&wid=2
607	c.shld.net	image/jpeg	8756 bytes	prod_1912261512?hei=210&wid=210&
623	c.shld.net	image/jpeg	11 kB	spin_prod_235515201?hei=210&wid=2

Fig. 7a

6. Close the browser and HTTP.pcap file.

Part II

Examine the contents of an FTP session

File Transfer Protocol (FTP) is a standard network protocol used to transfer computer files between a client and server on a computer network. **FTP** is built on a client-server model architecture and uses separate control and data connections between the client and the server.

1. Click File-Open and select the provided FTP.pcap file. Set the filter to FTP. Fig. 8

No.	Time	Source	Destination	Protocol	Length	Info
4	0.004399	192.168.0.193	192.168.0.114	FTP	84	Response: 220 Chris Sanders FTP Server
5	0.005259	192.168.0.114	192.168.0.193	FTP	69	Request: USER csanders
6	0.006560	192.168.0.193	192.168.0.114	FTP	91	Response: 331 Password required for csanders.
7	0.007647	192.168.0.114	192.168.0.193	FTP	65	Request: PASS echo
8	0.009936	192.168.0.193	192.168.0.114	FTP	84	Response: 230 User csanders logged in.
9	0.010088	192.168.0.114	192.168.0.193	FTP	60	Request: SYST
10	0.011397	192.168.0.193	192.168.0.114	FTP	73	Response: 215 UNIX Type: L8
11	0.011529	192.168.0.114	192.168.0.193	FTP	60	Request: FEAT
12	0.013500	192.168.0.193	192.168.0.114	FTP	133	Response: 211-Extensions supported:
13	0.013710	192.168.0.114	192.168.0.193	FTP	80	Request: CLNT FlashFXP 3.4.0.1145
14	0.014991	192.168.0.193	192.168.0.114	FTP	88	Response: 200 "FlashFXP 3.4.0.1145" noted.
15	0.017594	192.168.0.114	192.168.0.193	FTP	61	Request: CWD /

Frame 4: 84 bytes on wire (672 bits), 84 bytes captured (672 bits)
Ethernet II, Src: AsustekC_40:76:ef (00:15:f2:40:76:ef), Dst: HonHaiPr_6e:8b:24 (00:16:ce:6e:8b:24)
Internet Protocol Version 4, Src: 192.168.0.193, Dst: 192.168.0.114
Transmission Control Protocol, Src Port: 21 (21), Dst Port: 1137 (1137), Seq: 1, Ack: 1, Len: 30
File Transfer Protocol (FTP)

Fig. 8

2. Now we will add a filter to Wireshark that includes a source address. In this case, the source IP is the server. Type the following in the filter box and click **Apply**:

ip.src == 192.168.0.193 Fig. 9

No.	Time	Source	Destination	Protocol	Length	Info
2	0.002319	192.168.0.193	192.168.0.114	TCP	62	21 → 1137 [SYN, ACK] Seq=0 Ack=1 win=16384 Len=0 MSS=1452 SACK_PERM
4	0.004399	192.168.0.193	192.168.0.114	FTP	84	Response: 220 Chris Sanders FTP Server
6	0.006560	192.168.0.193	192.168.0.114	FTP	91	Response: 331 Password required for csanders.
8	0.009936	192.168.0.193	192.168.0.114	FTP	84	Response: 230 User csanders logged in.
10	0.011397	192.168.0.193	192.168.0.114	FTP	73	Response: 215 UNIX Type: L8
12	0.013500	192.168.0.193	192.168.0.114	FTP	133	Response: 211-Extensions supported:
14	0.014991	192.168.0.193	192.168.0.114	FTP	88	Response: 200 "FlashFXP 3.4.0.1145" noted.
16	0.022128	192.168.0.193	192.168.0.114	FTP	109	Response: 250 CWD command successful. "/" is current directory.
18	0.024814	192.168.0.193	192.168.0.114	FTP	85	Response: 257 "/" is current directory.
24	2.655705	192.168.0.193	192.168.0.114	FTP	74	Response: 200 Type set to I.
26	2.659071	192.168.0.193	192.168.0.114	FTP	67	Response: 213 4980924
28	2.663711	192.168.0.193	192.168.0.114	FTP	103	Response: 227 Entering Passive Mode (192,168,0,193,28,86)

Fig. 9

3. Change the source address to the client IP address by adjusting the IP address to *192.168.0.114*

4. Click on packet number 5 and expand the File Transport Protocol (FTP) details. Fig. 10

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.0.114	192.168.0.193	TCP	62	1137 → 21 [SYN] Seq=0 win=16384 Len=0 MSS=1460 SAC
3	0.002338	192.168.0.114	192.168.0.193	TCP	54	1137 → 21 [ACK] Seq=1 Ack=1 win=17424 Len=0
5	0.005259	192.168.0.114	192.168.0.193	FTP	69	Request: USER csanders
7	0.007647	192.168.0.114	192.168.0.193	FTP	65	Request: PASS echo
9	0.010088	192.168.0.114	192.168.0.193	FTP	60	Request: SYST
11	0.011529	192.168.0.114	192.168.0.193	FTP	60	Request: FEAT
13	0.013710	192.168.0.114	192.168.0.193	FTP	80	Request: CLNT FlashFXP 3.4.0.1145
15	0.017594	192.168.0.114	192.168.0.193	FTP	61	Request: CWD /
17	0.023511	192.168.0.114	192.168.0.193	FTP	59	Request: PWD
19	0.175118	192.168.0.114	192.168.0.193	TCP	54	1137 → 21 [ACK] Seq=77 Ack=316 win=17109 Len=0
20	0.345347	192.168.0.114	63.245.209.21	TCP	54	4844 → 80 [FIN, ACK] Seq=1 Ack=1 win=16755 Len=0
22	0.449805	192.168.0.114	63.245.209.21	TCP	54	4844 → 80 [ACK] Seq=2 Ack=2 win=16755 Len=0

Destination Port: 21	
[Stream index: 0]	
[TCP Segment Len: 15]	
Sequence number:	1 (relative sequence number)
[Next sequence number:	16 (relative sequence number)]
Acknowledgment number:	31 (relative ack number)
Header Length: 20 bytes	
Flags: 0x018 (PSH, ACK)	
Window size value: 17394	
[Calculated window size: 17394]	
[Window size scaling factor: -2 (no window scaling used)]	
Checksum: 0x0a98 [validation disabled]	
[Good checksum: False]	
[Bad checksum: False]	
Urgent pointer: 0	
[SEQ/ACK analysis]	
File Transfer Protocol (FTP)	
USER csanders\r\n	
Request command: USER	
Request arg: csanders	

Fig. 10

- What is the client user name for this FTP session? _____
 - What is the password to complete the session establishment? _____
5. Apply the following filter and select packet number 33 to view the size of the file requested on the server side of the session: Fig. 11

Filter: ip.src==192.168.0.193 Expression... Clear Apply Save						
No.	Time	Source	Destination	Protocol	Length	Info
12	0.013500	192.168.0.193	192.168.0.114	FTP	133	Response: 211-Extensions supported:
14	0.014991	192.168.0.193	192.168.0.114	FTP	88	Response: 200 "FlashFXP 3.4.0.1143" noted.
16	0.022128	192.168.0.193	192.168.0.114	FTP	109	Response: 250 cwd command successful, "/" is current directory.
18	0.024814	192.168.0.193	192.168.0.114	FTP	85	Response: 257 "/" is current directory.
24	2.655705	192.168.0.193	192.168.0.114	FTP	74	Response: 200 Type set to I.
26	2.659071	192.168.0.193	192.168.0.114	FTP	67	Response: 213 4980924
28	2.663711	192.168.0.193	192.168.0.114	FTP	103	Response: 227 Entering Passive Mode (192,168,0,193,28,86)
30	2.664960	192.168.0.193	192.168.0.114	TCP	62	7254 → 1140 [SYN, ACK] Seq=0 Ack=1 win=16384 Len=0 MSS=1452 SACK_PERM=1
33	2.667464	192.168.0.193	192.168.0.114	FTP	158	Response: 150 Data connection accepted from 192.168.0.114:1140; transfer starting for Music.mp3 (4980924 bytes).
34	2.668383	192.168.0.193	192.168.0.114	FTP-DATA	1506	FTP Data: 1452 bytes
35	2.668689	192.168.0.193	192.168.0.114	FTP-DATA	1506	FTP Data: 1452 bytes
37	2.670340	192.168.0.193	192.168.0.114	FTP-DATA	1506	FTP Data: 1452 bytes
38	2.670640	192.168.0.193	192.168.0.114	FTP-DATA	1506	FTP Data: 1452 bytes

Fig. 11

6. Apply a final filter to try to determine the contents of the file transferred from the server. **Fig. 12**

Filter: ip.src==192.168.0.193 && ftp-data Expression... Clear Apply Save						
No.	Time	Source	Destination	Protocol	Length	Info
34	2.668383	192.168.0.193	192.168.0.114	FTP-DATA	1506	FTP Data: 1452 bytes
35	2.668689	192.168.0.193	192.168.0.114	FTP-DATA	1506	FTP Data: 1452 bytes
37	2.670340	192.168.0.193	192.168.0.114	FTP-DATA	1506	FTP Data: 1452 bytes
38	2.670640	192.168.0.193	192.168.0.114	FTP-DATA	1506	FTP Data: 1452 bytes

Frame 34: 1506 bytes on wire (12048 bits), 1506 bytes captured (12048 bits)						
Ethernet II, Src: AsustekC_40:76:ef (00:15:f2:40:76:ef), Dst: HonHaiPr_6e:8b:24 (00:16:ce:6e:8b:24)						
Internet Protocol Version 4, Src: 192.168.0.193, Dst: 192.168.0.114						
Transmission Control Protocol, Src Port: 7254 (7254), Dst Port: 1140 (1140), Seq: 1, Ack: 1, Len: 1452						
Source Port: 7254						
Destination Port: 1140						
[Stream index: 2]						
[TCP Segment Len: 1452]						
Sequence number: 1 (relative sequence number)						
[Next sequence number: 1453 (relative sequence number)]						
Acknowledgment number: 1 (relative ack number)						
Header Length: 20 bytes						
Flags: 0x010 (ACK)						
window size value: 65535						
[Calculated window size: 65535]						
[window size scaling factor: -2 (no window scaling used)]						
Checksum: 0xa394 [validation disabled]						
[Good Checksum: False]						
[Bad Checksum: False]						
Urgent pointer: 0						
[SEQ/ACK analysis]						
FTP Data (1452 bytes data)						

0000	00 16 ce 6e 8b 24 00 15 f2 40 76 ef 08 00 45 00	...n.\$..@v...E.
0010	05 d4 29 6e 40 00 80 06 49 32 c0 a8 00 c1 c0 a8	..)mè... I2.....
0020	00 72 1c 56 04 74 ea af 92 95 1b aa 1e 4d 50 10	.r.v.t... ..MP.
0030	ff ff a3 94 00 00 49 44 33 03 00 00 00 00 1f 76ID 3.....v
0040	54 41 4c 42 00 00 00 26 00 00 00 59 6e 6b 6e 6f	TALB...&...Unkno
0050	77 6e 20 41 6c 62 75 6d 20 28 32 2f 32 36 2f 32	wn Album "(2/26/2
0060	30 30 36 20 31 31 3a 31 32 3a 32 30 20 41 4d 29	006 11:1 2:20 AM)
0070	54 49 54 32 00 00 00 08 00 00 00 54 72 61 63 6b	TIT2.... ...Track
0080	20 32 4d 43 44 49 00 00 00 28 00 00 33 00 2b 00	2MCDI... (.3.+.
0090	39 00 36 00 2b 00 34 00 42 00 30 00 43 00 2b 00	9.6.+4. B.O.C.+.
00a0	38 00 37 00 45 00 37 00 2b 00 44 00 44 00 34 00	8.7.E7. +.D.O.4.
00b0	32 00 00 00 54 52 43 4b 00 00 00 02 00 00 00 32	2...TRCK2
00c0	54 43 4f 4e 00 00 00 08 00 00 00 55 6e 6b 6e 6f	TCON.... ...Unkno
00d0	77 6e 50 52 49 56 00 00 00 06 00 00 50 63 61 6b	wnPRIV... ...Peak
00e0	56 61 75 65 00 14 5b 00 00 50 52 49 56 00 00 00	Value...[...PRIV..
00f0	00 11 00 00 41 76 65 72 61 67 65 4c 65 76 65 6cAver ageLeve1
0100	00 84 0d 00 00 54 50 45 31 00 00 00 0f 00 00 00TPE 1.....
0110	55 6e 6b 6e 6f 77 6e 20 41 72 74 69 73 74 54 4c	Unknown ArtistTL
0120	45 4e 00 00 00 08 00 00 00 32 30 37 37 32 30 00	EN..... .207720.
0130	nn nn nn nn nn nn nn nn nn nn nn nn nn nn nn nn	

Fig. 12

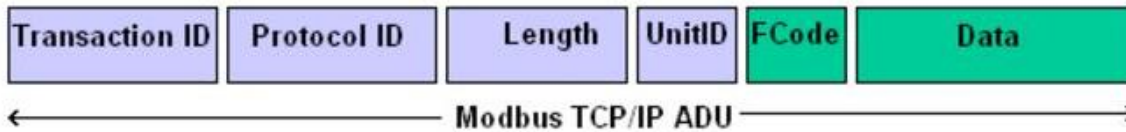
7. Close the FTP.pcap file.

Part III

Modbus TCP/IP is simply the **Modbus** Remote Transmission Unit (RTU) protocol with a **TCP** interface that runs on Ethernet. The **Modbus** messaging structure is the application protocol that defines the rules for organizing and interpreting the data independent of the data transmission medium. Modbus/TCP is typically used with control systems communication.

NOTE: Versions of Modbus data may use **asa-appl-pro**. Asa-appl-pro is an encapsulating mechanism for port 502, the standard port for transporting TCP data communication with the PLC.

The Modbus/TCP frame structure:



During the Query and Response process between the programming device and PLC, Modbus uses function codes to determine what action the devices must take. There are eight basic function codes used during the communication process.

Function code	Description
1	Read Coil.
2	Read Discrete Input.
3	Read Holding Register.
4	Read Input Register.
5	Write Coil.
6	Write Register.
15	Write Coils.
16	Write Registers.

By applying Modbus filters in Wireshark, we can determine what function the devices are performing at a given moment.

1. Click File-Open and select the provided modbus.pcap. Set the filter to **Modbus** or **tcp.port eq 502** (if using asa-appl.pro).
2. Select packet number 3. This should be the first packet in the sequence. Expand Modbus/TCP and Modbus in the details pane. Fig. 13

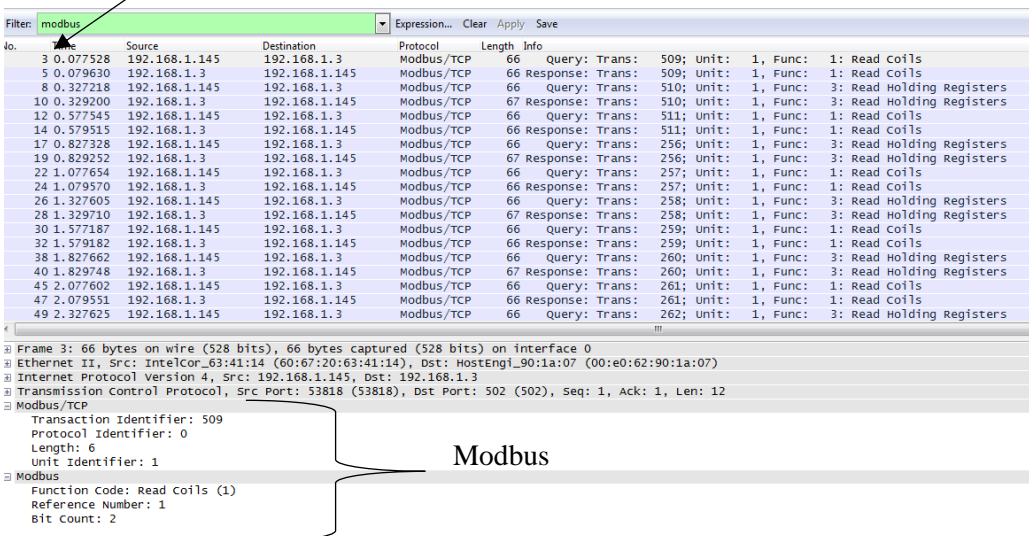


Fig. 13

3. In packet sequence #3, 192.168.1.145 is the programming device and 192.168.1.3 is the PLC. The payload of the TCP transmission contains the following information. Fig. 14

Fields	Length	Description -	Client	Server
Transaction Identifier	2 Bytes	Identification of a MODBUS Request / Response transaction.	Initialized by the client	Recopied by the server from the received request
Protocol Identifier	2 Bytes	0 = MODBUS protocol	Initialized by the client	Recopied by the server from the received request
Length	2 Bytes	Number of following bytes	Initialized by the client (request)	Initialized by the server (Response)
Unit Identifier	1 Byte	Identification of a remote slave connected on a serial line or on other buses.	Initialized by the client	Recopied by the server from the received request

Modbus TCP/IP Application Data Unit (ADU)

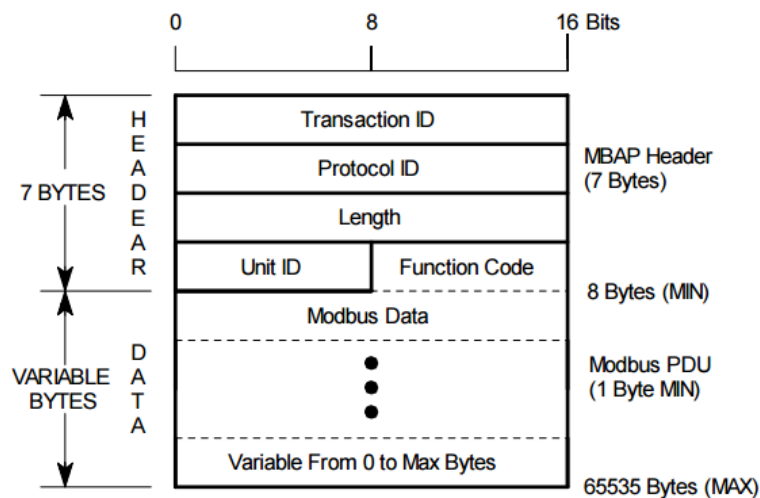


Fig. 14

- For every query, there should be a response. Packet number **3** contains the transaction ID of **509**. The response from the PLC should have the matching ID (**509**). This is packet number **5** in your capture.

The query is asking for the status of two coils in the program, as noted in the Bit Count. The response from the PLC returns the status of the coils with a data code of x03, meaning both outputs are currently ON. I.e. 00000011. If there were 3 outputs that were on and being monitored in this program, the query Bit Count would be 3 and the response Data would be 07

The Read Coil Status query specifies the starting coil (output channel) and quantity of coils to be read. The Read Coil Status in the response message is packed as one coil or channel per bit of the data field as 1 for ON (conducting current), and 0 for OFF (not conducting).

Viewing the output of both Query and Response packets, it is noted that for the status of the coils read, a response is sent. Fig. 15 and 15a

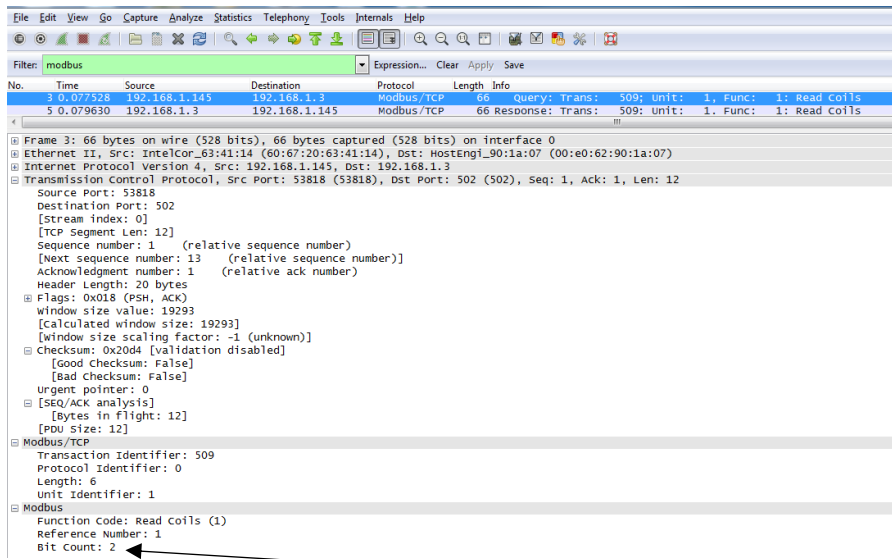


Fig 15

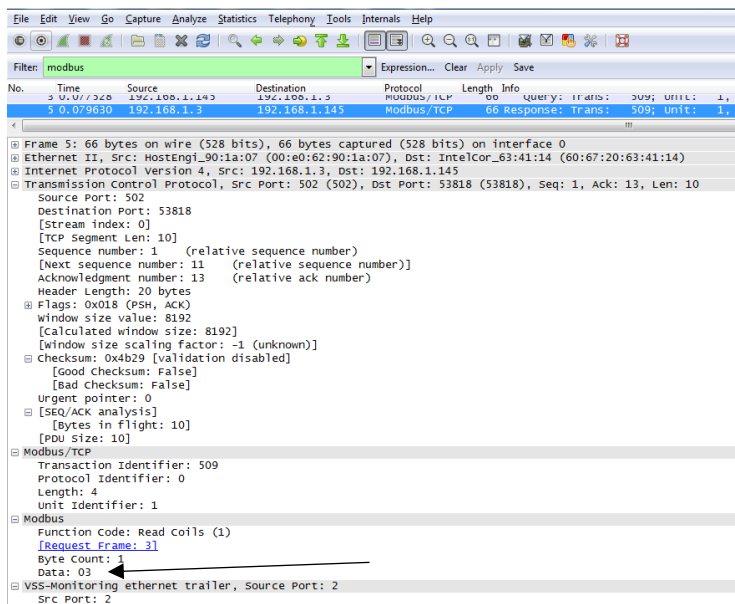


Fig 15a.

In a future lab, we will use this knowledge to exploit a PLC.