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.

<u>PART I</u>

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

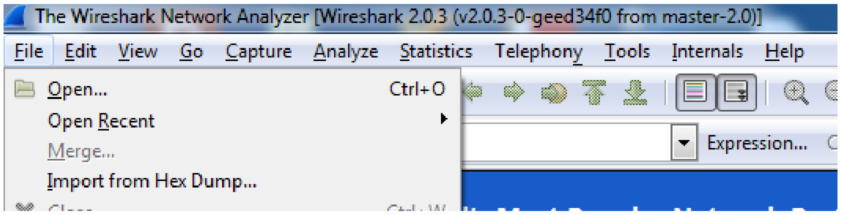


Fig. 1

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

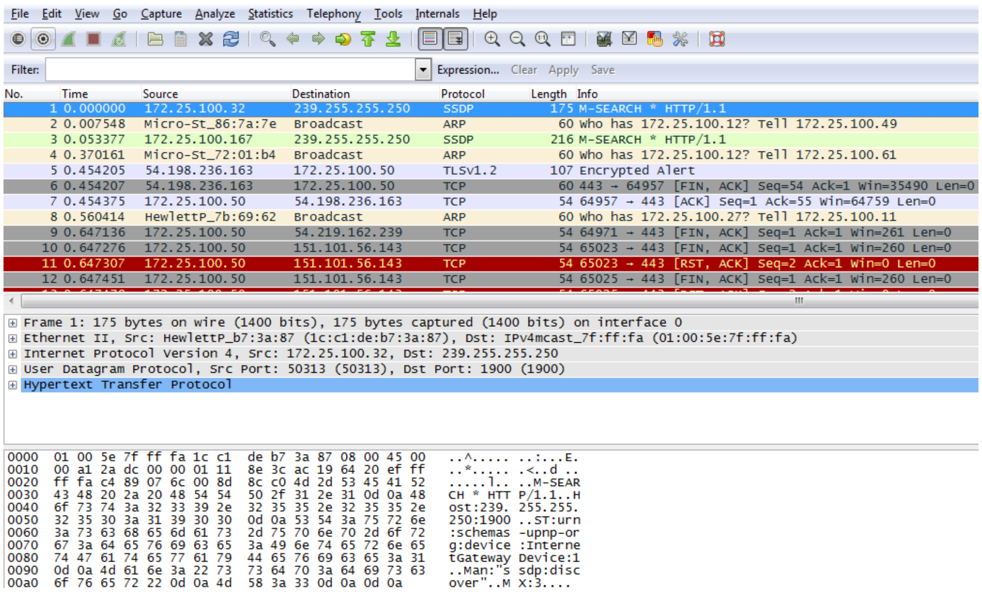


Fig. 2

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

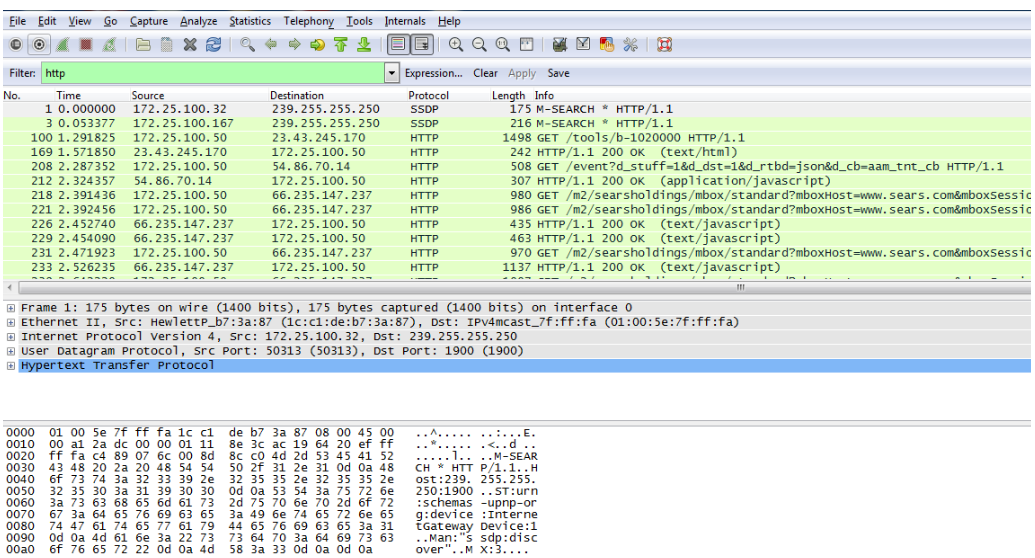


Fig. 3

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

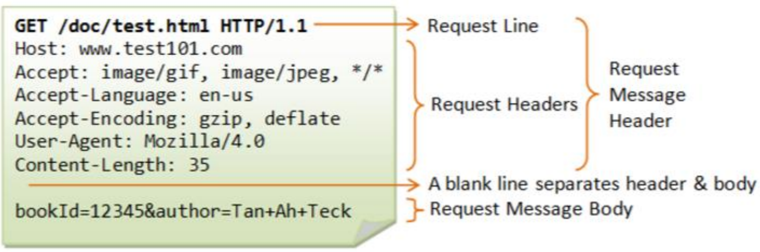


Fig. 4

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

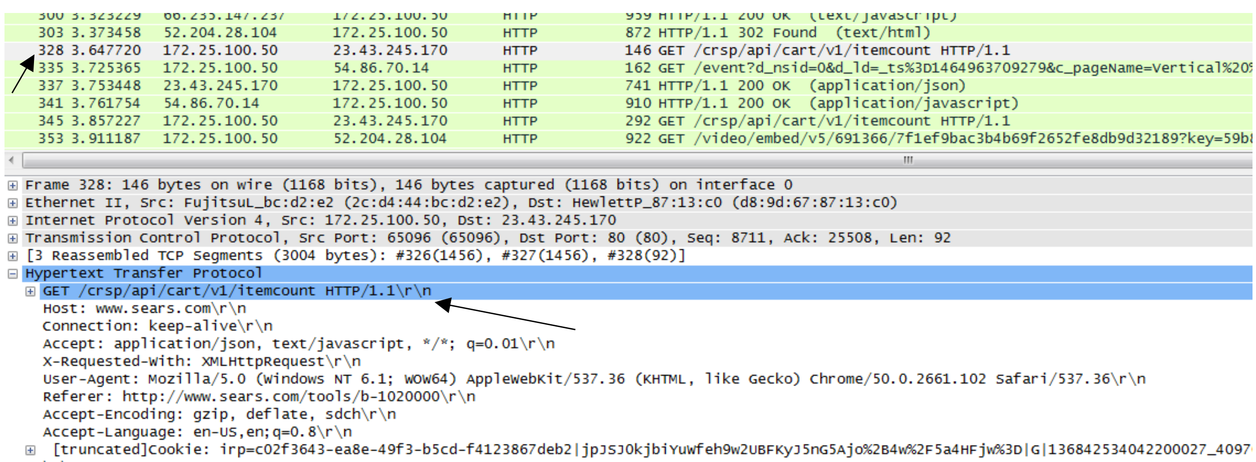


Fig. 5

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

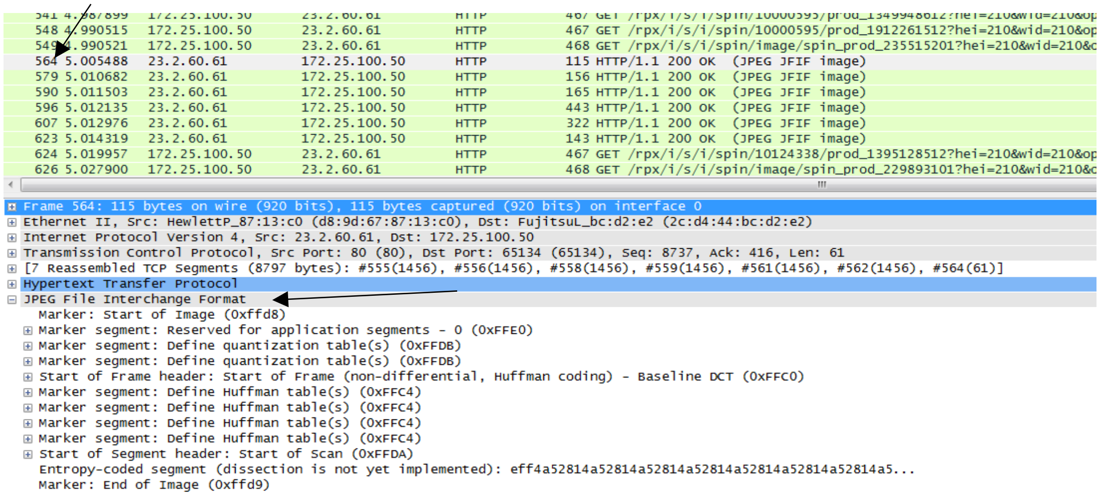


Fig 6

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.

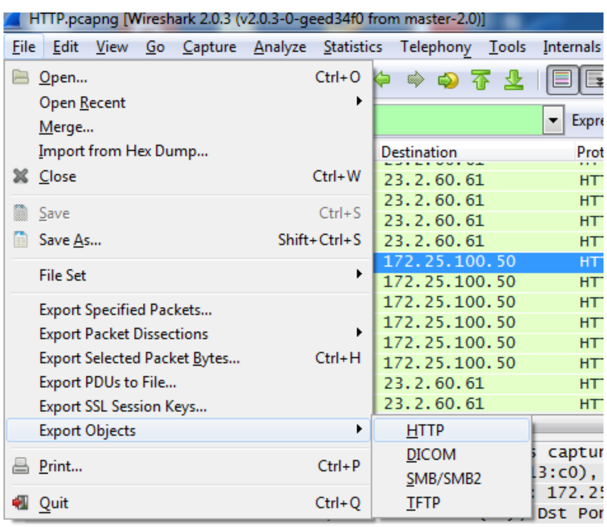


Fig. 7

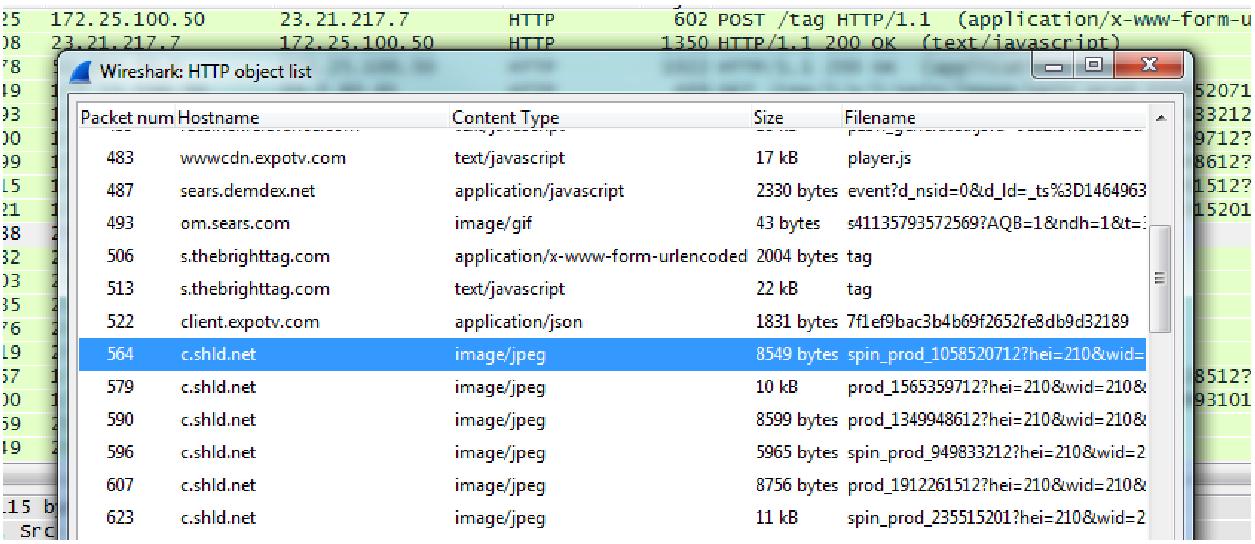


Fig. 7a

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

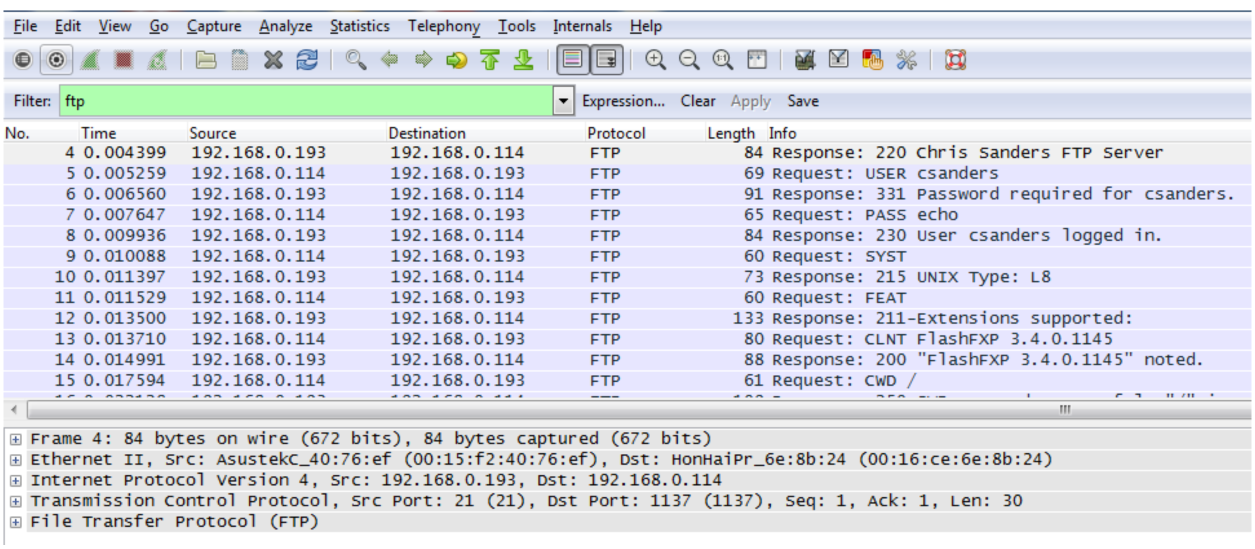


Fig. 8

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

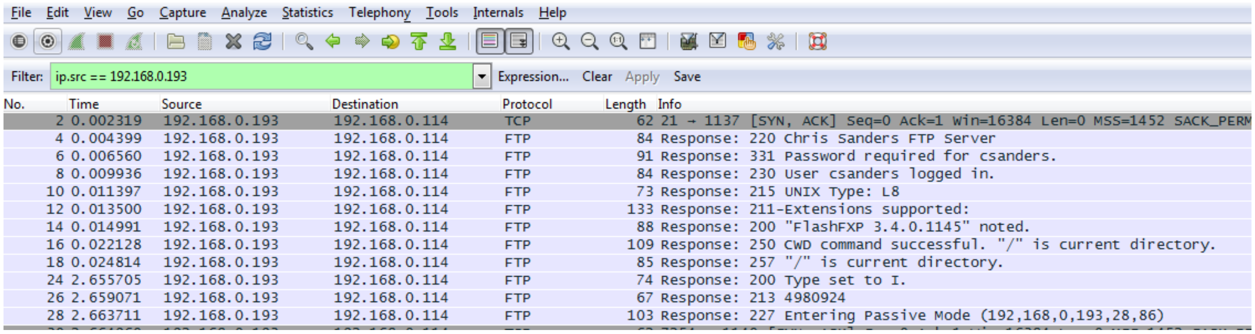


Fig. 9

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

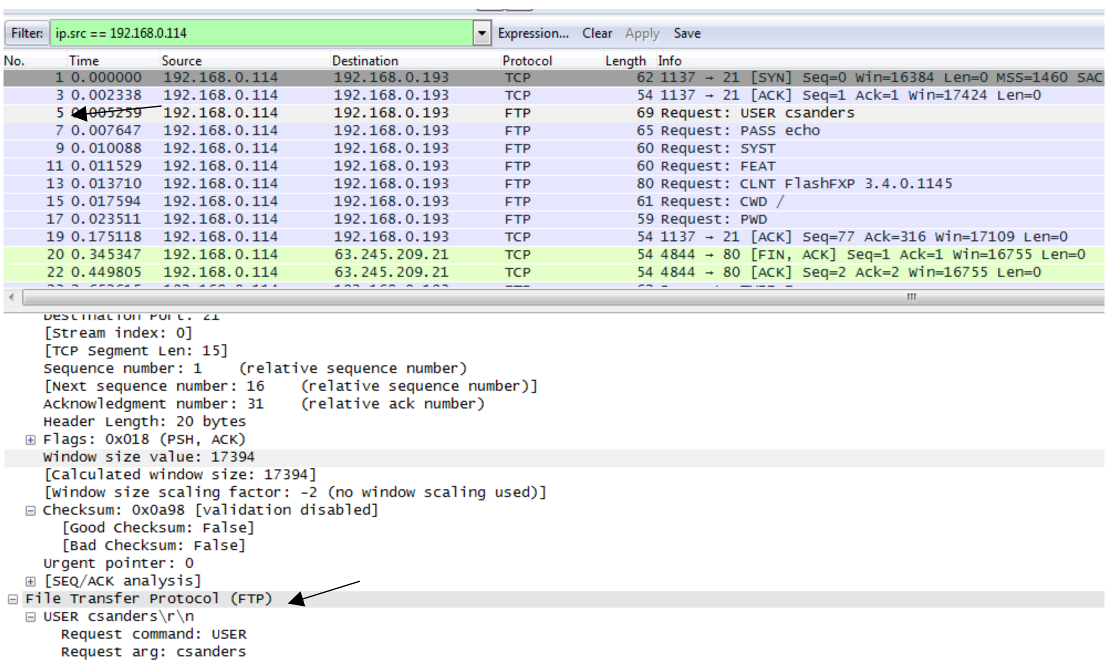


Fig. 10

- 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

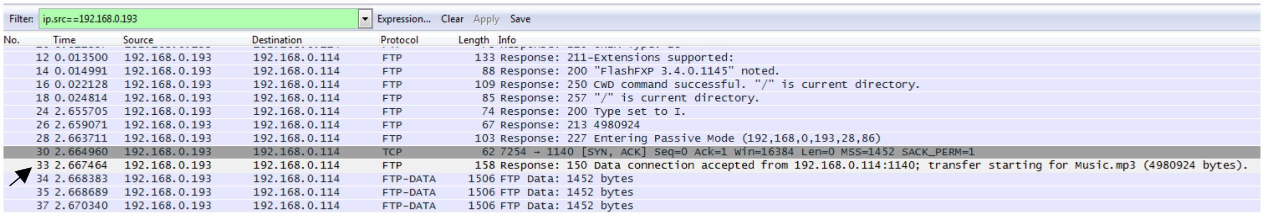


Fig. 11

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

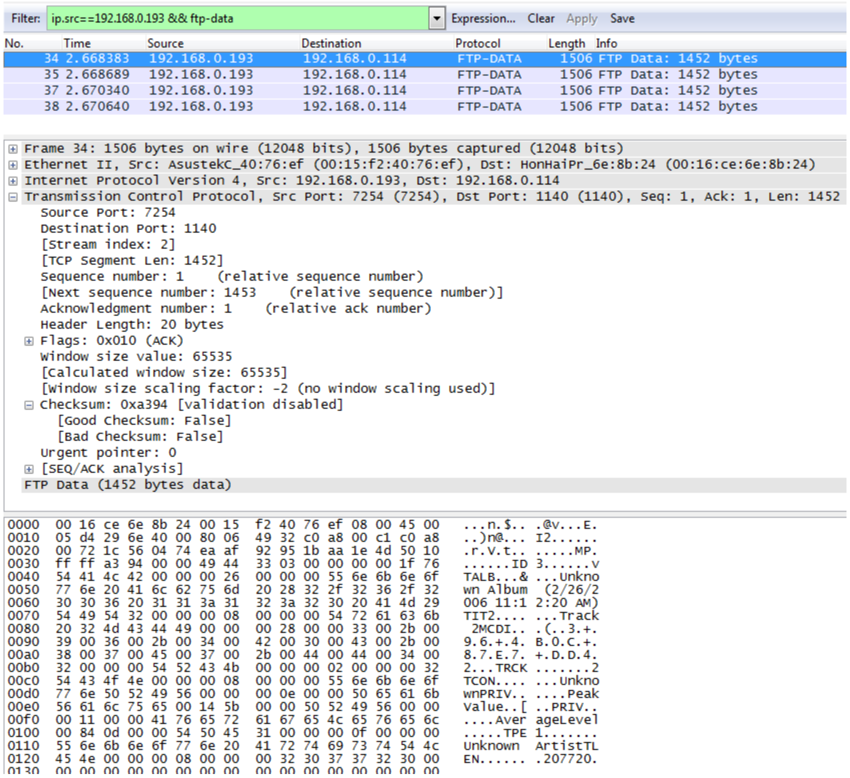


Fig. 12

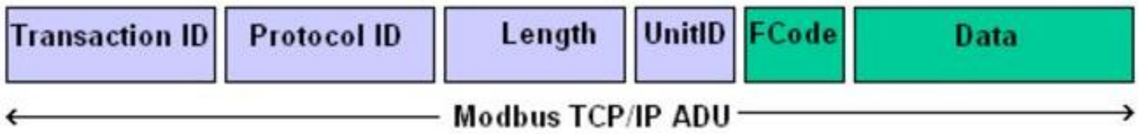
7. Close the FTP.pcap file.

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

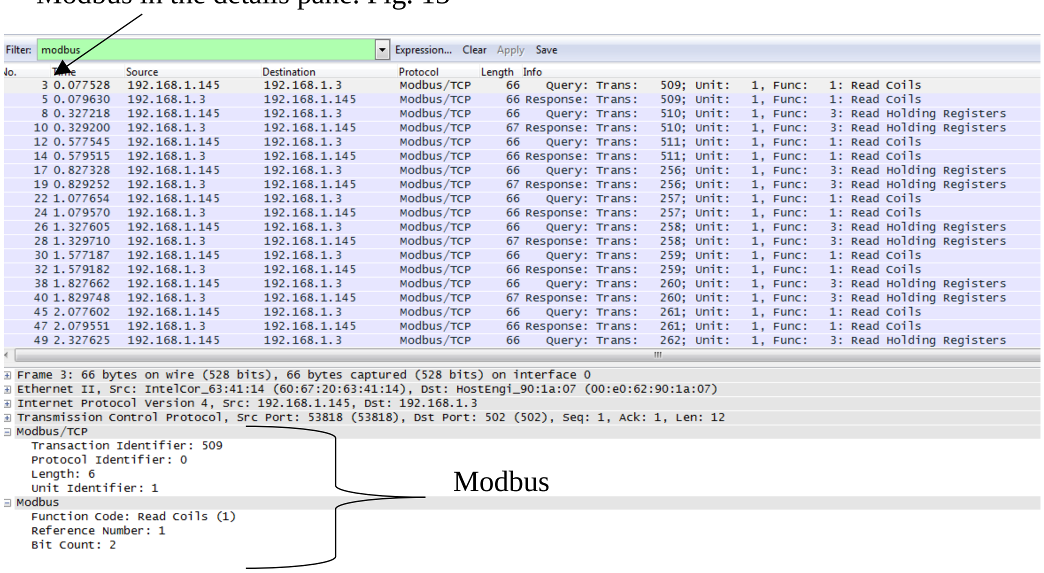


Fig. 13

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

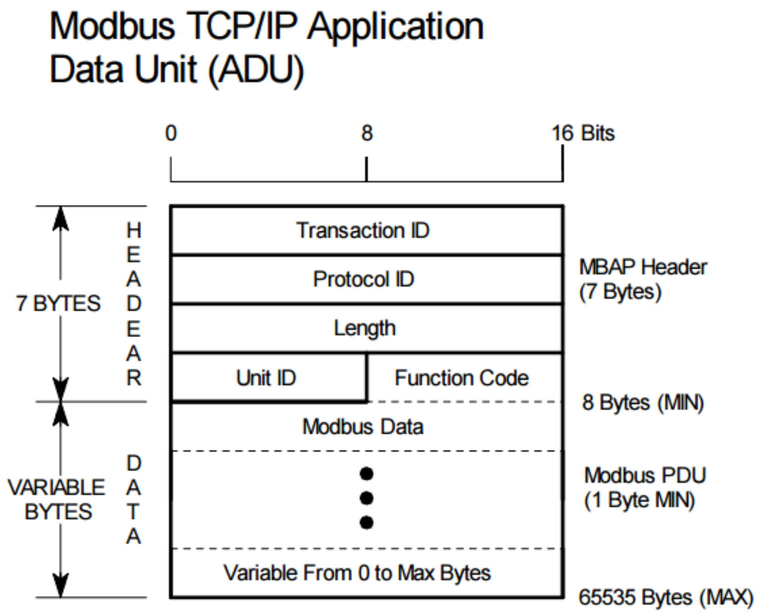


Fig. 14

4. 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

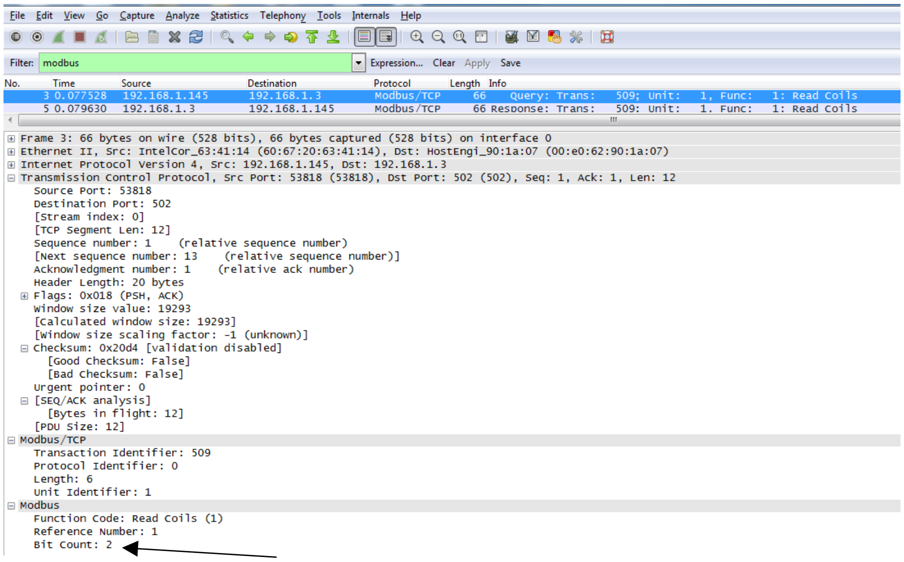


Fig 15

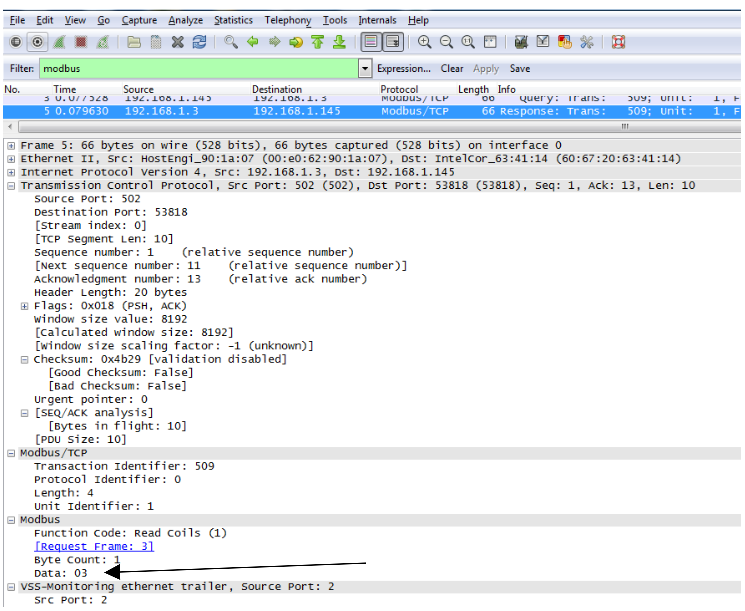


Fig 15a.

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