

Packet Analysis Fundamentals

Learning Outcomes

After completing these activities you should be able to:

- Analyze network traffic using a common network traffic analysis utility
- Capture network traffic using a common packet capture utility
- Explain the purpose of the Address Resolution Protocol (ARP)
- Install networking utilities for capturing and analyzing network traffic
- Install networking utilities to initiate a shell session on a remote host
- Install networking utilities to transfer files between hosts

Outside of Class

In preparation for and reinforcement of in class activities, complete the following activities:

- Print and bring the assignment to class for in class work
- Review the below information and documents
- Complete the activities included in *In Class*

Documents

Document	Version		Due Date
	Student	Post/Solution	
Assignment	student	solution	03 Oct 2019*
Packet Capture: <code>arpBootup.pcapng</code>	student	-	-
Laptop Software Installation	(See <i>Assignment</i> section below)	-	01 Oct 2019

*: You shall have the packet capture portion of the assignment complete before coming to class on 01 Oct 2019

- Read (Request for Comments)
 - RFC 826 (~10 pages)

In Class

Assignment

The software to capture and analyze network traffic has been installed on the lab machines already. Outside of class you will need to install the software on your laptop; you may install the analysis software under both Windows and your Linux VM. The software installation instructions are at the end of this section.

Today we will get an introduction to Wireshark, but first make sure you have downloaded the packet capture (pcap) file above, `arpBootup.pcapng`. The

note: `ng` In recent years the use of `ng` for utilities and file formats has risen. `ng` almost universally appears at the end of the

ng in `arpbootup.pcapng` stands for next generation, and has added features over the original pcap format.

command name or file extension. For commands a hyphen (-) may precede the ng, that is less common for file extensions.

Wireshark Introduction

Wireshark is commonly used to analyze network traffic; Wireshark is GUI tool that is available for Windows, macOS, Linux, and other operating systems. Additionally, Wireshark can capture network traffic. There are other tools that can capture and analyze network traffic, but Wireshark is the most widely-used network traffic analyzer (*Wireshark.org*).

Before we start Wireshark let's take a look at what Wireshark we will be using. It is important to understand what versions of a tool we are using. We need to know the features included in that version or not included in that version. Additionally, we need to be aware of the vulnerabilities associated with the tools that we use.

A basic question to ask about a program is where is the program in the file system; i.e. when I enter in a command, what binary file will be run? E.g. When we enter: `$ man ifconfig` where is `man` within the file system? The `which` utility can answer this question. The `PATH` environment variable is a list of directories that are searched through when a command is entered. The shell executes the first file whose name matches the command name from the command line, the shell searches through directories listed in the `PATH` environment variable in order.

Activity: `which` Craft

1. Read the following sections of the `which` man page: Name, Synopsis, Description
2. Enter: `$ which nmcli`
3. Enter: `$ ls -l /usr/bin | grep -B 5 -A 5 nmcli`
4. Think, Observe results

So `which` program we will execute, the next question is what version of the program is that program? There are two common ways you can ask a shell utility to report its version. Some command line utilities have a short option, `-v`, to output the version of that program. Some command line utilities have a long option, `--version`, to output the version of that program. Hmm, if only there was a way we could see which way, short or long option, a specific command uses to output its version information.



```
$ man <commandToMan> # < and > in a command line example indicate that you need to specify the associated argument value
```

In the example command line above you should change `<commandToMan>` with the name of the command you want to look up in the man pages. E.g. `$ man <commandToMan>` → `$ man nmcli`.

Activity: Version Control

1. Enter: `$ dig -v`
2. Think, Observe
3. Enter: `$ python3 --version`
4. Think, Observe

With this knowledge, complete Question 1 on the assignment.

Start Wireshark:

1. Start a shell session

in operations: Background Commands Backgrounding is a good way to start a GUI command while continuing to use the

2. Enter: `$ wireshark &`

same shell session.

The single `&` at the end of the command has

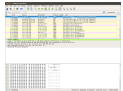
the shell start the command in the background; i.e. you are able to enter other shell commands in the shell session while the backgrounded command continues to execute.

3. You should see something similar to the below under Linux. The Windows, macOS, and Linux versions of Wireshark look and and behave similarly.



Wireshark Start Up (Ubuntu 16.04 LTS) (Hover over image to expand).

Use the *File* → *Open* menu item to open a packet capture file; open the `arpBootup.pcapng` file that you downloaded. You should now be in the main Wireshark view, similar to the below.



Wireshark Main View.

The main Wireshark view has three horizontal panes: packet list (top), packet details (middle), packet bytes (bottom). The packet details and packet bytes panes show information about the packet selected in the packet list pane.

packet list [Wireshark pane]

Lists the packets in the pcap in chronological order; the packet arrival time relative to the earliest packet in the pcap is displayed in the *Time* column.

Each row represents a packet; selecting a packet updates the packet details and packet bytes panes with the selected packet.

Columns represent information from the packet; columns to display may be selected from the various protocols that Wireshark can analyze.

packet details [Wireshark pane]

Displays information about the selected packet from the packet list pane.

Information is organized hierarchically with the lower (outer) layer protocols at the top, and upper (inner, encapsulated) layers at the bottom; Application layer protocols are at the bottom of the packet details pane.

The protocols in the packet can be selected and expanded/compressed to show/hide packet details.

The layout in the packet details pane can be thought of as a visual representation of data encapsulation.

packet bytes [Wireshark pane]

Displays bytes of the selected packet from the packet list pane.

The far left column shows the byte offsets for the bytes in the packet.

The middle section displays the hexadecimal representation of the bytes in two groupings.

The right section displays the ASCII text representation of the bytes in two groupings.

The bytes associated of the selected (from packet details pane) protocol (light gray) and protocol field (blue) are highlighted.

Curiosity may kill the cat, but exploration and experimentation are valid learning techniques. Don't be shy about exploring features of tools beyond what we introduce in this course. You should read the documentation before you blindly take an action, but in general explore the tools. Don't worry we will let you know if a command or action may cause system configuration or stability issues. You can break network settings with `ifconfig` or `nmccli`.

You could fill up a disk drive by leaving Wireshark in capture mode. Wireshark can take a long time to process large pcaps (50+ MB in size). You need to know the capabilities and limitations of your tools.

Activity: Know Your Tools (Try It!)

- Take a few minutes exploring the pcap and Wireshark

in the Fleet: Capabilities and Limitations Just like other war fighting domains, tools (weapons) in the cyber domain have capabilities and limitations. In order to be effective in any domain you need to know the capabilities and limitations of your own systems as well as those of the adversary. The common military terminology for Capabilities and Limitations is *Caps and Lims* (written), caps is pronounced *capes*.

- Select packets from the *packet list*, observe how the other two panes are affected by your packet selection
- Expand the packet details and select various protocols and protocol fields within a packet in the *packet details* pane, observe how the packet bytes pane is affected
- Move your mouse over and around, and select bytes in the *packet bytes* pane, observe how the packet details pane is affected

Answer Questions 2–4 on the assignment.

Like most GUI and many CLI utilities you can customize Wireshark to suite your preferences. Complete the below activity as an introductory exploration of the Wireshark preferences, and to setup Wireshark for activities.

Activity: Poker Face — Don't Tip Your Hand

1. Open the *Wireshark Preferences* window: Edit → Preferences



Wireshark Preferences window.

2. Make note of the information displayed in the *Source* and *Destination* columns
3. Select *Name Resolution* in the tree view
4. Uncheck (de-select) the following check boxes as necessary:
 - Resolve MAC addresses (relevant to Question 5)
 - Resolve transport names
 - Resolve network (IP) addresses
 - Use captured DNS packet data for address resolution
 - Use an external network name resolver
 - Only use the profile "hosts" file
 - Resolve VLAN IDs
 - Enable OID resolution
 - Suppress SMI errors
5. Click *OK*
6. Observe the information displayed in the *Source* and *Destination* columns

Answer Question 5 on the assignment; toggle the associated flag as necessary. When you are finished leave the *Resolve MAC addresses* setting disabled.

Activity: If it's in the game, it's in the game

1. Reopen the *Wireshark Preferences* window
2. Select and expand *Protocols* in the tree view
3. Take a few minutes to look through the list of protocols that Wireshark interprets
4. When you are done close the preferences without making any additional changes

Wireshark is an interpreter of network protocols. Wireshark makes use of an interpreter language, each protocol that Wireshark can decode has an associated *dissector*. A dissector is the description of the protocol in the interpreter language of Wireshark. This modular approach to Wireshark's design has many benefits.

One thing about interpretation is that it takes time. Wireshark can do a lot, and if you ask it to (by loading a large pcap) it will try to do a lot.

Activity: Enough is Enough

1. Scroll and select one of the ARP packets in the *packet list* pane.
2. Open the *Wireshark Enabled Protocols* window: Analyze → Enabled Protocols...



Wireshark Enabled Protocols window.

3. Reposition the *Wireshark Enabled Protocols* so that you can see the packet list and packet details panes
4. Scroll and de-select the *ARP/RARP* protocol
5. Click *OK*
6. Observe the information in the main Wireshark view

Answer Question 6 on the assignment. When you are finished leave the *ARP/RARP* dissector enabled.

The following activity will have you configure the *packet list* pane in the main Wireshark view. You have some freedom to setup the display as you see fit. The only two requirements are that you display the minimum columns specified and keep the minimum columns in the same order. You are free to title the columns as you see fit, and to add additional columns as you desire. You will be able to save these settings, can easily switch between multiple saved settings. You are encouraged to explore and change settings as we go through the various analysis activities.

Instructor Preference. The following configuration is my (LCDR Hoffmeister's) personal preference, there is no expectation that you use the below configuration.



The rationale behind the below configuration is that there are many tools used in cyber operations. Many tools have their own mini scripting language. Some tools within the same area of the domain share some common mini scripting language components. Many network traffic analysis tools support Berkley Packet Filters (BPF); Wireshark's filter language is an object oriented like BPF. In the below configuration column names are based on the Wireshark filter Language where possible; the ordering is based on the order in which the bytes appear in the packet, i.e. the order in which the bits appear on the wire (with the exception of the Ethernet addresses). Additionally, the below configuration reinforces the layers of the TCP/IP Stack.

LCDR Hoffmeister's Personal
Wireshark Preferences

Title	Type [→ Fields]
no.	Number
time	Time (format as specified)
eth.src	Custom → eth.src
eth.dst	Custom → eth.dst
ip.src	Custom → ip.src
ip.dst	Custom → ip.dst

Title	Type [→ Fields]
port.src	Source port
port.dst	Destination port
proto	Protocol
info	Information

Activity: It's Your Jeep

1. Setup a new configuration profile:

1. Open the *Wireshark Configuration Profiles* window: Edit → Configuration Profiles...
2. Click the + button to create a new configuration profile
3. Give the new configuration profile a descriptive name (course, your name/tag, etc.)
4. Click *OK*

2. Return to the Wireshark Preferences window

3. Expand the *Appearances* branch

4. Select *Columns*

5. Review the below interface instructions:

- Use + to add a column
 - Use - to delete a column
 - You may drag columns to change ordering
- If the preferences window hangs, press *Alt-F4* to close the preferences and try again/another way
- Double click a cell in a column to edit the value (text entry, list selection)

6. Configure your main Wireshark view according to the following:

SY205 Minimum
Wireshark Columns

Type [→ Fields]
Number
Time (format as specified)
Custom → eth.src
Custom → eth.dst
Custom → ip.src
Custom → ip.dst
Source port

Type [→ Fields]

Destination port

Protocol

Information

Packet Filtering and Analysis

You might be starting to realize that a lot of packets go across networks. Hopefully the previous assignments opened your eyes to the number of packets a single simple message can generate. And your quick perusal of the list of protocols Wireshark knows about is quite long. A lot of protocols compounded with potentially a lot of packets per protocol can lead to large pcaps.

note: Capture vs Collect The verbs capture and collect have very similar meanings in the English language. The term *capture* is often used when referring to data in transit; e.g. we capture network traffic. Where as, the term *collect* is often used when referring to data at rest; e.g. we collect digital forensics.

When capturing packets we typically want to capture all packets, and would like to capture the complete packet, this is known as *full packet* capture. Full packet capture may not be realistic, for a number of technical or ethical reasons. Over time we may not desire to retain full packet capture files.

Additionally, in live operations we often focus on common sets of protocols based on the situation. This is all to say that we would like a way to filter packets for analysis, and save those filtered packets independent of the original pcap. Wireshark allows us to create display filters, and save packets based on filter results.

Instead of an address bar, Wireshark has a *filter* bar. The filter bar is just below the tool bar, directly above the packet list pane; it says *Apply a display filter ...* as background text. The display filter turns red when there is a syntax error with the filter, and turns green when a syntactically correct filter is entered. Pressing *Enter* will apply the syntactically correct display filter.

Activity: Display Filter

1. Make sure you have the original `arpBootup.pcapng` pcap opened in Wireshark
2. Enter the following display filter: `arp`
Like many languages, case matters in display filters
3. Observe the results of applying the filter

The filter applied resulted in only Address Resolution Protocol (ARP) packets being displayed. Complete the following activity with the ARP filter still active.

Activity: These are the Packets You're Looking For

1. With the ARP filter still active
2. Export the filtered packets: File → Export Specified Packets



Wireshark Export Specified Packets dialog.

3. Select the following *Packet Range* options:

note: Stay Organized We will be creating and analyzing a lot of pcaps. Many assignments will span multiple course calendar days. We will analyze a given pcap multiple times; i.e. the pcaps you create for one activity or assignment may be used again for a another activity or assignment. You will share your pcaps with other classmates, and analyze pcaps shared by other classmates. Keep your pcaps organized; the manner in which you keep your packets organized locally will be up to you. There will be a course wide Google Drive shared directory to facilitate pcap sharing between students and instructors.

- Displayed
- All packets

4. For *File name* enter: `arp-only.pcapng`

5. Click *Save*

Complete Question 7 on the assignment.

Reopen `arpbootup.pcapng` before continuing to the next section.

Packets sent across a network do not contain timing information themselves; i.e. the sending host does not add a time stamp to the packet when it is sent.

note: metadata Metadata is data about data. For example, in a file system the last time the file was modified is metadata.

But time information is important to network architects, network operators, and to network traffic analysis for security purposes. Like most information from information systems, pcaps can be used for multiple purposes. Packet capturing tools add time stamps and other packet metadata to pcaps. The capturing tool gets the metadata from the network interface card, operating system, user when capture process is started or during analysis, or calculates some metadata fields.

Metadata in pcaps comes from the host conducting the packet capture, or from the packet analyst. The bits of the packet sent across the data are not considered packet metadata, the bits are the actual data transmitted between communicating hosts. Packet metadata is part of the pcap file format, each packet can have metadata associated with it. Wireshark displays the packet metadata.

Activity: Packet Metadata

1. Select Packet 11 in the packet list pane; expand the *Frame* section in the packet details pane

2. Take a minute to look at the type of metadata associated with a packet

All time standards have an *epoch*, an epoch is a reference date/time from which all other times are relative to; i.e. an epoch is the 0 on a timeline.

POSIX time (a.k.a. UNIX epoch) has an epoch of 0000 1 Jan 1970 UTC; networking systems, and thus pcaps, use POSIX time for epoch.

3. Collapse the *Frame*, metadata, section in the packet details pane

Currently the Ethernet II is the most common version of the Ethernet protocol, Ethernet is a Data Link layer wired network protocol. In SY205, we will simply say Ethernet when referring to Ethernet II. The success of Ethernet led to later Data Link layer protocols to adapt Ethernet features, specifically MAC addresses. You have probably heard of, and used WiFi and Bluetooth; WiFi and Bluetooth both use the same size MAC address (48 bits) used in the Ethernet protocol (if it's not broke, don't fix it).

As mentioned earlier the sections in the packet details pane are arranged to represent the data encapsulation that occurs in a packet as the data flows down the network stack from the Application layer at the top.

Activity: Filter from Packet Data

1. Ensure Packet 11 is still selected


2. Expand the Ethernet frame information in the packet details pane

3. Right-click *Destination: ff...* in the packet details pane

4. Select *Prepare a filter* → *Selected*

5. DO NOT apply the filter yet

6. Answer Question 8 on the assignment

7. Apply the generated filter by pressing *Enter* in the filter bar, or clicking the  to the right of the filter bar

8. Answer Question 9 on the assignment

Hint: It's called a *status bar* for a reason

Protocol addresses with all 1's are called *broadcast* addresses. Broadcast addresses are received by any host within the *broadcast domain*. We will explore what a broadcast domain is in the *Networking Infrastructure* material.

Packets destined for broadcast addresses are usually the result of the sending host asking a question. E.g. I need an IP address, can someone give me one? Can someone tell me what the MAC address is of the host with IP address X.X.X.X?

Generally, if a receiving host cannot answer the question, then they simply ignore the request. That is, if the receiving host does not know the answer, then they do not send a reply back to the sender of the broadcast message.

With this knowledge, complete Question 10 on the assignment.

Address Resolution Protocol

Address Resolution Protocol (ARP)

Data Link layer protocol used to convert between MAC addresses at the Data Link layer and IP (Network) addresses at the Network layer.

ARP is a Data Link layer and Network layer agnostic protocol; i.e. ARP is designed to convert between arbitrary sized addresses, not just 6 byte Ethernet MAC addresses and 4 byte IPv4 addresses.

A host sends an ARP request when it knows the IP address of a remote host and knows that the remote host is on the same local (Data Link layer) network, but does not know the remote hosts MAC address.

Wireshark allows the analyst to build complex filters, we can use logical boolean operators to build complex display filters; e.g. display packets of Protocol X that include Host A. Wireshark uses logical operators that are found in lower level and higher level programming languages, namely C, C++, Python. The following table lists the comparison and the logical boolean operators for Wireshark display filters.

Wireshark Display Filters Logical Boolean Operators

Operator Precedence	Boolean Operation	Wireshark Operator	Python 3 Operator	C, C++ Operator
Highest	Comparisons	(eq , ==),	== ,	== ,
		(ne , !=),	!= ,	!= ,
		(lt , <),	< ,	< ,
		(le , <=),	<= ,	<= ,
		(gt , >),	> ,	> ,
		(ge , >=)	>=	>=
-	Not	not , !	not	!
	And	and , &&	and	&&
Lowest	Or	or ,	or	

Activity: Do you know ...?

1. With the Destination filter still applied, select one of the ARP packets in the *packet list* pane
2. In the *packet details* pane right-click the ARP header

3. Select *Apply as Filter* → ...and *Selected*

4. There should be four packets remaining in the packet list

5. Select the last packet and review the details and bytes of the packet

6. Read RFC 826: An Ethernet Address Resolution Protocol (~10 pages)

Yes, it is a little on the long side, you don't need to commit the protocol field sizes and implementation details to memory. But you should read through the entire RFC to start to get a feel for RFCs are written.

7. Complete Question 11 on the assignment

Activity: I Know Hex and Human

1. Clear the display filter, click the *x* button on the right side of the filter bar

2. Enter the following display filter: `eth.type == 0x0800`

3. Think/Discuss, Observe

4. Complete Question 12 on the assignment

5. Clear the display filter

6. Create a filter that meets the following criteria:

- Source IP address (`ip.src`) is `24.6.170.101` ; and
- Destination MAC address is not `ff:ff:ff:ff:ff:ff`

7. Complete Question 13 on the assignment

Besides protocol analysis Wireshark can calculate packet statistics. Statistics can be a powerful analysis tool. But statistics are not a cure all. Statistics can and should be used to highlight situations that warrant further analysis, but statistics can be just as easily misused to mask situations. If you are presented with statistics, you need to be clear on what and how the data was analyzed to calculate the statistics.

Activity: Cleaned Up Bludhaven Using Statistics

1. Clear the display filter

2. Explore the first six statistical reports in the *Statistics* menu

3. Answer Question 14 on the assignment

Packet Capturing

You will need to install software in your VM before you can complete this section of the assignment. The instructions for installing the required software are discussed at the end of the assignment. The lab machines in MI300 already have the software installed, and your account already has the appropriate permissions on the lab machines.

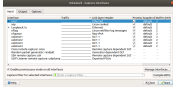
Wireshark can be used to capture packets. The packets captured can be filtered in a similar way as the display filters. A fundamental capturing option is whether the interface is placed in *promiscuous* mode or not. Promiscuous mode allows the interface to pass all traffic that comes across the interface to the operating system, regardless of whether the traffic is addressed to that specific interface or not. Without promiscuous mode enabled the interface will only pass the traffic addressed to the interface, normal networking operation.

Unless otherwise specified we will be capturing packets in promiscuous mode.

The next activities will have you capture packets using Wireshark. If you are unable to capture packets, then contact your instructor.

Activity: Badge Earned — First Packet Captured

1. Open the Wireshark Capture Interfaces window, *Capture* → *Options*



Wireshark Capture Interfaces window.

2. Explore the three tabs in the Capture Interfaces window: *Input*, *Output*, *Options*

3. Go to the *Input* tab, and select your Ethernet interface

4. Click the *Start* button

5. You should start seeing packets populate the main Wireshark view

6. Explore the packet capture actions on the tool bar:

Perform each action a handful of times, no need to save a pcap.

- Start (Blue Fin button)
- Stop (Red Square button)
- Restart (Green Fin button)

There is no pause button, *Restart* restarts the capture using the current settings.

7. Once you are done exploring stop capturing packets, no need to save a pcap

8. Before you go to the next activity, ask yourself if you are surprised by how many packets you captured in that short amount of time

Activity: Looking Behind the Curtain

1. You will be saving a pcap at the end of this activity

2. Start a new packet capture session on the Ethernet interface

3. Enter the following display filter: `http`

4. Open a new tab in your web browser, and navigate to: `http://courses.cyber.usna.edu/SY205/`

The use of HTTP vice HTTPS

5. With the capture still in progress, and within 3 minutes of navigating to

`http://courses.cyber.usna.edu/SY205/` complete the following in a shell from the same host you are capturing packets on

1. Navigate to the directory that you will save pcap file to (future step)

2. Enter: `$ arp -n -i <ETHER_INTERFACE> > httpHostARPCache.txt`

Replace `<ETHER_INTERFACE>` with the name of the Ethernet interface that you are capturing packets on; e.g.

`enp0s25` sans (without) `<>`.

3. Explore the contents of `httpHostARPCache.txt` using `less`

6. Think/Discuss, Observe

7. Stop the capture about a minute (or so) after the website loaded

8. Complete Question 15.a.–15.b. on the assignment

If Wireshark stopped at analyzing individual packets that would be great, and we would still be using it. But Wireshark goes beyond an individual packet, after all you've already determined that large messages are fragmented into smaller messages and sent in individual packets. Wireshark can follow communication streams; i.e. Wireshark can comprise the upper layer protocol messages into a single communication, leaving out the details of the underlying lower layer protocols.

Activity: Streamer

1. With the HTTP display filter applied, right-click on the first *HTTP* (not TCP) packet in the packet list

2. Select *Follow* → *TCP Stream*

3. Read through the TCP Stream (Think/Discuss, Observe):

The messages are color coded red and blue based on sender.

- Information about the requester (client)
- Protocol version and encodings supported
- Encoded content
- Explore the *Conversation* and *Show ... data as* drop downs

Student Question: What compression was used to send the content?

4. Click the *Close* button

5. Right-click on the same HTTP packet in the packet list

6. Select *Follow* → *HTTP Stream*

7. Click the *Close* button

8. Clear the display filter

9. Export the captured packets (*File* → *Export Specified Packets...*), save the pcap as: *http.pcapng*

Wireshark has a command line companion called `tshark`. Wireshark can bog down on large pcaps due to the large amount of processing, specifically GUI related, Wireshark is doing. `tshark` is a great companion to Wireshark because we can use `tshark` to filter packet data for analysis with tools outside of Wireshark.

Take a few minutes and read the Name, Synopsis, and Description sections of the `tshark` man page.

The first feature of `tshark` we will explore is the *read filter*. The read filters in `tshark` are conveniently the same as the display filters in Wireshark. That means if you are having trouble writing a valid `tshark` read filter, you can draft a display filter in Wireshark and make use of Wireshark's GUI coloring (red vs. green). The basic syntax for using read filters in `tshark` is below.

```
$ tshark -r <PCAP_FILENAME> -Y "<READ FILTER>" -T fields -e <FIELD_NAME> -e <FIELD_
```

`-r <PCAP_FILENAME>` : Means read from `<PCAP_FILENAME>`; replace `<PCAP_FILENAME>` with the name of the pcap file sans (without) `<>`.

`-Y "<READ FILTER>"` : Means apply `<READ FILTER>` to the packets.

`-T fields` : Sets the text output format for the packets, in this case output selected fields.

`-e <FIELD_NAME>` : Wireshark field specification syntax; e.g. `eth.dst`.

Activity: If only I had a shell (Part 1)

1. Start a shell session, and navigate to the directory that you saved `http.pcapng` in

2. Enter: `$ tshark -r http.pcapng -Y "http" -T fields -e eth.src -e eth.dst`

3. Think/Discuss, Observe

You should identify the output as the list of source and destination MAC addresses from the HTTP packets from `http.pcapng`. This is good but we would like to see the unique pairings of interfaces, much like we saw the communications pairings in Wireshark. Well, there are some UNIX CLI utilities that can help with this: `sort` and `uniq`.

Your intuition is serving you correct. `sort` lexically sorts the input text (see `$ man sort`). `uniq` omits consecutive, repeated lines from text input (see `uniq` man page). Each is usefully in their own right, but combined they can be even better.

With this knowledge, complete Questions 16–17 on the assignment.

Wireshark and `tshark` have a wide range of filters that are available due to the manner in which the protocol dissectors are designed and implemented. Regarding filters, the protocol and its fields can be specified in a dotted syntax; e.g. `eth.src`, `eth.dst`, `ip.src`, `port.src`, `http.host`. We will continue to explore more protocols and fields for use in filtering in support of analysis. This simple syntax allows the analyst to create arbitrary filters for packet analysis.

In practice an analyst can use Wireshark interactively to develop and test an analysis process. The resulting process can be automated and connected to other programs for external (to Wireshark) correlation and analysis using `tshark`.

With this knowledge, complete the remainder of the assignment.

Software Installation

Data collection and data analysis are often separated in cyber domain operations, network traffic analysis is no different. This is to say that data across systems operating in the cyber domain are collected, often times aggregated, from one system and analysis is performed on another system. There are security and other operational reasons for separating data collection and data analysis.

There are a number of different networking utilities that are commonly used to capture and analyze network traffic. Will we primarily use the Wireshark suite of tools to capture and analyze network traffic. The lab machines in MI300 already have the requisite software installed to analyze network traffic.

The below is a rough draft of the software installation instructions, more polished instructors will appear later.

Install the following software:

Software Installation Requirements

Software	Windows	Ubuntu VM
PuTTY	via USNA Windows Software Center	-*
WinSCP	via USNA Windows Software Center	-*
Wireshark	Windows 64-bit	via Distribution Package Manager

* Client software already installed as part of base operating system.

Ubuntu VM Software Install Instructions

Follow the below instructions to install Wireshark on Ubuntu:

1. Save your work in other applications, and exit other applications (you will reboot your VM during the installation process)

note: You Said No, Needed to Say Yes So, when you read the dialog about allowing normal users to capture packets is a security risk,

2. Start a shell session

you said *No* do not allow normal users to capture packets via Wireshark. [...]

3. Enter:

```
$ sudo apt install wireshark wireshark-doc tshark
```

Switch user do (switch to *root*), have `apt` install `wireshark`, `wireshark-doc`, `tshark`. `apt` is the package manager (similar to Windows Software Center) for Debian/Ubuntu. `tshark` is a command line companion to `wireshark`.

4. Enter: *Y* to install packages

5. You will be prompted about how to configure `wireshark`, a Yes/No dialog. Enter *YES* to allow all users to capture packets.

6. Enter: `$ sudo usermod -a -G wireshark <yourUsername>`

This adds you to the *wireshark* group.

7. Enter: `$ sudo apt remove appmenu-qt5`

Switch user do, have `apt` remove (uninstall) `appmenu-qt5`; there are integration issues between `appmenu-qt5` and `wireshark`.

8. Enter: `$ sudo shutdown -r now`

Switch user do, restart the system now.

9. Log back in, and start a shell session

10. Enter: `$ groups`

You should see *wireshark* listed in the output. If you do not see your instructor (bring your laptop).

11. Enter: `$ wireshark --version`

You should see the version information from `wireshark`

12. Enter: `$ tshark --version`

You should see the version information from `tshark`