Lab 10 – DNS Filters and Special Ops

Author: Raymond Ng
Course Number/Section: IS 3413-006
Date: November 15, 2022

INTRODUCTION

The purpose of this lab was to allow the user to experiment with DNS filters and special operators used in Wireshark. Moreover, it allows the user to become more familiarized with the use of Wireshark.

PROCESS

I will be using the `PCAP` file, `THMDNS.pcapng`, provided by *TryHackMe.com* for this section of the lab in Wireshark. [1]

1. The address of the DNS server is `192.168.4.1`. To find the answer, I selected the first packet of the `PCAP` file and looked for the `Destination Address` in the Packet Details Pane via Wireshark (*Figure 1*).

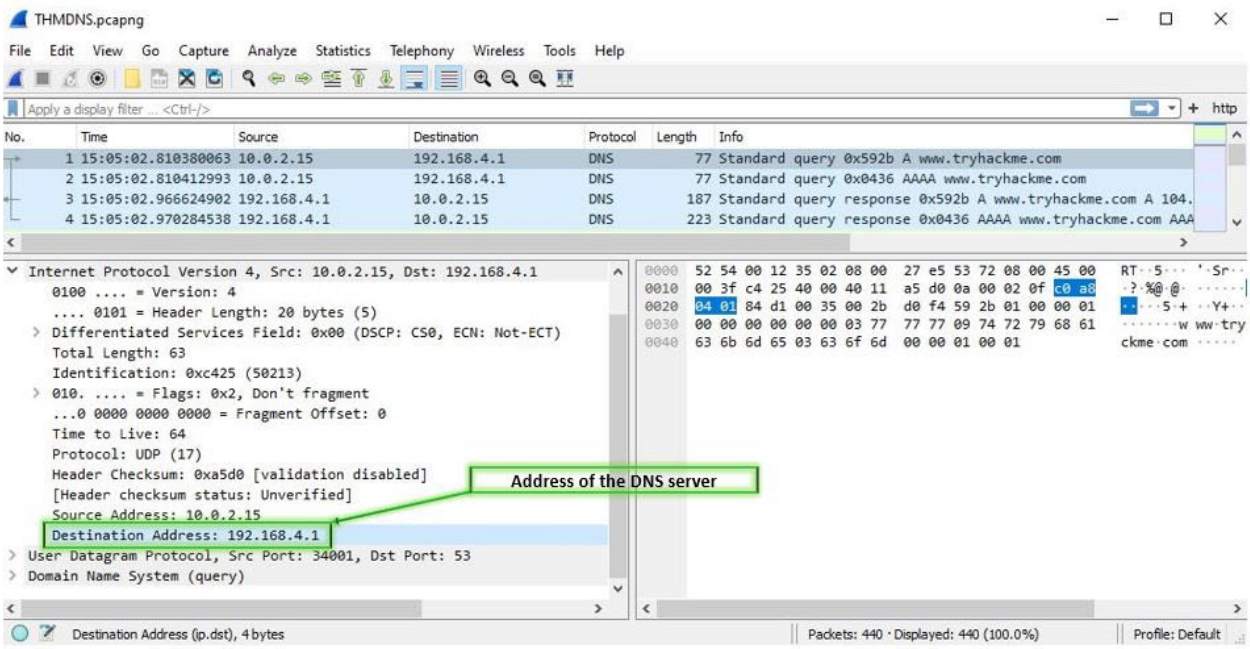

Figure 1: Address of the DNS server via Wireshark

2. Using `dns.flags.response == 0`, setting a filter for DNS queries, returned `44` packets (*Figure 2*).

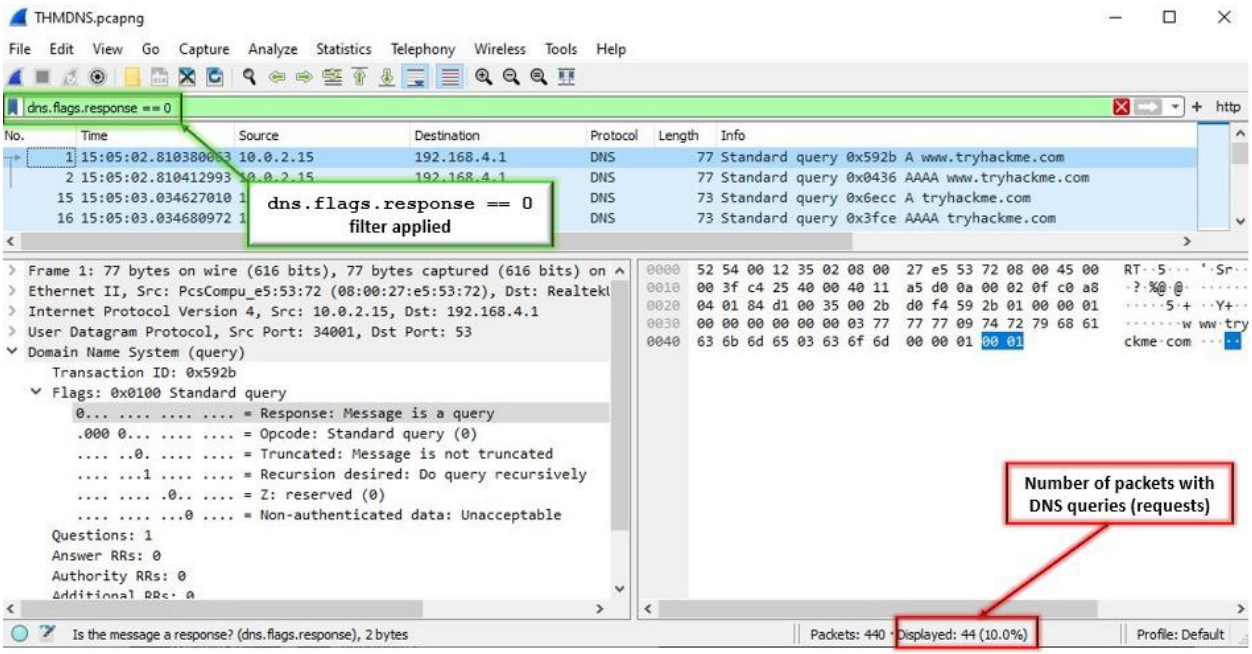

Figure 2: Number of packets with DNS queries (requests) via Wireshark

3. Using `dns.flags.response == 1`, setting a filter for DNS responses, returned `44` packets (*Figure 3*). I simply changed the `0` from the previous filter used to a `1`. In binary, `0`'s are requests and `1`'s are responses. Since I had 44 queries from the previous task and discovered 44 responses on this task, I know that every request had a response.

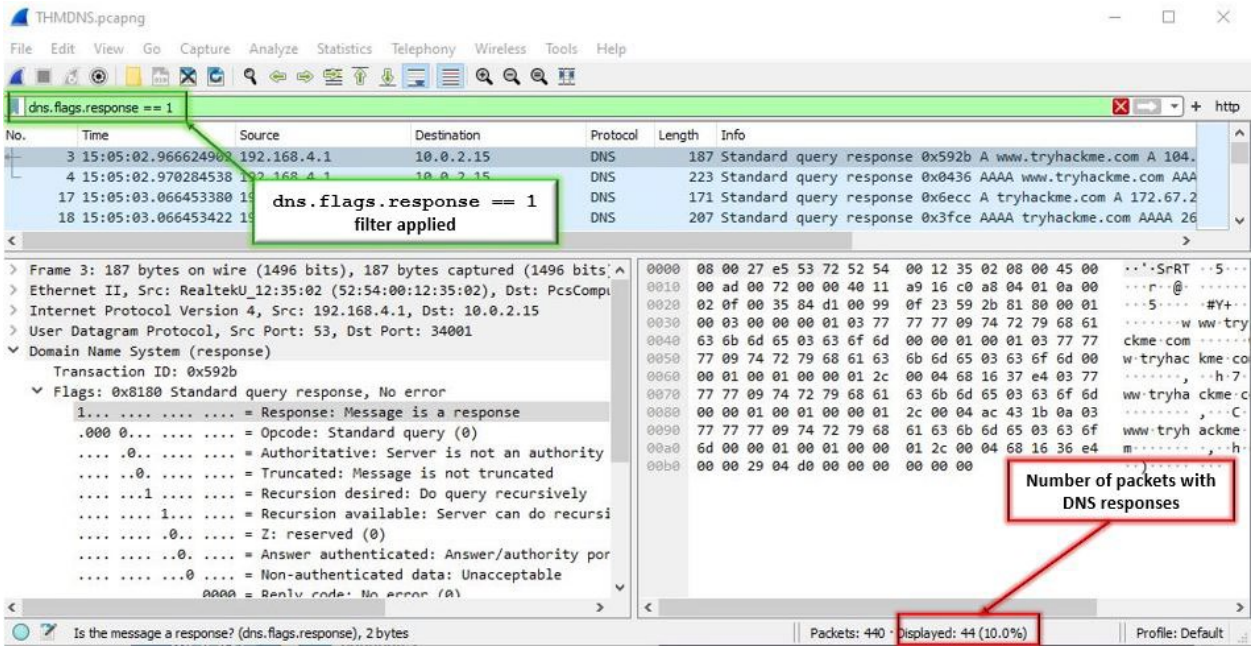

Figure 3: Number of packets with DNS responses via Wireshark

4. Using `dns.qry.name contains hack` filter, I discovered `12` packets that contain the word "hack" in the DNS query name, both requests and responses (*Figure 4*).

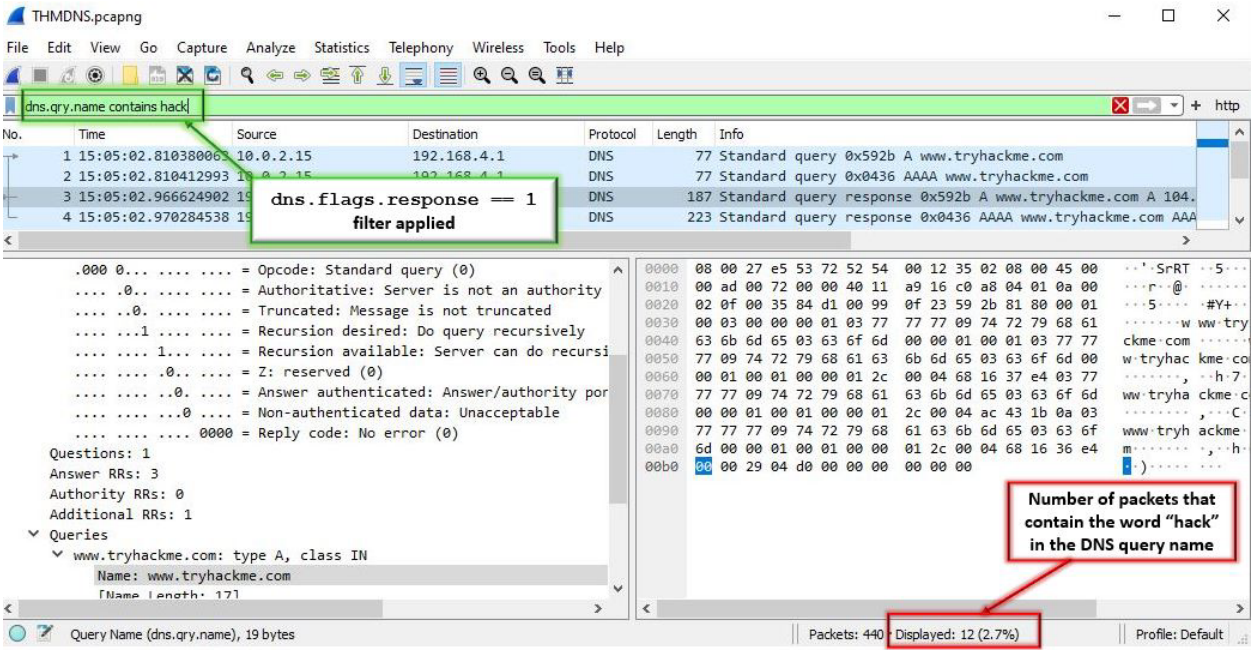

Figure 4: Number of packets that contain the word "hack" in the DNS query name

5. Observing the first and third packets in the packet list pane I know that the third packet is a response to the first packet based on their transaction ID, `0x592b`. Selecting the third packet, the response, I went to the packet details pane and expanded everything under `Domain Name System (response)` and examined the content in `Answers`. The third address resolved to www.tryhackme.com was `104.22.54.228` (*Figure 5*).

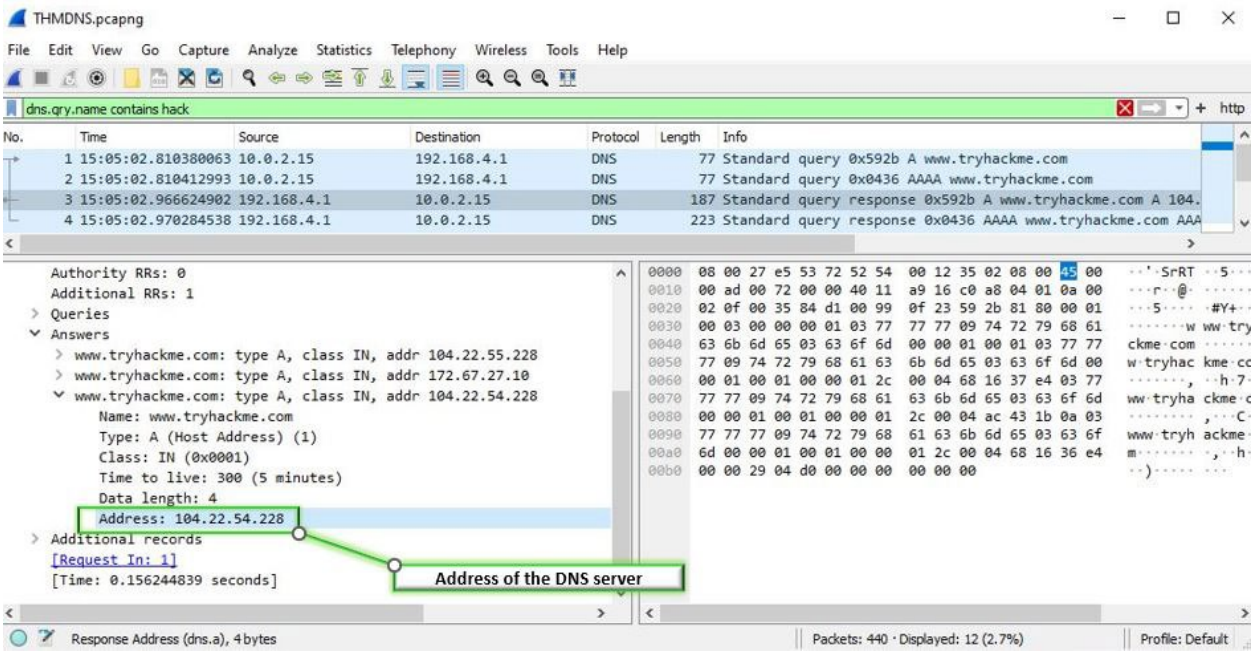

Figure 5: Third address resolved to www.tryhackme.com

6. Observing the same location via Wireshark where I found the third address that resolved to www.tryhackme.com, looking at `Time to live`, I know that the record will be stored for `300` seconds in the client cache (*Figure 6*).

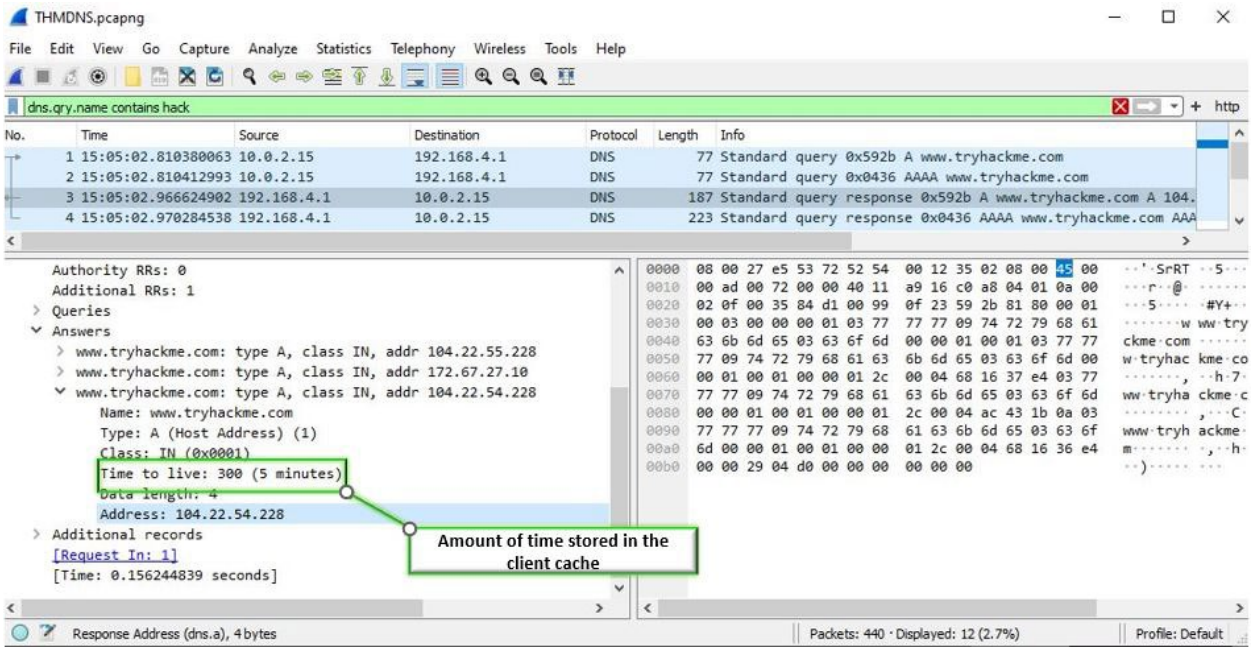

Figure 6: Amount of time the record will be stored in the client cache

7. By applying `Answer RR` as a column in Wireshark, and applying `DNS` filter, I discovered that packet (or frame number) `62` carried the most answer RRs (*Figure 7*).

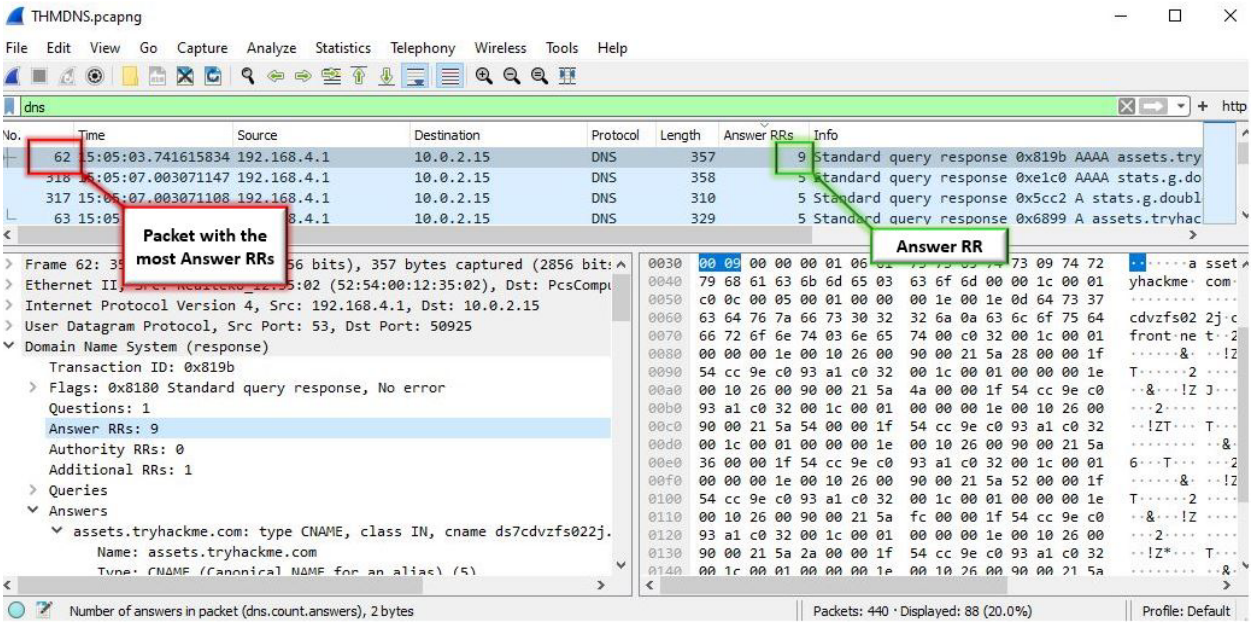

Figure 7: Packet (frame number) with the most Answer RRs via Wireshark

8. `dns.qry.type==28` is the filter to use to display all DNS queries (and responses) for IPv6 addresses. To determine the appropriate filter—and watching Chris Greer's YouTube video [2]—I examined the contents in the packets detail pane. Specifically, observing the contents under Domain Name System (response), looked at the `Type` associated with each answer I noticed `AAAA (IPv6 Address) (28)`. By clicking on dragging one of those aforementioned entries into where I normally apply a filter in Wireshark created a filter, which I simply modified to produce queries instead of responses. (*Figure 8*)

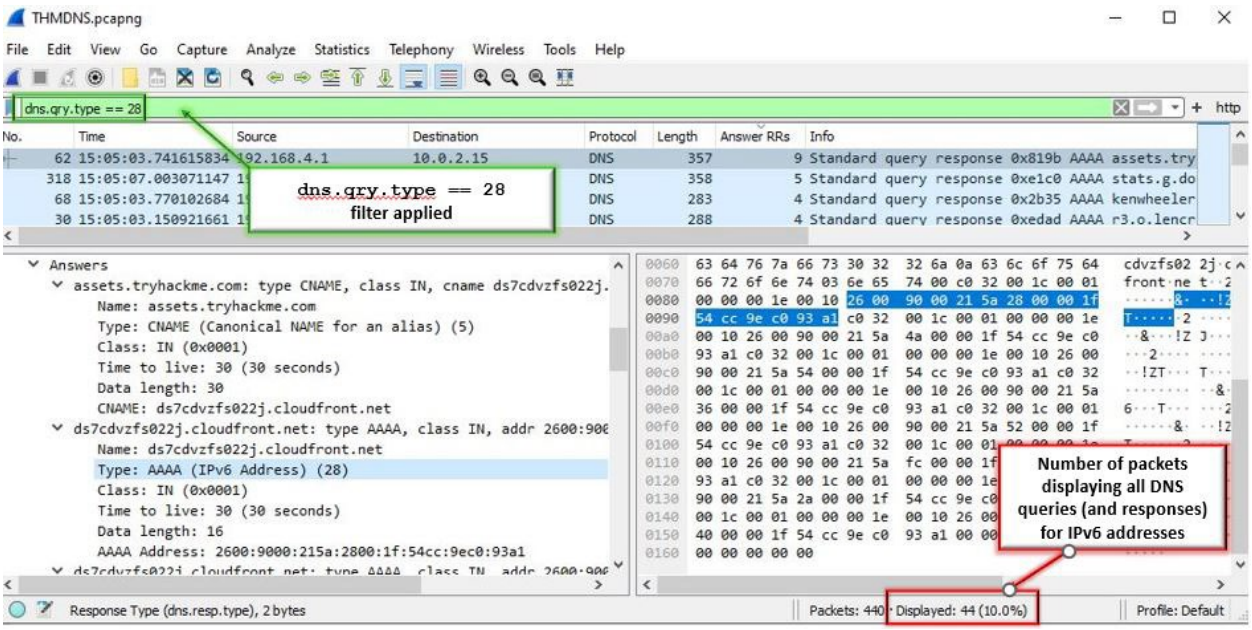

Figure 8: Number of packets display all DNS queries and responses for IPv6 addresses

I will be using the `PCAP` file, `THMWeb.pcapng`, provided by *TryHackMe.com* for this section of the lab in Wireshark. [1]

1. By using `frame matches "assets"` filter in Wireshark, I discovered that `10` packets contain the word "assets", regardless of case (*Figure 9*).

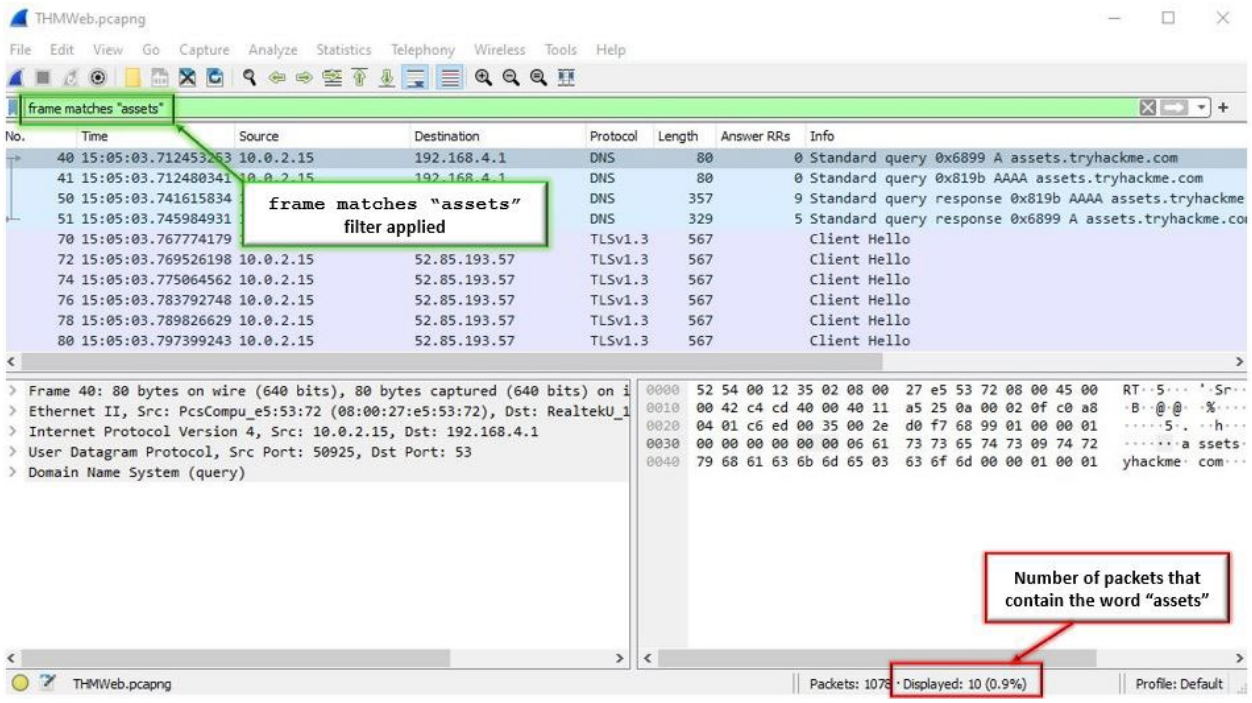

Figure 9: Number of packets that contain the word "assets" via Wireshark

2. Using `frame contains "AWS"` filter, I discovered that packet number `804` contains the string "AWS" (*Figure 10*).

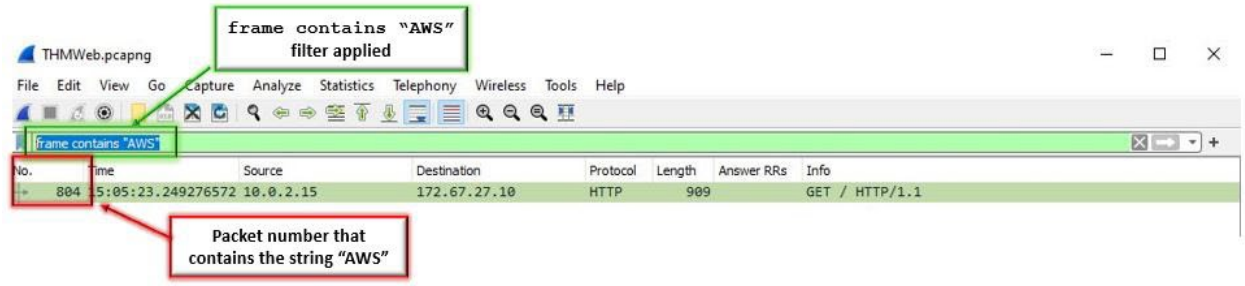

Figure 10: Packet number with the string "AWS" via Wireshark

3. Using `frame matches ".org|.com"` filter, I discovered that `28` packets contain the strings ".org" or ".com" (*Figure 11*).

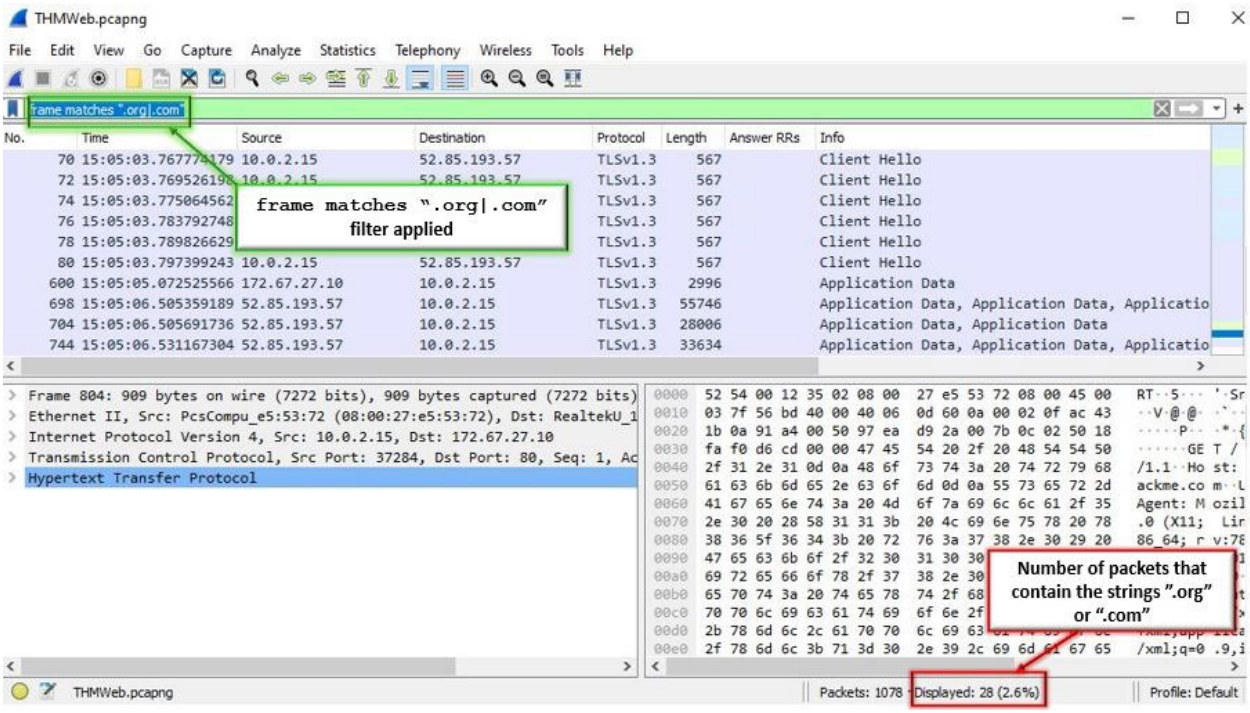

Figure 11: Number of packets that contain the strings ".org" or ".com" via Wireshark

4. Using `tcp.port in {60404..60406}` filter, I discovered that `32` packets are associated with TCP ports 60404 through 60406 (*Figure 12*).

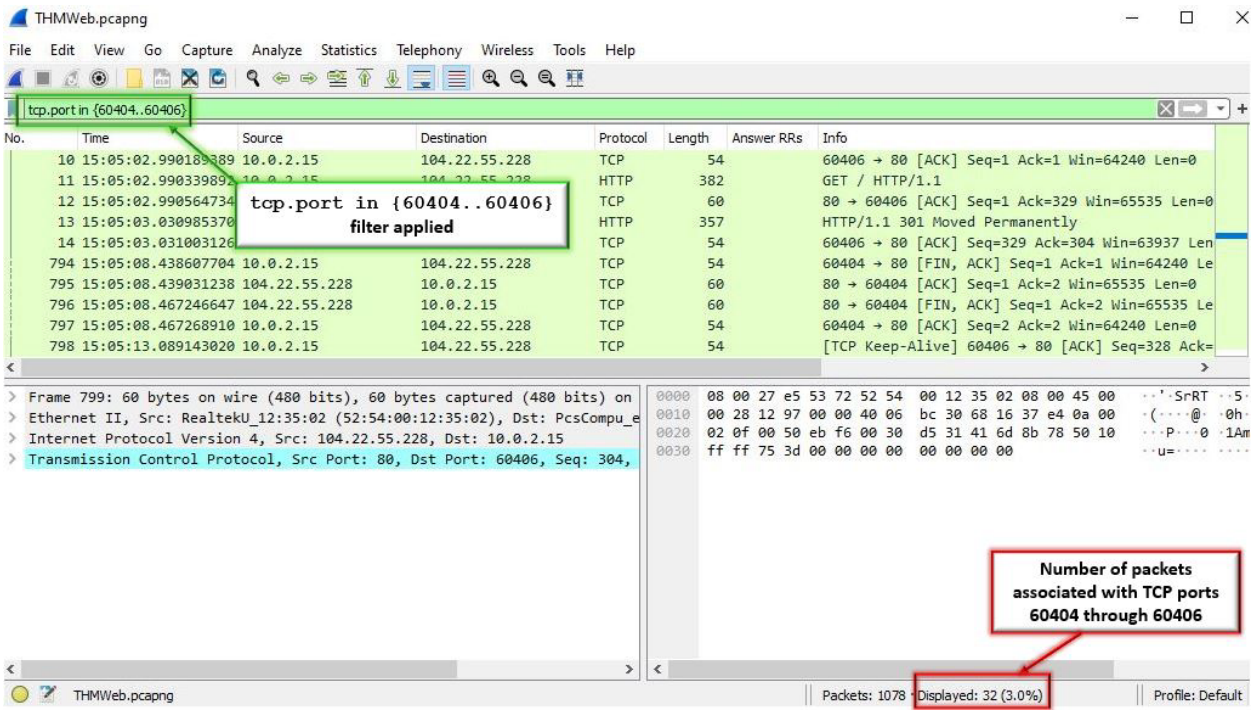

Figure 12: Number of packets associated with TCP ports 60404 through 60406

5. Using `frame contains "GET"` filter, I discovered that `2` packets contain the string “GET” (*Figure 13*).

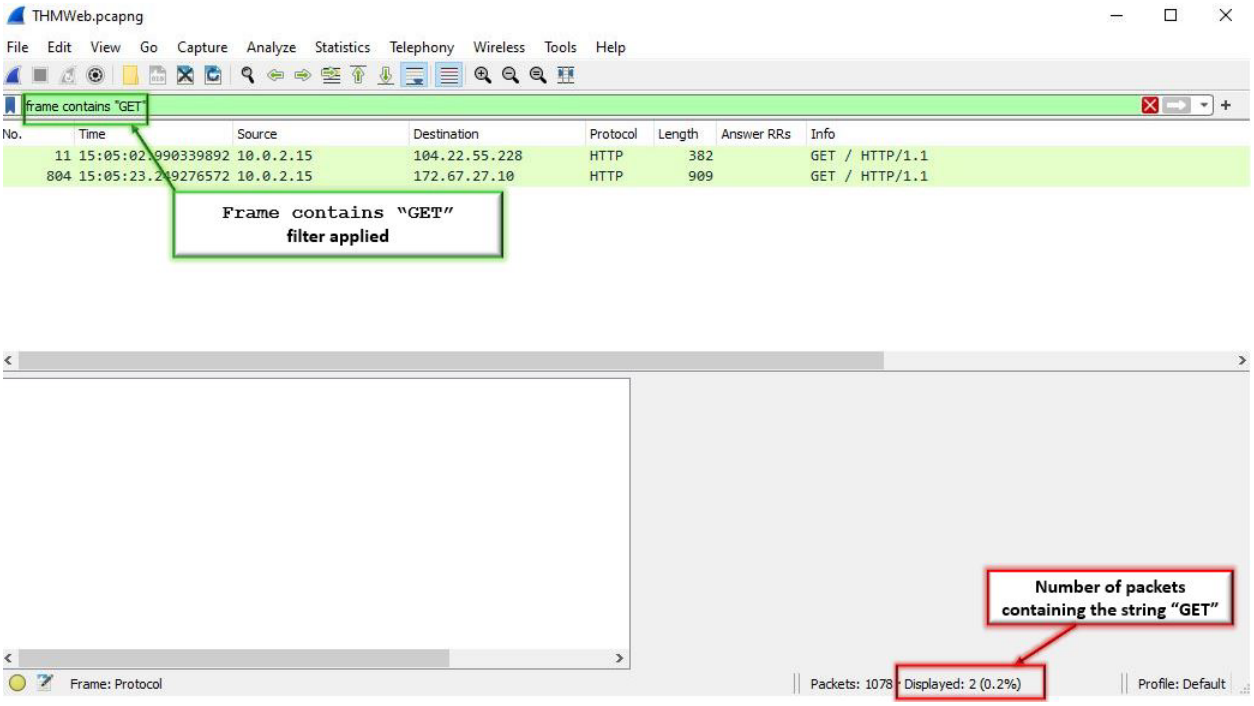

Figure 13: Number of packets containing the string "GET" via Wireshark

LIMITATIONS/CONCLUSION

As an introductory experiment for the novice user like myself, I thought the lab's difficulty was simple. I do not assess there were any limitations because everything was executed in a live environment versus a controlled environment, like on a virtual machine. The biggest takeaway from this lab was learning how to use proper symbols and syntax to execute filters in Wireshark for the user's desired results.

REFERENCES

[1] TryHackMe [Online]. "Wireshark Filters", 2022. Available: https://tryhackme.com/room/wiresharkfilters [Accessed: 08-Nov-2022]

[2] Greer, *YouTube* [Online]. "TryHackMe WIRESHARK Filters Walkthrough", September 6, 2022. Available: https://www.youtube.com/watch?v=-MLkdg4s4ew [Accessed: 08-Nov-2022]

COLLABORATION

The entirety of this lab was executed independently by the author. No additional collaboration to report.