# **Intro to Wireshark**

(Quelle: TryHackMe Adventskalender, Tag 9, <https://tryhackme.com/room/adventofcyber3> übernommen, angepasst)

Passendes Youtube-Video: <https://www.youtube.com/watch?v=LnBT1qubCnc> (45 Minuten)

Passendes Lösungsformular: <https://forms.office.com/Pages/ResponsePage.aspx?id=KD8PHtdlokW6B_SGKKJ1dCrTaL9oAf5MsM5Ggui4vhZUNEkzVjRNR0JDV1BEUFdCQUlYUDhRWkRIUi4u>

**What is Wireshark?**

Wireshark is pre-installed on the THM AttackBox, and you can just launch the THM AttackBox by using the Start AttackBox button and opening Wireshark.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

If you want to do this challenge on your own computer, Wireshark can be downloaded from here: [Download Wireshark](https://www.wireshark.org/download.html)

For this task, we have prepared a PCAP file to download and follow the instructions and apply the required packet analysis skills using different scenarios. If you're using the AttackBox you don't need to download the file as it's already on the machine; the AoC3.pcap file is in the following location: /root/Rooms/AoC3/Day9/AoC3.pcap.

We can run Wireshark and import the provided PCAP file as follows,

Terminal

user@thm$ sudo wireshark /root/Rooms/AoC3/Day9/AoC3.pcap

The Wireshark interface has five major components: The command menus are standard pulldown menus located at the top of the window. Of interest to us now are the File and Capture menus. The File menu allows you to save captured packet data or open a file containing previously captured packet data, and exit the Wireshark application.

The display filter section is where we can specify rules to find certain packets. The listing of captured packets shows all sent and received network packets. This section includes important fields that are used in the filter section, such as source and destination IP addresses, protocols, and information. Next, the details of a selected packet header. This section shows details about selected network packets and shows all headers, including all TCP layers information. Finally, the packet content which shows the packet content in hexadecimal and ASCII format.

Ein Bild, das Tisch enthält.

Automatisch generierte Beschreibung

**Filters**

[Berkeley Packet Filter (BPF)](https://en.wikipedia.org/wiki/Berkeley_Packet_Filter) syntax is used in packet analyzers to filter specific packets pre-capture. Filtering packets is beneficial when locating information within a packet capture process. Wireshark's syntax is the primary method we will use in this room to locate certain protocols and information. Below we have included some examples of using Wireshark display filters:

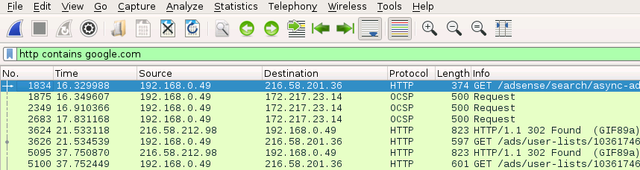
Let's assume that we are looking for all packets that have been sent or received by the following IP address: 172.21.2.116. Thus, the following filter is helping us to display all network packets that have the IP address 172.21.2.116: ip.addr == 172.21.2.116

Ein Bild, das Text enthält.

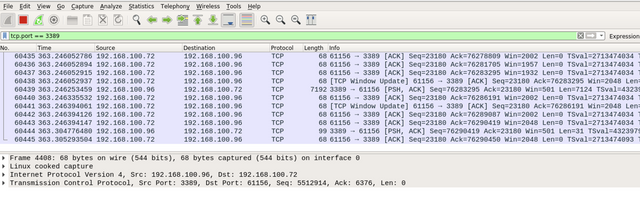
Automatisch generierte Beschreibung

As a result, we can see that we were able to show only the packet(s) that we needed.

Next, we can also specify certain protocols, such as HTTP showing all packets for this protocol. We can also specify a domain name to narrow down the search. The following example shows that we are looking for HTTP packets that have the google.com  domain name: http contains google.com



Next, we can also look at a specific port. Let's try to filter and list all packets for remote desktop protocol. We can do that by using tcp.port : tcp.port == 3389



However, assume that you are monitoring the network traffic and you want to exclude the RDP packets. In this case, we can use the not Wireshark rule as shown below.

Ein Bild, das Tisch enthält.

Automatisch generierte Beschreibung

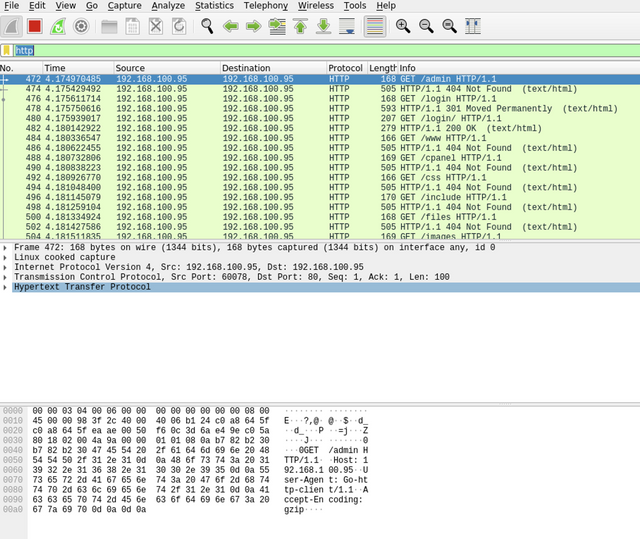
Next, we will perform the packet analysis technique using Wireshark on various protocols, including HTTP, FTP, and DNS.

**Note**: Make sure to open the Wireshark tool, and the AoC3.pcap file is imported to follow up with the content.

**HTTP #1 - GET**

We will be performing a packet analysis on the pre-captured network packets in the PCAP to analyze HTTP traffic. Once we open the PCAP using Wireshark, it will show a lot of network traffic since we connect using RDP protocol, making our work harder. However, we will be using a Wireshark filter to focus on web requests only. In this case, we are dealing with the HTTP protocol. Therefore, we can use Wireshark rules to make it easier. We can apply the following rule to the Wireshark filter to show only the HTTP traffic: http.

The following figure shows all HTTP traffic that has been sent to and from the webserver. In the first packet, we can see that the web request is sent from our machine 192.168.100.95  to the webserver, which is also our machine. In the information section, we can see also that we sent a GET request trying to get /admin content. On the next packet, the webserver replied to the sender with an HTTP response code 404, which means it is not found. Suppose we need more details on a particular packet. In that case, we can expand (using the > sign) the Hypertext transfer protocol subtree for more details about the sender request, such as the IP address of the host, web user agent, and other information.



We can specify the HTTP method. In this case, we can specify to show all the GET requests using the following filter: http.request.method == GET. Keep following the GET HTTP requests and answer the question in this task.

**HTTP #1 - POST**

In the second scenario, we check HTTP POST requests used to log into a web portal. In this section, we will be performing packet analysis to capture the username and password of the POST request. Make sure to open the PCAP file to perform the packet analysis. Your goal in this section is to inspect the HTTP packet to get the username and password. For more details, you double-click on the required packet that has the POST request. We can see all the required information we need in the hexadecimal section in cleartext. In addition, there are other ways that show more information the HTTP request, which is to follow the HTTP stream. We can do that by right-clicking on the required packet and then selecting follow -> TCP Stream to see the web requests and their details.

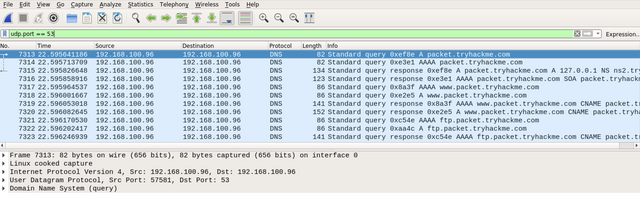
In this section, we have to find the POST request and explore all packet information to answer question 2 and 3 in this task.

**DNS**

DNS is like a giant phone book that takes a URL (Like https://tryhackme.com/) and turns it into an IP address. This means that people don’t have to remember IP addresses for their favorite websites.

The provided PCAP file has DNS packets that have been received by our server. The domain name that we will look into is packet.tryhackme.com. Please note that is the domain name used in the PCAP file as an example.

First, make sure that you open the PCAP file. Then, using the Wireshark filter, we will specify DNS packets. This can be done using the udp.port filter. By default, the DNS protocol is running on UDP port 53 and sometimes on TCP port 53. In this task, we will go with the UDP port. Thus, the Wireshark filter will be as follows: udp.port == 53 or dns only.



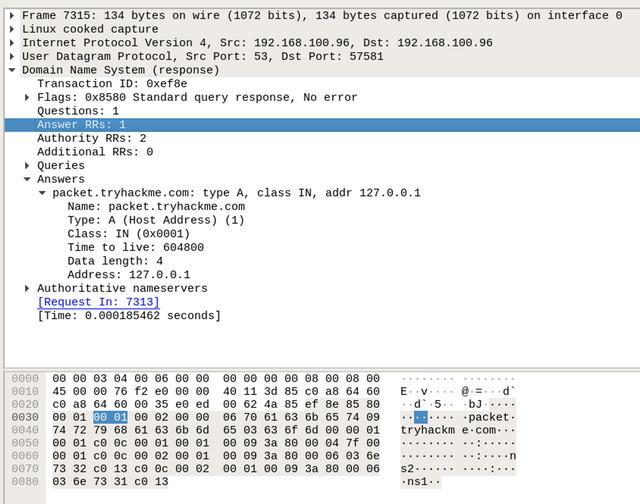
By double-clicking on the first DNS packet, we can see as follows:

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

By expanding the Domain Name System (query) subtree, it is obvious that this packet is a question Questions: 1, where the client queries a domain name and looks for an answer from the server. Also, by checking the Queries subtree, we can see that the client is asking for a type A record of the packet.tryhackme.com domain name.

Next, we will be looking at the response to the DNS request. In Wireshark, note that in the query request, which is the first DNS request, in the info field, we can see Transaction ID with a value of query 0xef8e. Note that this query reference number may vary in your situation. Thus, this query reference number is used to find the right DNS response of that query. To confirm that, check the field section where showing query response 0xef8e A packet.tryhackme.com.

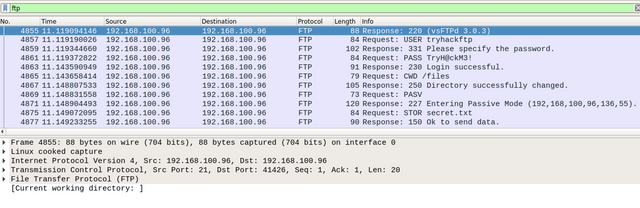


By double-clicking on the required packet, we can see that it has an answer section, where it has the answer to our query, which is Address: 127.0.0.1. This means that the type A DNS request of the packet.tryhackme.com  is 127.0.0.1. Note that the PCAP file may have different IPs or extra packets that have not been discussed. There are other DNS requests that have been generated in the PCAP file. Now, dig more into DNS queries and responses in Wireshark and answer question 4 in this task.

**FTP**

The provided PCAP file has FTP packets that have been received by our server. For this task,  we will be looking at FTP traffic in Wireshark. Like other protocols, we will be using Wireshark filters to look into FTP packets. By default, the FTP server is listening on TCP port 21. Therefore, we filter and list all FTP packets by using the following filter: ftp or tcp.port == 21

The following figure shows all FTP packets, and we can see all packet details by checking the field section. It is obvious that the sender is connected to the FTP server and the server responded with the FTP server header, vsFTPd 3.0.3. Next, the user sent an FTP request with tryhackftp as a user command which is the username of the FTP server.



Keep checking the FTP packets in Wireshark and answer question 5 and 6 in this task. We can also show all FTP commands by choosing the first FTP packet and right-clicking on it. Then, select follow -> TCP-Steam.

The user logged into the FTP server using a username and password and uploaded a secret file. Thus, the current Wireshark filter will not show the packet of the actual uploaded file. Instead, we can use the ftp-data Wireshark filter to list the packets that have the content of the file. Apply this filter and answer the final question in this task.

**Questions**

**Question 1 (HTTP GET)**

In the **HTTP #1 - GET** requests section, which directory is found on the web server?

Ein Bild, das Tisch enthält.

Automatisch generierte Beschreibung

🡪 Forms answer 1

**Question 2 (HTTP POST)**

What is the username and password used in the login page in the **HTTP #2 - POST** section?

Ein Bild, das Text, Monitor, Screenshot enthält.

Automatisch generierte Beschreibung

🡪 Forms answer 2 (Format username:password)

**Question 3 (HTTP POST User Agent)**

What is the User-Agent's name that has been sent in **HTTP #2 - POST** section?

🡪 Forms answer 3 (Full string)

**Question 4 (DNS)**

In the DNS section, there is a TXT DNS query. What is the flag in the message of that DNS query?

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

🡪 Forms answer 4 (Full string)

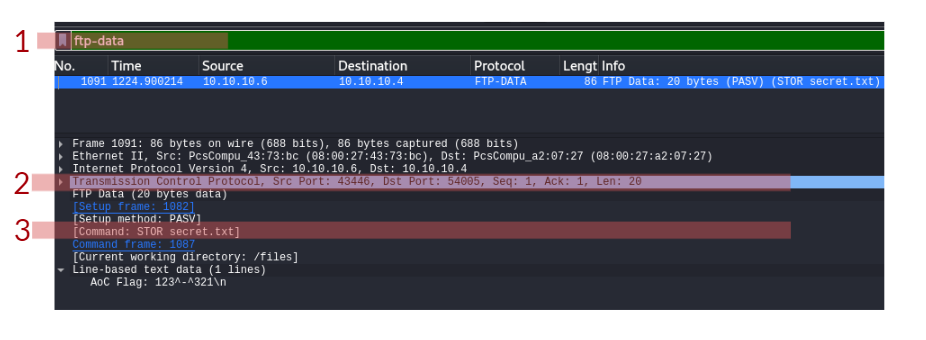
**Question 5 (FTP Username / Password)**

In the FTP section, what is the FTP login password?

🡪 Forms answer 5 (password)

**Question 6 (FTP File transfer)**

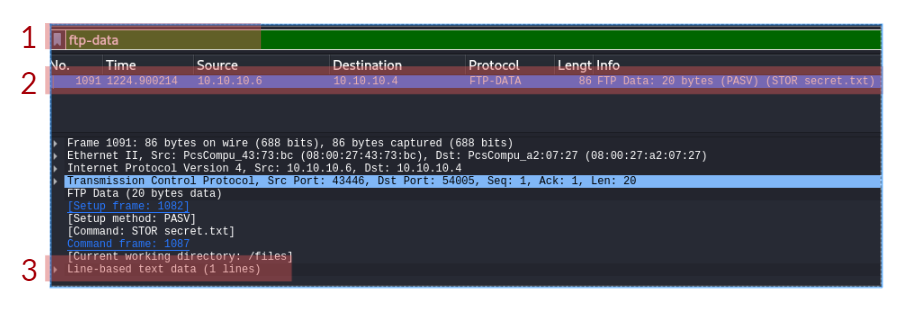
In the FTP section, what is the FTP command used to upload the secret.txt  file?



🡪 Forms answer 6 (FTP command in capital letter)

**Question 7 (FTP File content)**

In the FTP section, what is the content of the secret.txt file?



🡪 Forms answer 7 (Full text)