Wireshark- Packet Capturing & Filtering

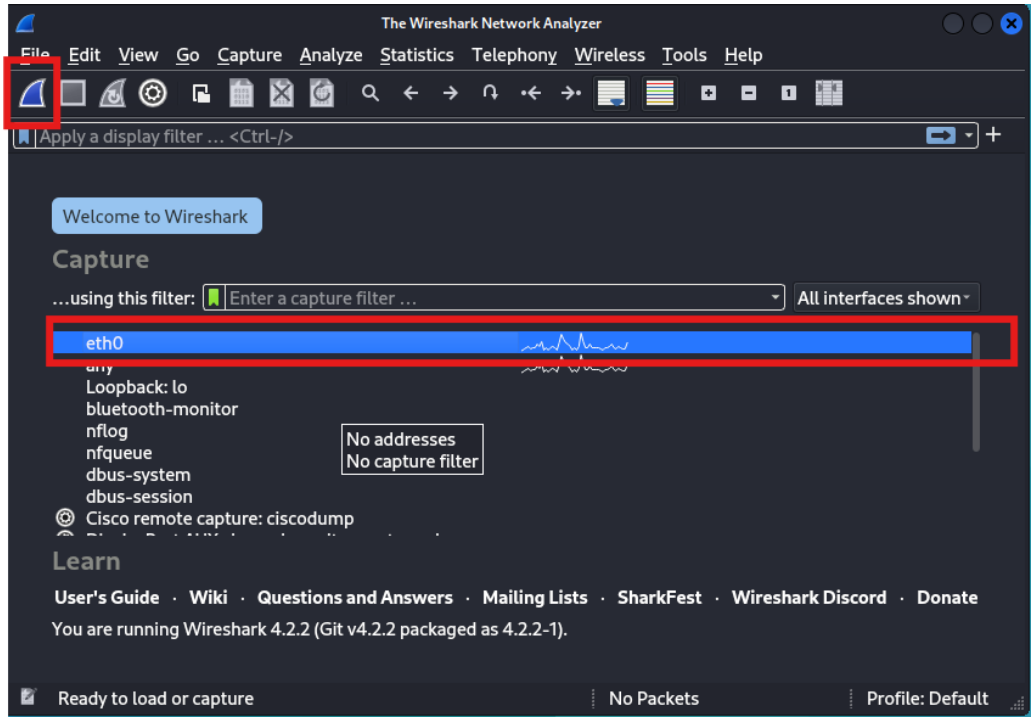
Wireshark is a robust, open-source network protocol analyzer widely used for network troubleshooting, analysis, software and protocol development, and cybersecurity education. It captures network packets in real time and presents them in a detailed, readable format, allowing users to see the data flowing through their network. With its powerful filtering and search capabilities, Wireshark helps identify network issues, monitor traffic patterns, and detect potential security threats. It supports a vast array of network protocols, making it an invaluable tool for IT professionals, network administrators, and cybersecurity experts. Its graphical user interface (GUI) and extensive documentation make it accessible for beginners while offering advanced features for experienced users.

Wireshark filters are essential tools that allow users to refine and manage the vast amount of data captured during network traffic analysis. These filters come in two main types: capture filters and display filters. Capture filters are applied before data collection begins, specifying which packets Wireshark should record, thus reducing the volume of captured data and focusing on relevant traffic. Display filters, on the other hand, are used after data has been captured to sift through the recorded packets, highlighting

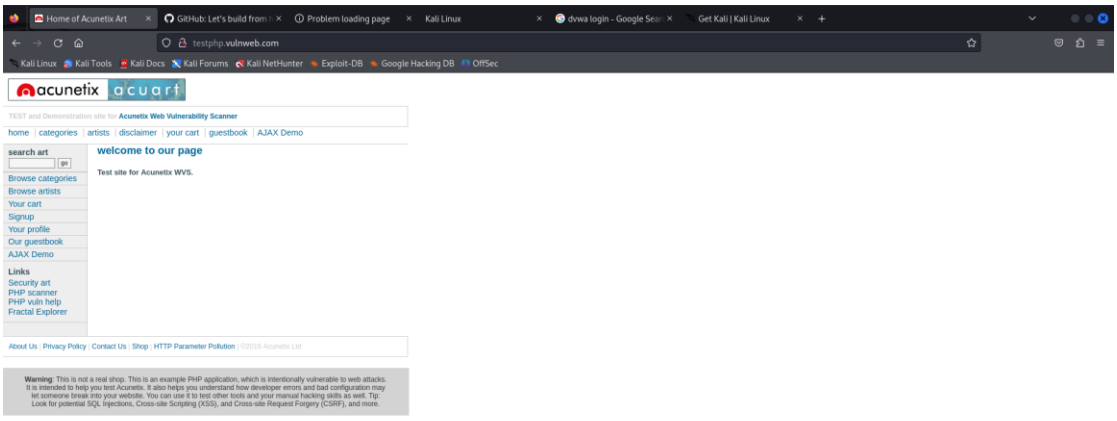
specific types of traffic or patterns of interest. By utilizing filters, users can isolate protocols, IP addresses, ports, or packet contents, making it easier to diagnose network issues, identify security breaches, and understand complex network interactions. The flexibility and power of Wireshark filters make them indispensable for efficient and effective network analysis.

Examples: -

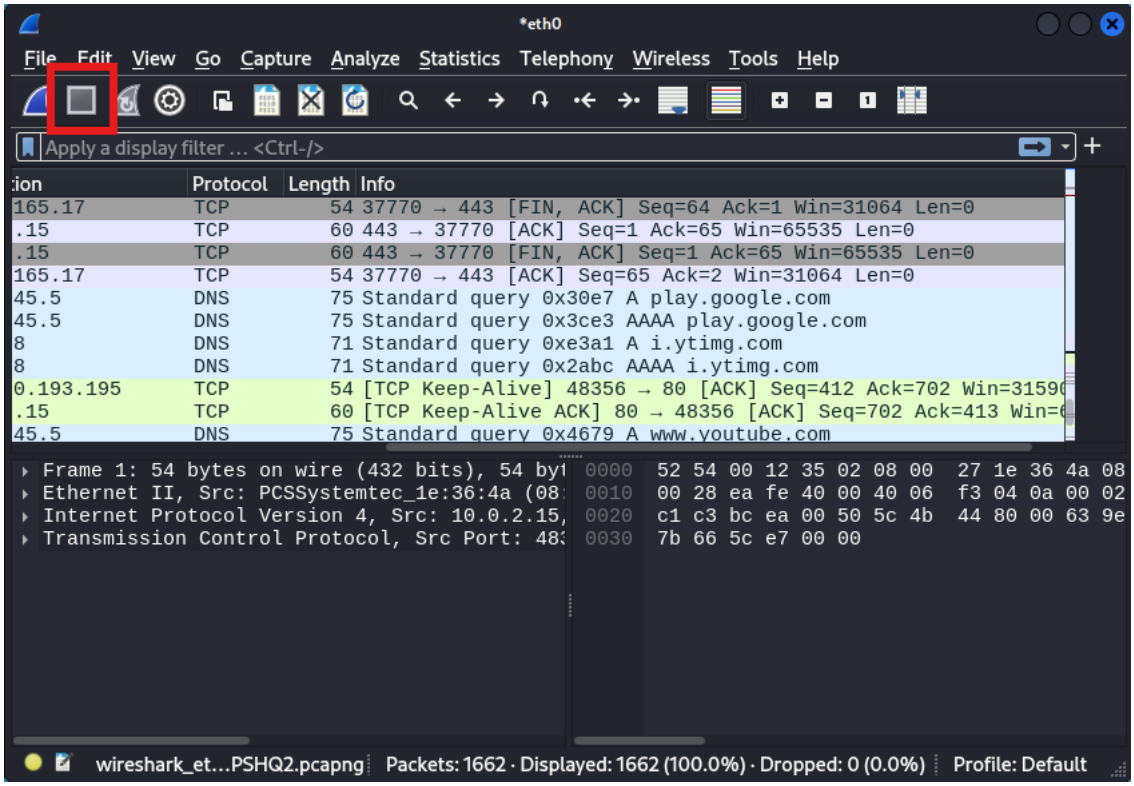
1. Open wireshark to perform packet sniffing and analysing. To do so, choice the desired interface and start the capturing using start button.



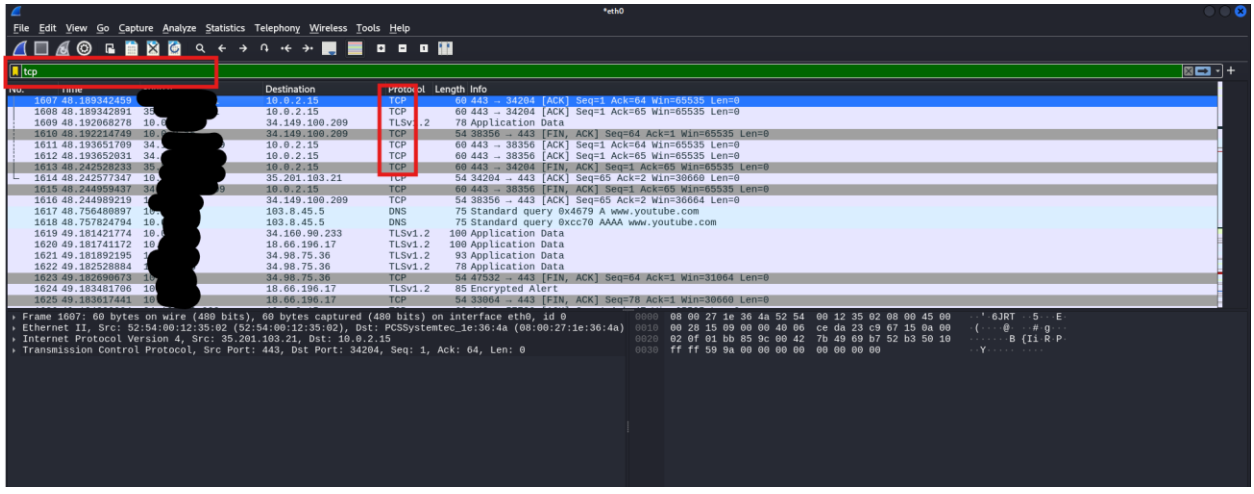
2. Now go to web browser and run various websites in order to capture few web requests.



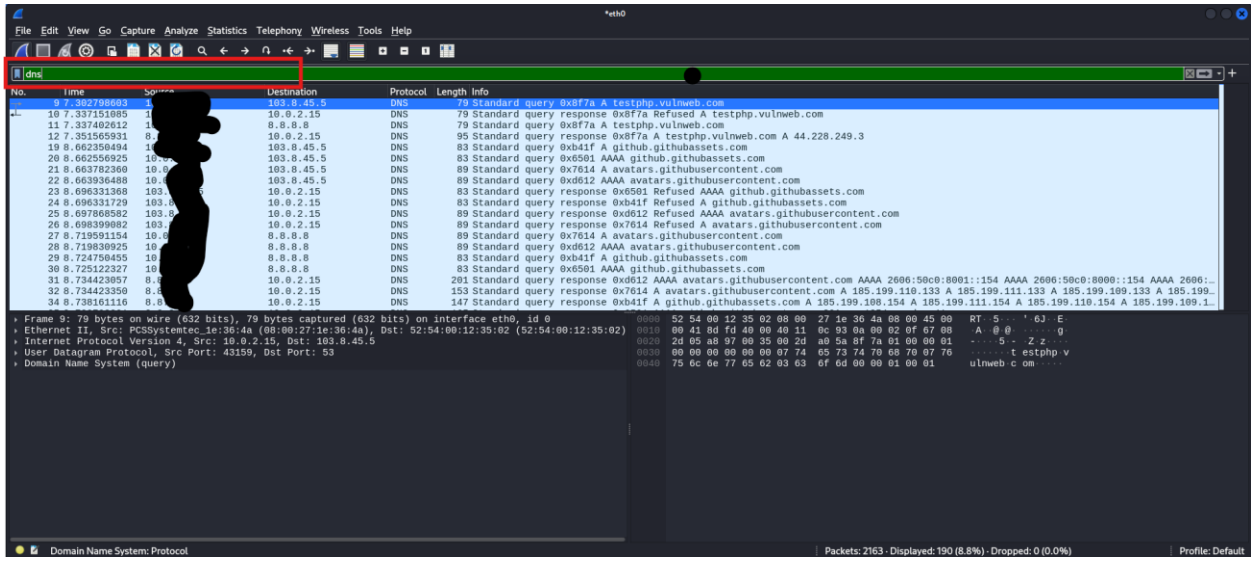
3. Now go back to the wireshark and stop the capturing using stop button to analyze the captured packets.



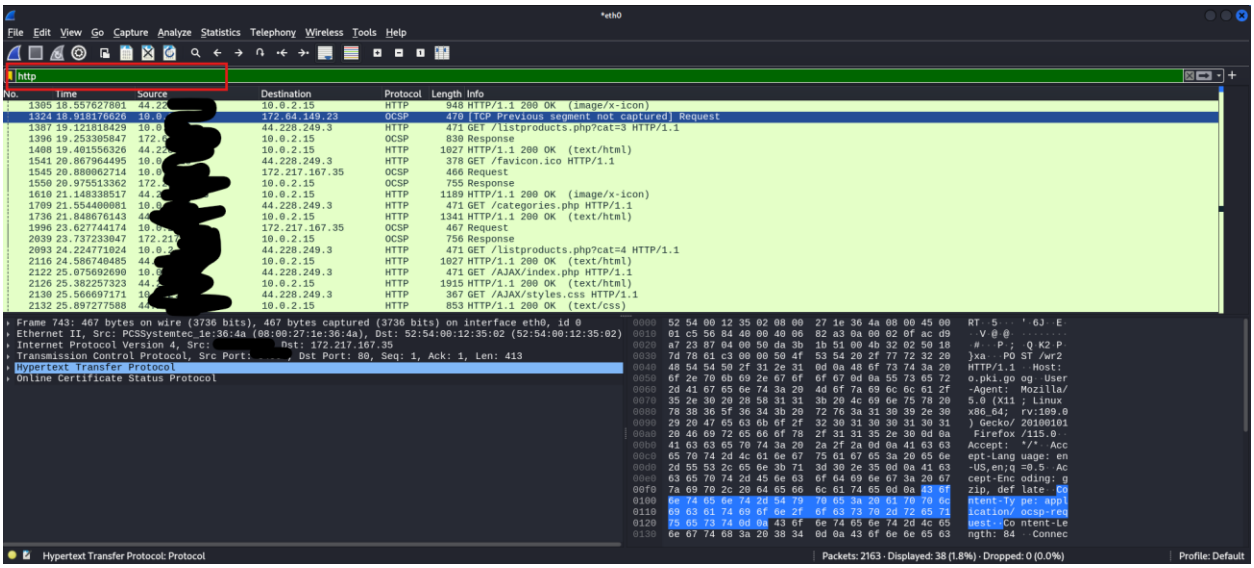
4. Now we will perform tcp filter to capture the tcp packets.



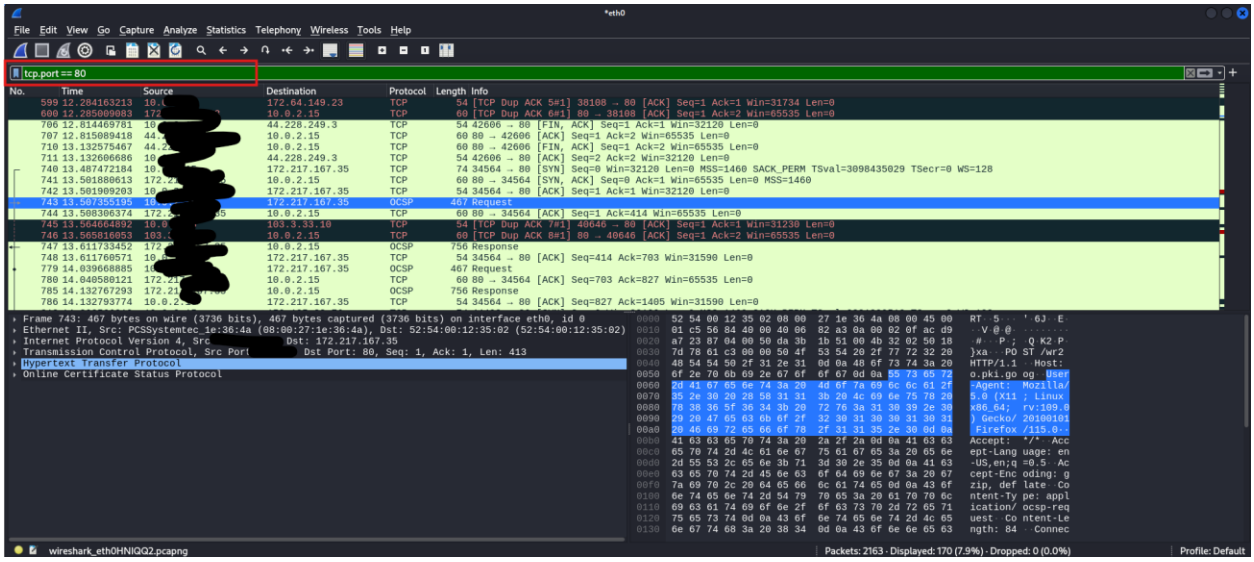
5. Here we use dns filter to capture DNS requests.



6. In this example, I use http filter to filter out http packets.



7. Now I have used tcp.port == 80 filter to capture the tcp packets that are being used on port 80.



8. In this example I used http.request filter to filter out the http requests packets.