Packet Sniffer

Submitted By

Muhammad Ashir
22K-4504

Department of Computer Science

National University of Computer & Emerging Sciences

Motivation

The increasing complexity and reliance on computer networks for communication, commerce, and critical infrastructure necessitates effective tools for network monitoring and analysis. Understanding network traffic flow, identifying potential issues, and ensuring security are paramount. Packet sniffers are fundamental tools for achieving these goals, providing insights into data transmission at a granular level. This project aims to develop a practical, Python-based packet sniffing tool with a graphical user interface (GUI) to simplify the process of capturing, visualizing, and analyzing network traffic in real-time. The development of this tool provides a valuable hands-on learning experience in network programming, protocol analysis, and the challenges of cross-platform network operations, directly applying concepts learned in the CS3001 Computer Networks course.

Overview

This project involves the creation of a Packet Sniffing Program using Python. The core idea is to develop a tool capable of capturing network packets live from a network interface, dissecting them to display header information for various protocols, and presenting this information to the user through an intuitive GUI.

- **Significance:** The project holds significance as it provides a practical tool for network administrators, security analysts, and students to monitor network activity, troubleshoot connectivity problems, and learn about network protocols.
- **Importance and Practicality:** In an era of complex network environments, the ability to easily inspect traffic is crucial for diagnostics and security. This tool offers a readily accessible way to perform such analysis. Its GUI makes it more user-friendly than purely command-line tools.
- **Academic Value:** This project reinforces theoretical knowledge of computer networks (TCP/IP stack, Ethernet, IP, TCP, UDP, ICMP, HTTP protocols) and provides practical experience in network socket programming, multithreading (for non-blocking GUI), GUI development (using Tkinter), and cross-platform considerations (especially regarding raw socket access).

Description of the Project

The Packet Sniffing Program is designed to address the need for a clear and interactive way to monitor network traffic.

- **Problem:** Network analysis often requires specialized, sometimes complex, tools. This project aims to provide a simpler, visually-oriented alternative built with accessible technologies like Python. The goal is to capture packets, parse their headers, identify protocols, and display the information comprehensibly.
- **Scope:** The sniffer operates on a local machine's network interface. It focuses on decoding common protocols within the TCP/IP suite: Ethernet, IPv4, TCP, UDP, ICMP, and HTTP. It provides functionality for real-time capture, packet detail viewing,

protocol-based filtering, and saving captured data to a PCAP file for offline analysis with tools like Wireshark. The implementation uses the Scapy library for packet manipulation and sniffing, and Tkinter for the GUI.

Background of the Project

Packet sniffing involves capturing data packets as they traverse a network interface. This requires accessing the network stack at a low level, often using raw sockets. Tools like Wireshark are industry standards, but building a custom sniffer offers learning benefits and potential customization.

Python, with libraries such as Scapy, provides powerful capabilities for network packet manipulation. Scapy simplifies the process of crafting, sending, capturing, and dissecting packets. However, accessing raw sockets necessary for sniffing presents platform-specific challenges:

- **Linux:** Generally allows raw socket access for privileged users (root) without requiring additional libraries for basic sniffing functionalities, although libraries like Scapy are still beneficial for ease of use and advanced features.
- **Windows:** Access to raw sockets for sniffing is more restricted for security reasons, especially in versions post-XP SP2. Standard Python sockets might face `PermissionError: [WinError 10013]` when attempting raw socket operations needed for sniffing without administrative rights. Libraries like **Scapy** on Windows circumvent these limitations by utilizing underlying packet capture libraries like **Npcap** (or the older WinPcap). Therefore, running the sniffer on Windows typically requires:
    1. Installing Npcap (recommended).
    2. Running the Python script with Administrator privileges.

This project leverages Scapy to handle these cross-platform differences, providing a consistent interface for packet sniffing, but the underlying requirements (Npcap, admin rights on Windows) must be met. The project utilizes classes defined in `ProtocolsClasses.py` to parse Ethernet, IPv4, TCP, UDP, and HTTP headers from the raw packet data captured by Scapy. The `Sniffer.py` file contains the main logic, including the Scapy sniffing loop, packet processing, and the Tkinter GUI implementation.

Project Category

This project falls under the category of **Network Utility / Application Development**. It involves developing a functional software application designed for network analysis and monitoring.

Features, Scope, and Modules

- **Key Features:**
    - **Real-time Packet Capture:** Captures packets live from a selected network interface.

   - **Multi-Protocol Support:** Decodes and displays information for Ethernet, IPv4, TCP, UDP, ICMP, and HTTP protocols.
   - **Graphical User Interface (GUI):** Implemented using Tkinter, providing a user-friendly way to view packets. Features include:
     - A list view displaying captured packets with key info (Serial No., Protocol, Source/Destination IP).
     - Color-coding for different protocols for easy identification (e.g., HTTP in blue, TCP in green).
     - A details pane showing comprehensive header information for the selected packet (e.g., MAC addresses, IP header details, TCP/UDP ports, flags, raw data).
   - **Packet Filtering:** Allows users to filter the displayed packets based on protocol type (TCP, UDP, ICMP, HTTP).
   - **PCAP File Saving:** Saves the captured raw packet data into a `capture.pcap` file, compatible with Wireshark and other analysis tools. This is handled by Scapy's `PcapWriter`.
   - **Cross-Platform Potential:** Designed using Python and Scapy, aiming for compatibility with Windows and Linux (subject to platform-specific requirements like Npcap and permissions).
- **Scope:**
   - Focuses on IPv4 traffic.
   - Provides basic filtering by common protocols.
   - GUI provides essential visualization and interaction.
   - Error handling is implemented for basic packet decoding.
- **Modules:** (Based on code structure and proposal)
   - **Sniffing Engine (`Sniffer.py` / Scapy):** Core module responsible for capturing raw packets from the network interface using Scapy's `sniff` function. Runs in a separate thread to keep the GUI responsive.
   - **Packet Processing (`Sniffer.py`):** Dissects raw packets using protocol classes. Extracts key information for display and filtering. Appends packets to the global list and updates the GUI.
   - **Protocol Classes (`ProtocolsClasses.py`):** Contains classes (Ethernet, IPv4, TCP, UDP, ICMP, HTTP) responsible for parsing the raw byte data of respective protocol headers.
   - **Graphical User Interface (`Sniffer.py` - `Sniffer_GUI` class):** Manages the Tkinter window, widgets (listbox, text areas, buttons), event handling (packet selection, button clicks), and dynamic updates of the display.
   - **PCAP Writer (`Sniffer.py` / Scapy):** Uses Scapy's `PcapWriter` to save captured raw packets to a file.

Feasibility Study

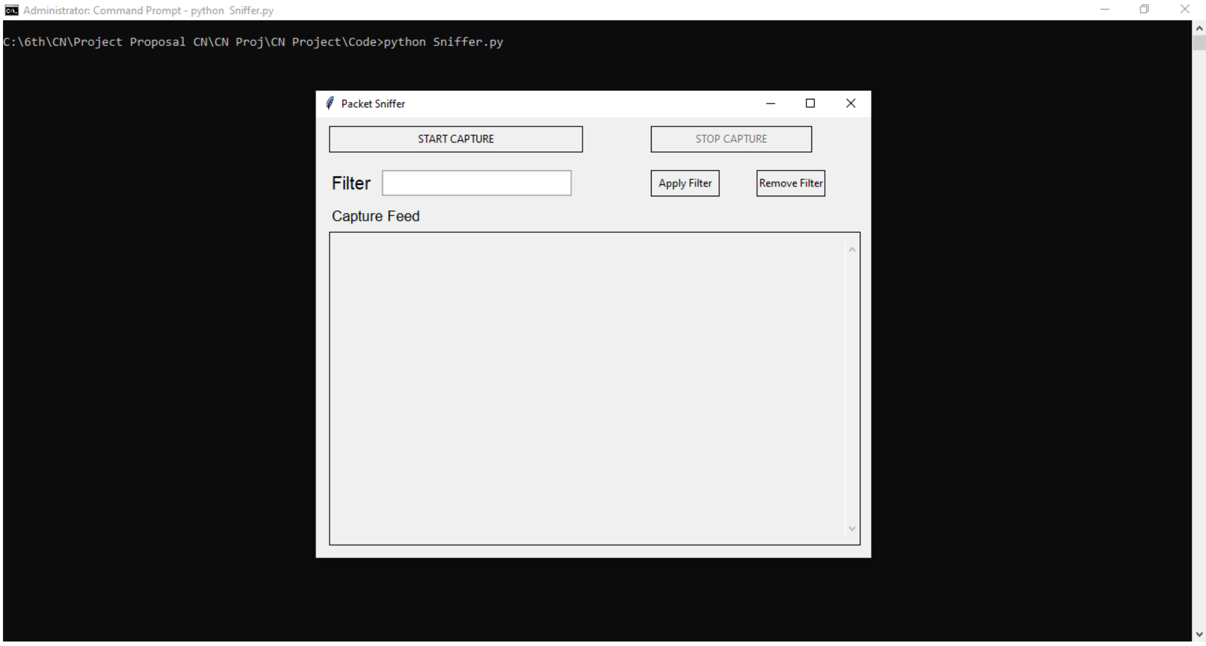
- **Technical Feasibility:**
   - The project is technically feasible using current technology. Python provides robust libraries like Scapy (for packet sniffing/manipulation) and Tkinter (for GUI development) which are well-suited for this task.

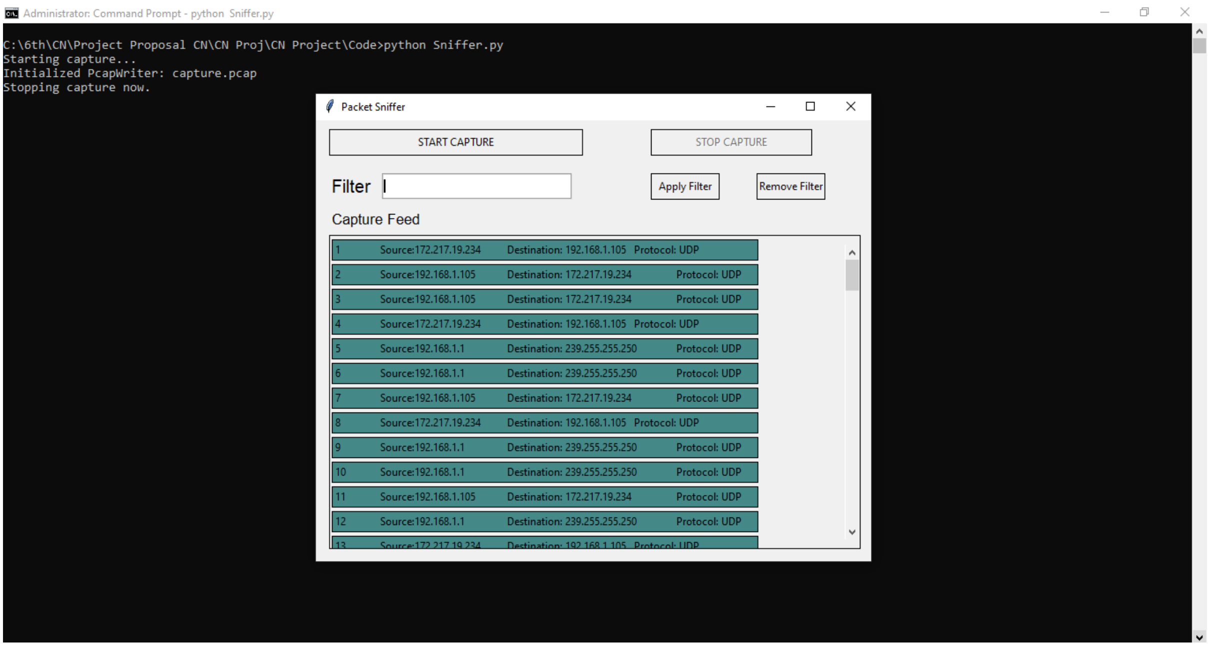
    - The core technical challenge lies in handling raw socket access permissions, particularly on Windows, which necessitates running as administrator and using Npcap with Scapy. This is a known requirement and manageable.
    - The provided code (ProtocolsClasses.py, Sniffer.py) demonstrates a working implementation of packet capture, parsing for multiple protocols, and GUI display, confirming technical viability.
    - Potential risks include performance bottlenecks under very high network load, which can be mitigated through efficient packet processing and potentially optimizing the GUI updates. Compatibility issues with specific network drivers or Npcap versions could arise but are generally resolvable.
- **Economic Feasibility:**
    - The project is highly economically feasible. It primarily relies on open-source software (Python, Scapy, Npcap) and standard hardware (a computer with a network interface card).
    - No significant software licensing costs are involved.
    - Development costs are primarily related to the time investment of the team members. Operational costs are negligible (standard electricity consumption).
    - The benefits include a practical learning experience and a useful network analysis tool, justifying the development effort.
- **Schedule Feasibility:**
    - The project scope, as demonstrated by the submitted code and outlined in the proposal, is achievable within a typical academic semester timeframe.
    - The proposal included a phased timeline (Setup, Core Sniffing, GUI, Filtering, Testing, Finalization) which appears reasonable for the complexity involved.
    - The existence of working code suggests the project is on track or completed, fitting within the schedule feasibility.

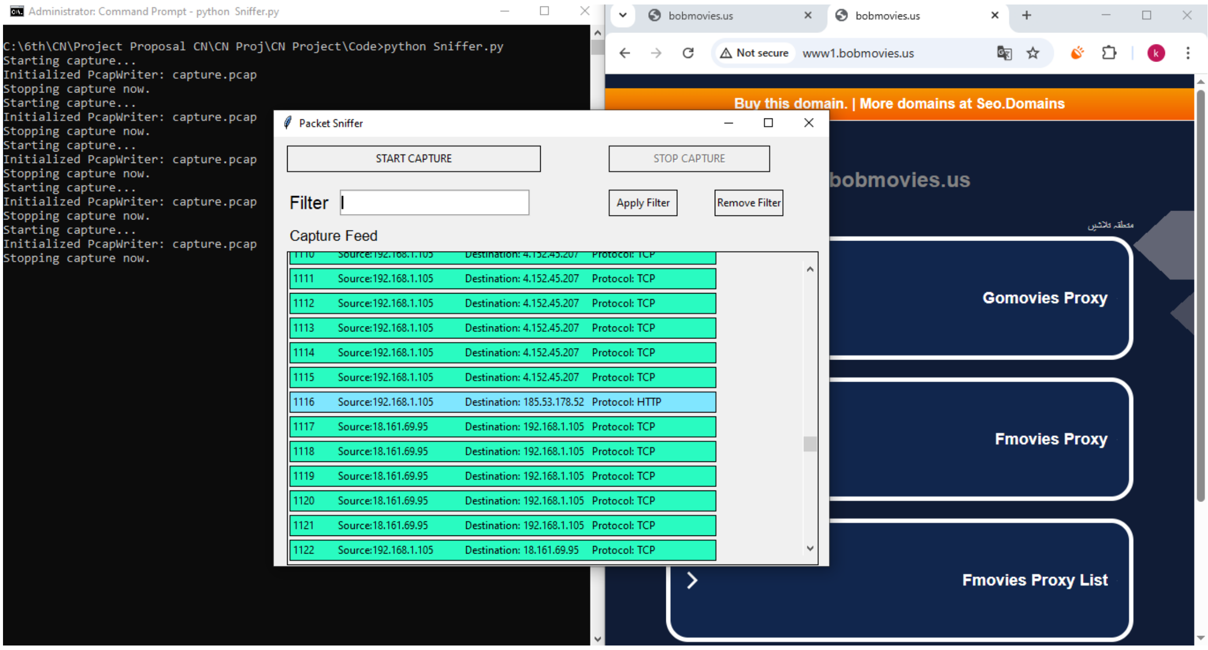
Hardware and Software Requirements

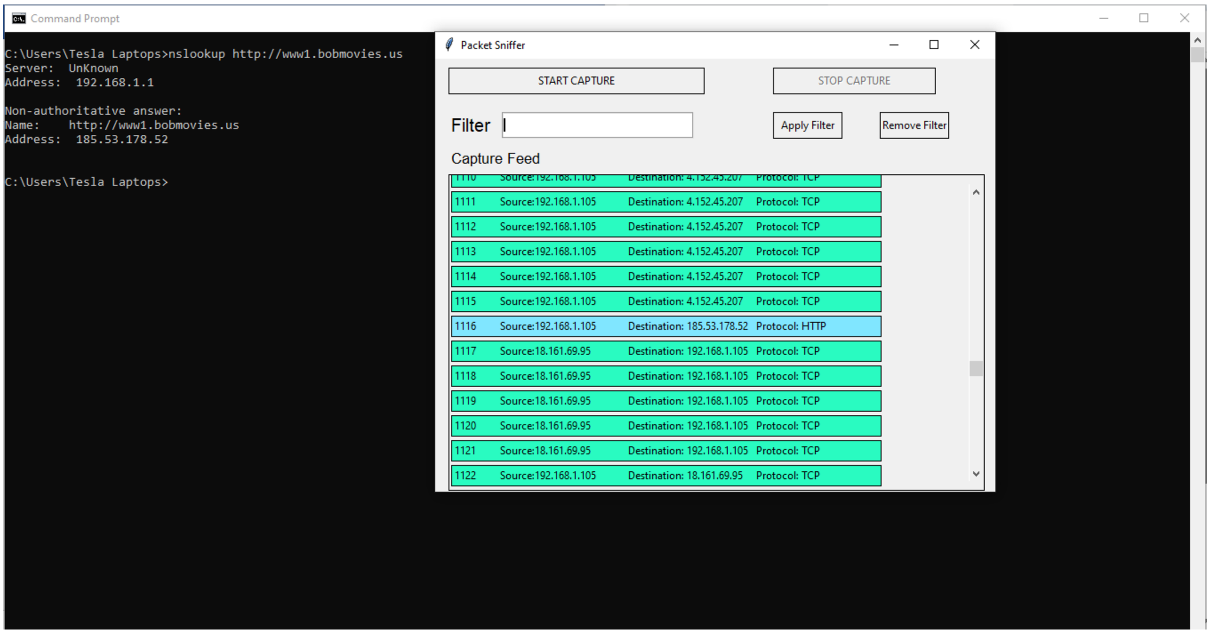
- **Hardware:**
    - A computer (Desktop or Laptop) with a Network Interface Card (Ethernet or Wi-Fi).
    - Sufficient RAM and CPU resources to run Python, Scapy, and handle packet capture (standard modern specifications are adequate).
- **Software:**
    - **Operating System:** Windows (10/11 recommended) or Linux (e.g., Ubuntu, Fedora).
    - **Python:** Version 3.7+ (as required by recent Scapy versions).
    - **Scapy Library:** Python package for packet manipulation (pip install scapy).
    - **Tkinter Library:** Typically included with standard Python installations.
    - **(Windows Specific) Npcap:** Packet capture library for Windows (successor to WinPcap). Must be installed separately. Scapy uses this on Windows.
    - **Administrator/Root Privileges:** Required to run the sniffer script to allow raw socket access and interface manipulation.
    - **(Optional) Wireshark:** For analyzing the generated .pcap files.

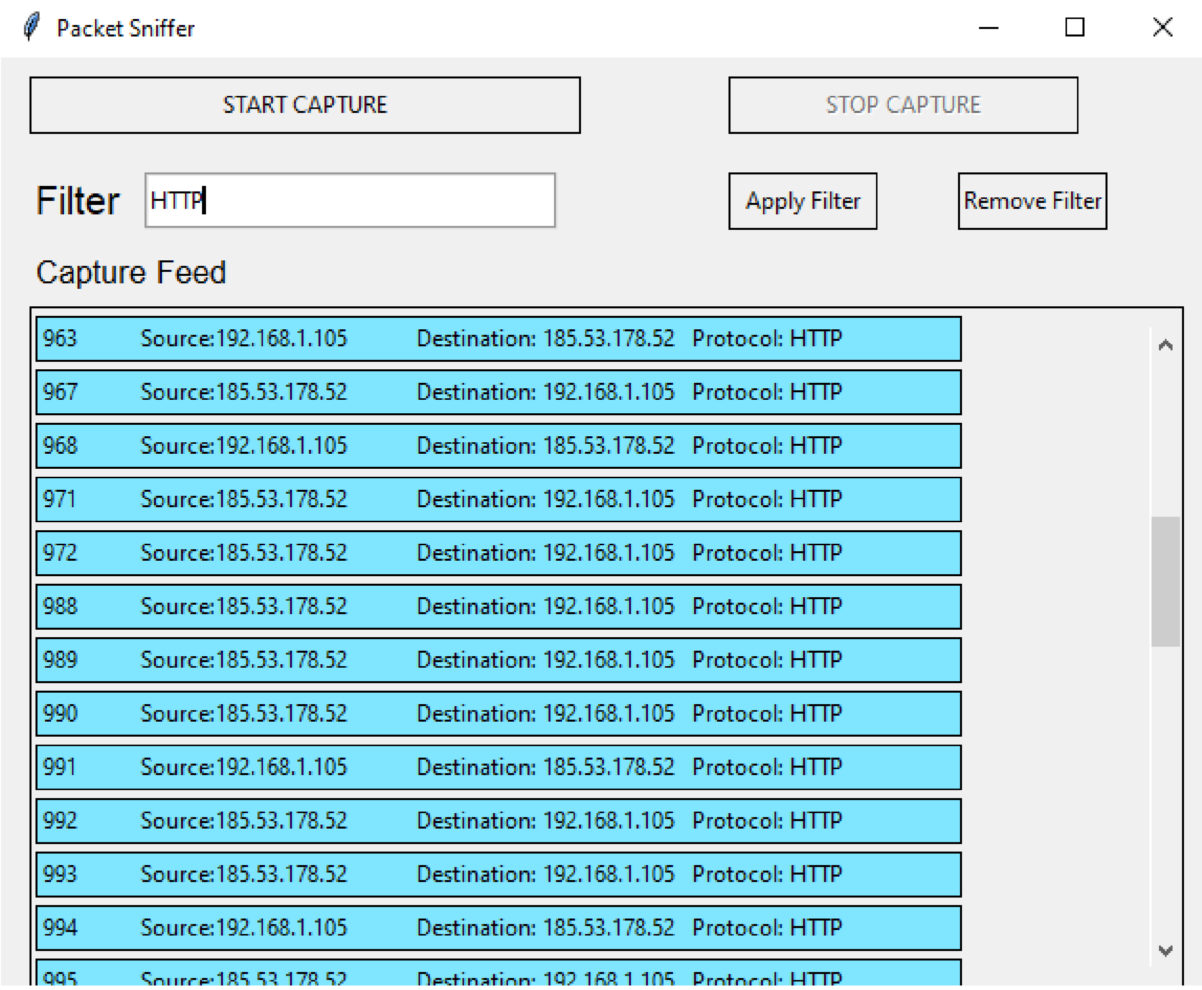
Diagrammatic Representation of the Overall System

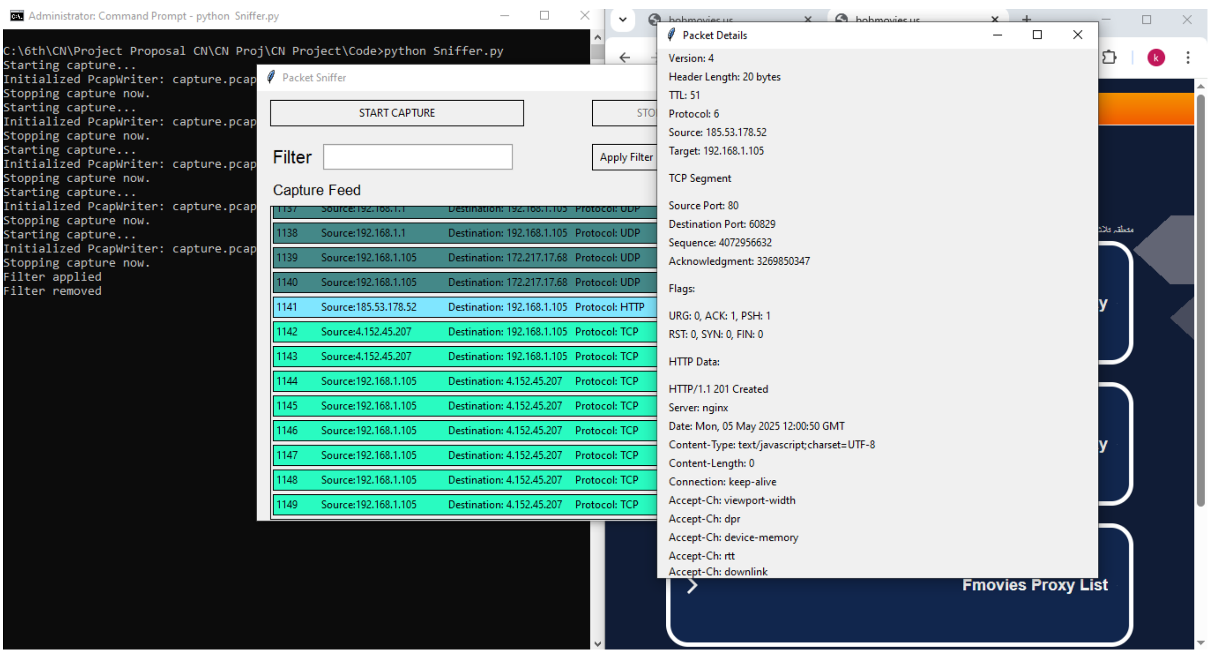


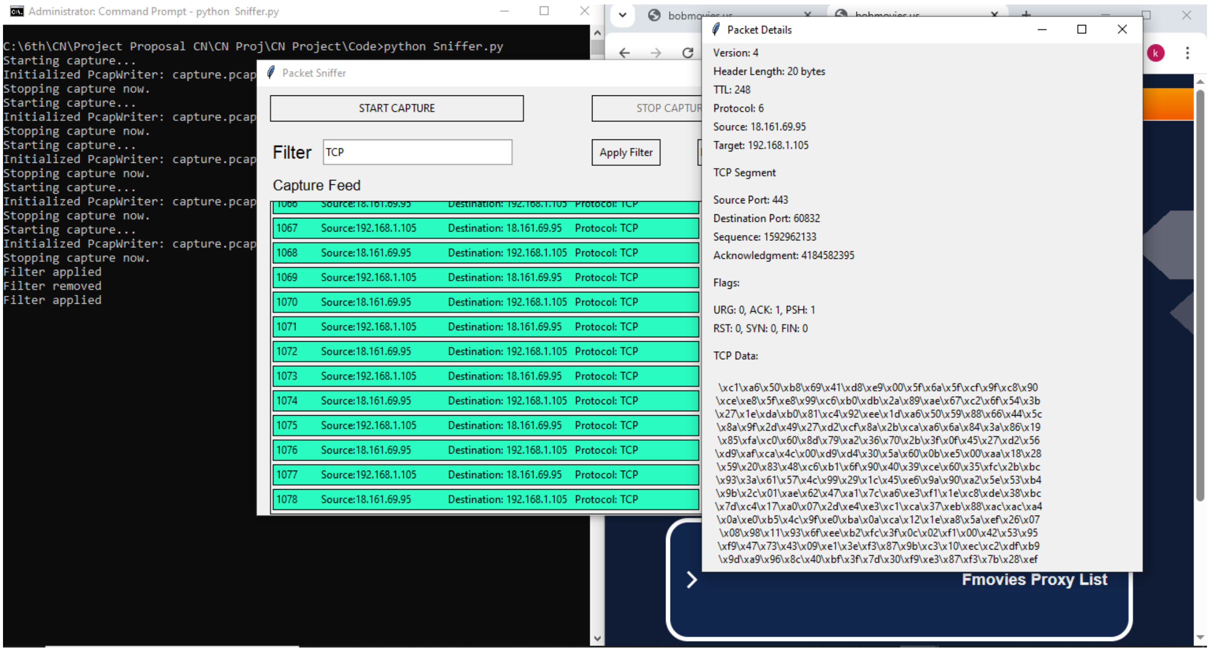


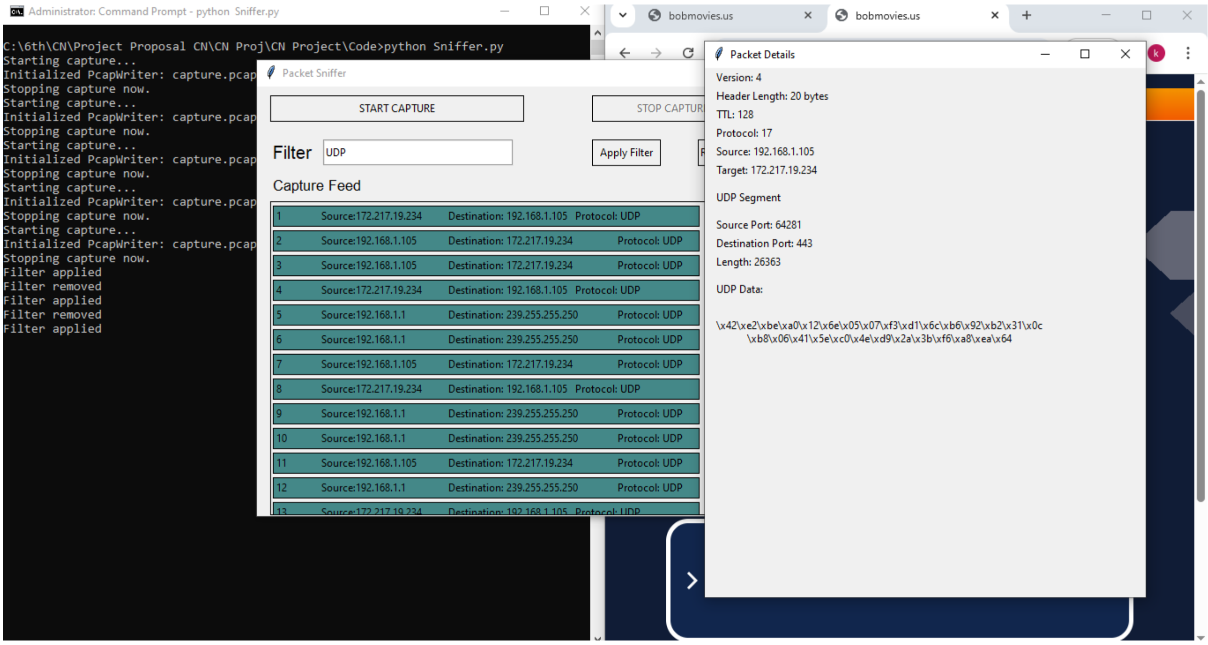

Administrator: Command Prompt - python Sniffer.py
C:\6th\CN\Project Proposal CN\CN Proj\CN Project\Code>python Sniffer.py
Starting capture...
Initialized PcapWriter: capture.pcap
Stopping capture now.
Starting capture...
Initialized PcapWriter: capture.pcap
Stopping capture now.
Starting capture...
Initialized PcapWriter: capture.pcap
Stopping capture now.
Starting capture...
Initialized PcapWriter: capture.pcap
Stopping capture now.
Starting capture...
Initialized PcapWriter: capture.pcap
Stopping capture now.
bobmovies.us
bobmovies.us
Not secure
www1.bobmovies.us
Buy this domain. | More domains at Seo.Domains
bobmovies.us
Gomovies Proxy
Fmovies Proxy
Fmovies Proxy List
Packet Sniffer
START CAPTURE
STOP CAPTURE
Filter
Apply Filter
Remove Filter
Capture Feed
1111 Source:192.168.1.105 Destination: 4.152.45.207 Protocol: TCP
1112 Source:192.168.1.105 Destination: 4.152.45.207 Protocol: TCP
1113 Source:192.168.1.105 Destination: 4.152.45.207 Protocol: TCP
1114 Source:192.168.1.105 Destination: 4.152.45.207 Protocol: TCP
1115 Source:192.168.1.105 Destination: 4.152.45.207 Protocol: TCP
1116 Source:192.168.1.105 Destination: 185.53.178.52 Protocol: HTTP
1117 Source:18.161.69.95 Destination: 192.168.1.105 Protocol: TCP
1118 Source:18.161.69.95 Destination: 192.168.1.105 Protocol: TCP
1119 Source:18.161.69.95 Destination: 192.168.1.105 Protocol: TCP
1120 Source:18.161.69.95 Destination: 192.168.1.105 Protocol: TCP
1121 Source:18.161.69.95 Destination: 192.168.1.105 Protocol: TCP
1122 Source:192.168.1.105 Destination: 18.161.69.95 Protocol: TCP

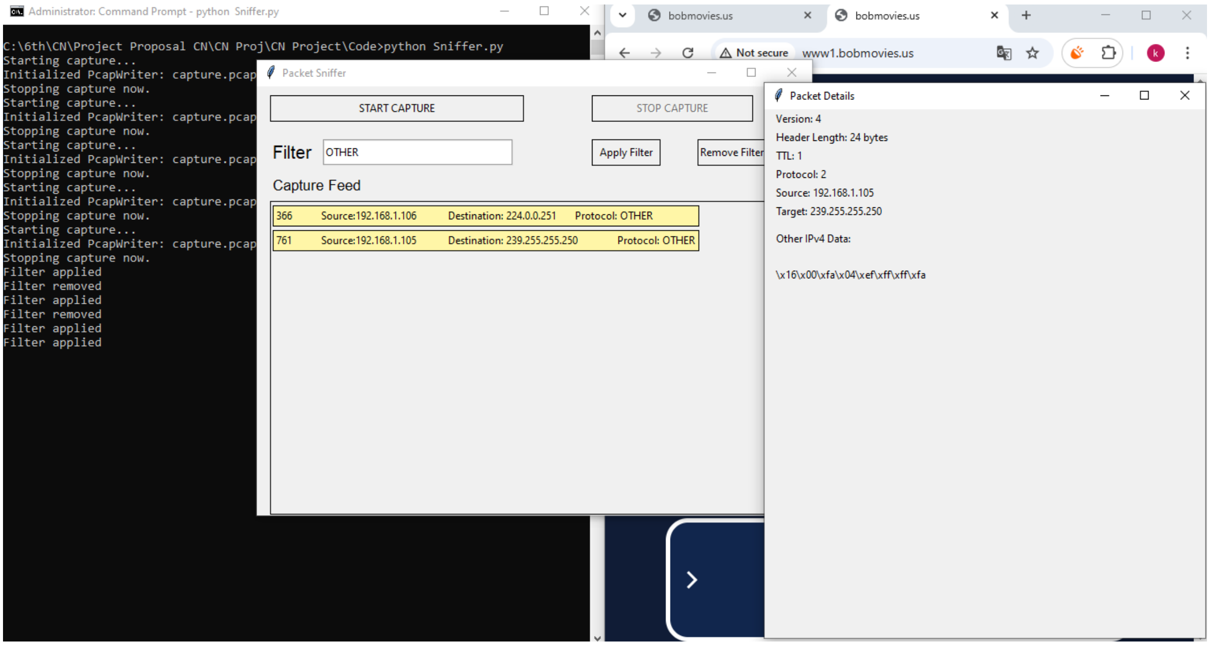

Command Prompt
C:\Users\Tesla Laptops>nslookup http://www1.bobmovies.us
Server: UnKnown
Address: 192.168.1.1
Non-authoritative answer:
Name: http://www1.bobmovies.us
Address: 185.53.178.52
C:\Users\Tesla Laptops>
Packet Sniffer
START CAPTURE
STOP CAPTURE
Filter
Apply Filter
Remove Filter
Capture Feed
1111 Source:192.168.1.105 Destination: 4.152.45.207 Protocol: TCP
1112 Source:192.168.1.105 Destination: 4.152.45.207 Protocol: TCP
1113 Source:192.168.1.105 Destination: 4.152.45.207 Protocol: TCP
1114 Source:192.168.1.105 Destination: 4.152.45.207 Protocol: TCP
1115 Source:192.168.1.105 Destination: 4.152.45.207 Protocol: TCP
1116 Source:192.168.1.105 Destination: 185.53.178.52 Protocol: HTTP
1117 Source:18.161.69.95 Destination: 192.168.1.105 Protocol: TCP
1118 Source:18.161.69.95 Destination: 192.168.1.105 Protocol: TCP
1119 Source:18.161.69.95 Destination: 192.168.1.105 Protocol: TCP
1120 Source:18.161.69.95 Destination: 192.168.1.105 Protocol: TCP
1121 Source:18.161.69.95 Destination: 192.168.1.105 Protocol: TCP
1122 Source:192.168.1.105 Destination: 18.161.69.95 Protocol: TCP

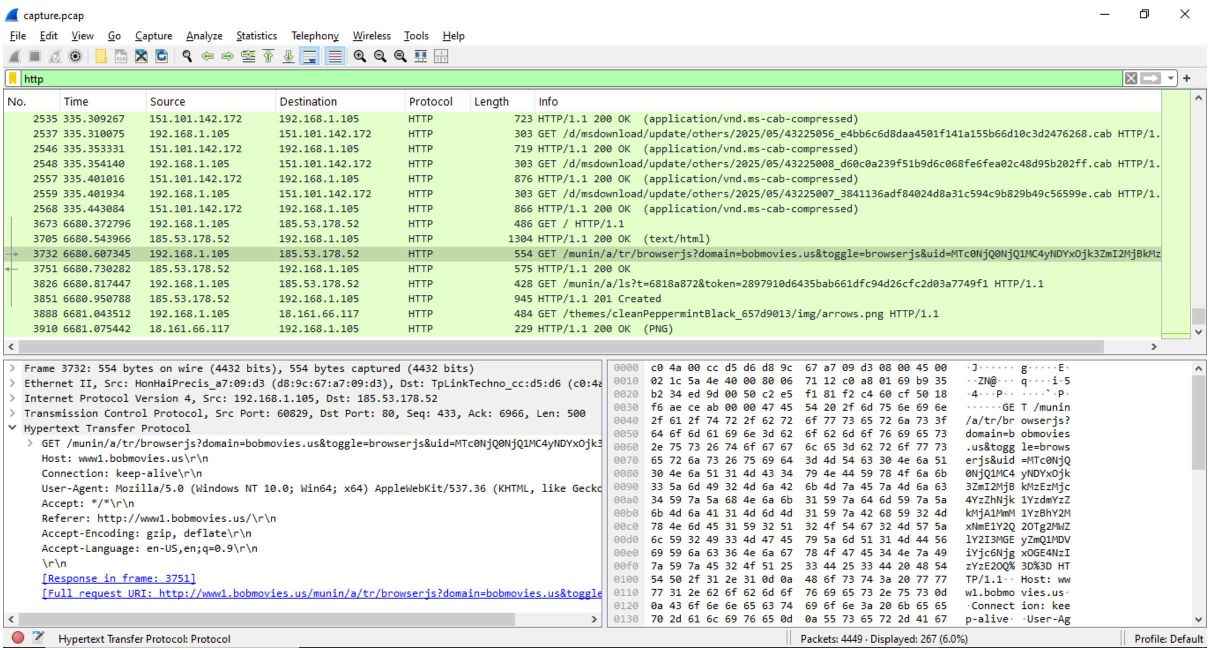

Packet Sniffer
START CAPTURE
STOP CAPTURE
Filter
HTTP
Apply Filter
Remove Filter
Capture Feed
963 Source:192.168.1.105 Destination: 185.53.178.52 Protocol: HTTP
967 Source:185.53.178.52 Destination: 192.168.1.105 Protocol: HTTP
968 Source:192.168.1.105 Destination: 185.53.178.52 Protocol: HTTP
971 Source:185.53.178.52 Destination: 192.168.1.105 Protocol: HTTP
972 Source:185.53.178.52 Destination: 192.168.1.105 Protocol: HTTP
988 Source:185.53.178.52 Destination: 192.168.1.105 Protocol: HTTP
989 Source:185.53.178.52 Destination: 192.168.1.105 Protocol: HTTP
990 Source:185.53.178.52 Destination: 192.168.1.105 Protocol: HTTP
991 Source:192.168.1.105 Destination: 185.53.178.52 Protocol: HTTP
992 Source:185.53.178.52 Destination: 192.168.1.105 Protocol: HTTP
993 Source:185.53.178.52 Destination: 192.168.1.105 Protocol: HTTP
994 Source:192.168.1.105 Destination: 185.53.178.52 Protocol: HTTP


Administrator: Command Prompt - python Sniffer.py
C:\6th\CN\Project Proposal CN\CN Proj\CN Project\Code>python Sniffer.py
Starting capture...
Initialized PcapWriter: capture.pcap
Stopping capture now.
Starting capture...
Initialized PcapWriter: capture.pcap
Stopping capture now.
Starting capture...
Initialized PcapWriter: capture.pcap
Stopping capture now.
Starting capture...
Initialized PcapWriter: capture.pcap
Stopping capture now.
Starting capture...
Initialized PcapWriter: capture.pcap
Stopping capture now.
Filter applied
Filter removed
Packet Sniffer
START CAPTURE
Filter
Apply Filter
Capture Feed
1138 Source:192.168.1.1 Destination: 192.168.1.105 Protocol: UDP
1139 Source:192.168.1.105 Destination: 172.217.17.68 Protocol: UDP
1140 Source:192.168.1.105 Destination: 172.217.17.68 Protocol: UDP
1141 Source:185.53.178.52 Destination: 192.168.1.105 Protocol: HTTP
1142 Source:4.152.45.207 Destination: 192.168.1.105 Protocol: TCP
1143 Source:4.152.45.207 Destination: 192.168.1.105 Protocol: TCP
1144 Source:192.168.1.105 Destination: 4.152.45.207 Protocol: TCP
1145 Source:192.168.1.105 Destination: 4.152.45.207 Protocol: TCP
1146 Source:192.168.1.105 Destination: 4.152.45.207 Protocol: TCP
1147 Source:192.168.1.105 Destination: 4.152.45.207 Protocol: TCP
1148 Source:192.168.1.105 Destination: 4.152.45.207 Protocol: TCP
1149 Source:192.168.1.105 Destination: 4.152.45.207 Protocol: TCP
Packet Details
Version: 4
Header Length: 20 bytes
TTL: 51
Protocol: 6
Source: 185.53.178.52
Target: 192.168.1.105
TCP Segment
Source Port: 80
Destination Port: 60829
Sequence: 4072956632
Acknowledgment: 3269850347
Flags:
URG: 0, ACK: 1, PSH: 1
RST: 0, SYN: 0, FIN: 0
HTTP Data:
HTTP/1.1 201 Created
Server: nginx
Date: Mon, 05 May 2025 12:00:50 GMT
Content-Type: text/javascript;charset=UTF-8
Content-Length: 0
Connection: keep-alive
Accept-Ch: viewport-width
Accept-Ch: dpr
Accept-Ch: device-memory
Accept-Ch: rtt
Accept-Ch: downlink
Fmovies Proxy List


Administrator: Command Prompt - python Sniffer.py
C:\6th\CN\Project Proposal CN\CN Proj\CN Project\Code>python Sniffer.py
Starting capture...
Initialized PcapWriter: capture.pcap
Stopping capture now.
Starting capture...
Initialized PcapWriter: capture.pcap
Stopping capture now.
Starting capture...
Initialized PcapWriter: capture.pcap
Stopping capture now.
Starting capture...
Initialized PcapWriter: capture.pcap
Stopping capture now.
Starting capture...
Initialized PcapWriter: capture.pcap
Stopping capture now.
Filter applied
Filter removed
Filter applied
Packet Sniffer
START CAPTURE
STOP CAPTURE
Filter
TCP
Apply Filter
Capture Feed
1067 Source:192.168.1.105 Destination: 18.161.69.95 Protocol: TCP
1068 Source:18.161.69.95 Destination: 192.168.1.105 Protocol: TCP
1069 Source:192.168.1.105 Destination: 18.161.69.95 Protocol: TCP
1070 Source:18.161.69.95 Destination: 192.168.1.105 Protocol: TCP
1071 Source:192.168.1.105 Destination: 18.161.69.95 Protocol: TCP
1072 Source:18.161.69.95 Destination: 192.168.1.105 Protocol: TCP
1073 Source:192.168.1.105 Destination: 18.161.69.95 Protocol: TCP
1074 Source:18.161.69.95 Destination: 192.168.1.105 Protocol: TCP
1075 Source:192.168.1.105 Destination: 18.161.69.95 Protocol: TCP
1076 Source:18.161.69.95 Destination: 192.168.1.105 Protocol: TCP
1077 Source:192.168.1.105 Destination: 18.161.69.95 Protocol: TCP
1078 Source:18.161.69.95 Destination: 192.168.1.105 Protocol: TCP
Packet Details
Version: 4
Header Length: 20 bytes
TTL: 248
Protocol: 6
Source: 18.161.69.95
Target: 192.168.1.105
TCP Segment
Source Port: 443
Destination Port: 60832
Sequence: 1592962133
Acknowledgment: 4184582395
Flags:
URG: 0, ACK: 1, PSH: 1
RST: 0, SYN: 0, FIN: 0
TCP Data:
Fmovies Proxy List


Administrator: Command Prompt - python Sniffer.py
C:\6th\CN\Project Proposal CN\CN Proj\CN Project\Code>python Sniffer.py
Starting capture...
Initialized PcapWriter: capture.pcap
Stopping capture now.
Starting capture...
Initialized PcapWriter: capture.pcap
Stopping capture now.
Starting capture...
Initialized PcapWriter: capture.pcap
Stopping capture now.
Starting capture...
Initialized PcapWriter: capture.pcap
Stopping capture now.
Starting capture...
Initialized PcapWriter: capture.pcap
Stopping capture now.
Filter applied
Filter removed
Filter applied
Filter removed
Filter applied
bobmovies.us
bobmovies.us
Packet Sniffer
START CAPTURE
STOP CAPTURE
Filter
UDP
Apply Filter
Capture Feed
1 Source:172.217.19.234 Destination: 192.168.1.105 Protocol: UDP
2 Source:192.168.1.105 Destination: 172.217.19.234 Protocol: UDP
3 Source:192.168.1.105 Destination: 172.217.19.234 Protocol: UDP
4 Source:172.217.19.234 Destination: 192.168.1.105 Protocol: UDP
5 Source:192.168.1.1 Destination: 239.255.255.250 Protocol: UDP
6 Source:192.168.1.1 Destination: 239.255.255.250 Protocol: UDP
7 Source:192.168.1.105 Destination: 172.217.19.234 Protocol: UDP
8 Source:172.217.19.234 Destination: 192.168.1.105 Protocol: UDP
9 Source:192.168.1.1 Destination: 239.255.255.250 Protocol: UDP
10 Source:192.168.1.1 Destination: 239.255.255.250 Protocol: UDP
11 Source:192.168.1.105 Destination: 172.217.19.234 Protocol: UDP
12 Source:192.168.1.1 Destination: 239.255.255.250 Protocol: UDP
Packet Details
Version: 4
Header Length: 20 bytes
TTL: 128
Protocol: 17
Source: 192.168.1.105
Target: 172.217.19.234
UDP Segment
Source Port: 64281
Destination Port: 443
Length: 26363
UDP Data:


Administrator: Command Prompt - python Sniffer.py
C:\6th\CN\Project Proposal CN\CN Proj\CN Project\Code>python Sniffer.py
Starting capture...
Initialized PcapWriter: capture.pcap
Stopping capture now.
Starting capture...
Initialized PcapWriter: capture.pcap
Stopping capture now.
Starting capture...
Initialized PcapWriter: capture.pcap
Stopping capture now.
Starting capture...
Initialized PcapWriter: capture.pcap
Stopping capture now.
Starting capture...
Initialized PcapWriter: capture.pcap
Stopping capture now.
Filter applied
Filter removed
Filter applied
Filter removed
Filter applied
Filter applied
Packet Sniffer
START CAPTURE
STOP CAPTURE
Filter
OTHER
Apply Filter
Remove Filter
Capture Feed
366 Source:192.168.1.106 Destination: 224.0.0.251 Protocol: OTHER
761 Source:192.168.1.105 Destination: 239.255.255.250 Protocol: OTHER
bobmovies.us
bobmovies.us
Not secure www1.bobmovies.us
Packet Details
Version: 4
Header Length: 24 bytes
TTL: 1
Protocol: 2
Source: 192.168.1.105
Target: 239.255.255.250
Other IPv4 Data:
\x16\x00\xfa\x04\xef\xff\xff\xfa


capture.pcap
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
http
No. Time Source Destination Protocol Length Info
2535 335.309267 151.101.142.172 192.168.1.105 HTTP 723 HTTP/1.1 200 OK (application/vnd.ms-cab-compressed)
2537 335.310075 192.168.1.105 151.101.142.172 HTTP 303 GET /d/msdownload/update/others/2025/05/43225056_e4bb6c6d8daa4501f141a155b66d10c3d2476268.cab HTTP/1.
2546 335.353331 151.101.142.172 192.168.1.105 HTTP 719 HTTP/1.1 200 OK (application/vnd.ms-cab-compressed)
2548 335.354140 192.168.1.105 151.101.142.172 HTTP 303 GET /d/msdownload/update/others/2025/05/43225008_d60c0a239f51b9d6c068fe6fea02c48d95b202ff.cab HTTP/1.
2557 335.401016 151.101.142.172 192.168.1.105 HTTP 876 HTTP/1.1 200 OK (application/vnd.ms-cab-compressed)
2559 335.401934 192.168.1.105 151.101.142.172 HTTP 303 GET /d/msdownload/update/others/2025/05/43225007_3841136adf84024d8a31c594c9b829b49c56599e.cab HTTP/1.
2568 335.443084 151.101.142.172 192.168.1.105 HTTP 866 HTTP/1.1 200 OK (application/vnd.ms-cab-compressed)
3673 6680.372796 192.168.1.105 185.53.178.52 HTTP 486 GET / HTTP/1.1
3705 6680.543966 185.53.178.52 192.168.1.105 HTTP 1304 HTTP/1.1 200 OK (text/html)
3732 6680.607345 192.168.1.105 185.53.178.52 HTTP 554 GET /munin/a/tr/browserjs?domain=bobmovies.us&toggle=browserjs&uid=MTc0NjQ0NjQ1MC4yNDYxOjk3ZmI2MjBkMz
3751 6680.730282 185.53.178.52 192.168.1.105 HTTP 575 HTTP/1.1 200 OK
3826 6680.817447 192.168.1.105 185.53.178.52 HTTP 428 GET /munin/a/ls?t=6818a872&token=2897910d6435bab661dfc94d26cfc2d03a7749f1 HTTP/1.1
3851 6680.950788 185.53.178.52 192.168.1.105 HTTP 945 HTTP/1.1 201 Created
3888 6681.043512 192.168.1.105 18.161.66.117 HTTP 484 GET /themes/cleanPeppermintBlack_657d9013/img/arrows.png HTTP/1.1
3910 6681.075442 18.161.66.117 192.168.1.105 HTTP 229 HTTP/1.1 200 OK (PNG)
Frame 3732: 554 bytes on wire (4432 bits), 554 bytes captured (4432 bits)
Ethernet II, Src: HonHaiPrecis_a7:09:d3 (d8:9c:67:a7:09:d3), Dst: TpLinkTechno_cc:d5:d6 (c0:4a
Internet Protocol Version 4, Src: 192.168.1.105, Dst: 185.53.178.52
Transmission Control Protocol, Src Port: 60829, Dst Port: 80, Seq: 433, Ack: 6966, Len: 500
Hypertext Transfer Protocol
GET /munin/a/tr/browserjs?domain=bobmovies.us&toggle=browserjs&uid=MTc0NjQ0NjQ1MC4yNDYxOjk3
Host: www1.bobmovies.us\r\n
Connection: keep-alive\r\n
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko
Accept: */*\r\n
Referer: http://www1.bobmovies.us/\r\n
Accept-Encoding: gzip, deflate\r\n
Accept-Language: en-US,en;q=0.9\r\n
\r\n
[Response in frame: 3751]
[Full request URI: http://www1.bobmovies.us/munin/a/tr/browserjs?domain=bobmovies.us&toggle
Hypertext Transfer Protocol: Protocol
Packets: 4449 · Displayed: 267 (6.0%)
Profile: Default