|  |
| --- |
| Photo displaying partial image of two pie charts on a canvas-textured page |
| Day-1  Cybersecurity analyst |
| |  |  |  | | --- | --- | --- | | MOHAN KUMAR | 4/6/25 | Wireshark | |

**ICMP Ping Traffic Analysis**

**Objective**

As part of the Mohan Cyber Schedule, a 4-month cybersecurity training plan, I set up a virtual lab in VMware to analyze network traffic. This report details the capture and examination of ICMP (Internet Control Message Protocol) packets generated by pinging from a Windows 10 VM to a Parrot OS VM, filtered and analyzed using Wireshark.

**Lab Setup**

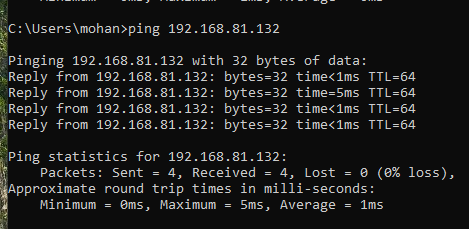
* **Environment**: VMware Workstation Player (free version).
* **Virtual Machines**:
  + Windows 10 Evaluation VM (IP: 192.168.81.128, 2GB RAM, NAT network).
  + Parrot OS Security Edition VM (IP: 192.168.81.132, 4GB RAM, NAT network).
* **Tool**: Wireshark (installed on Parrot OS via sudo apt install wireshark -y).

**Methodology**

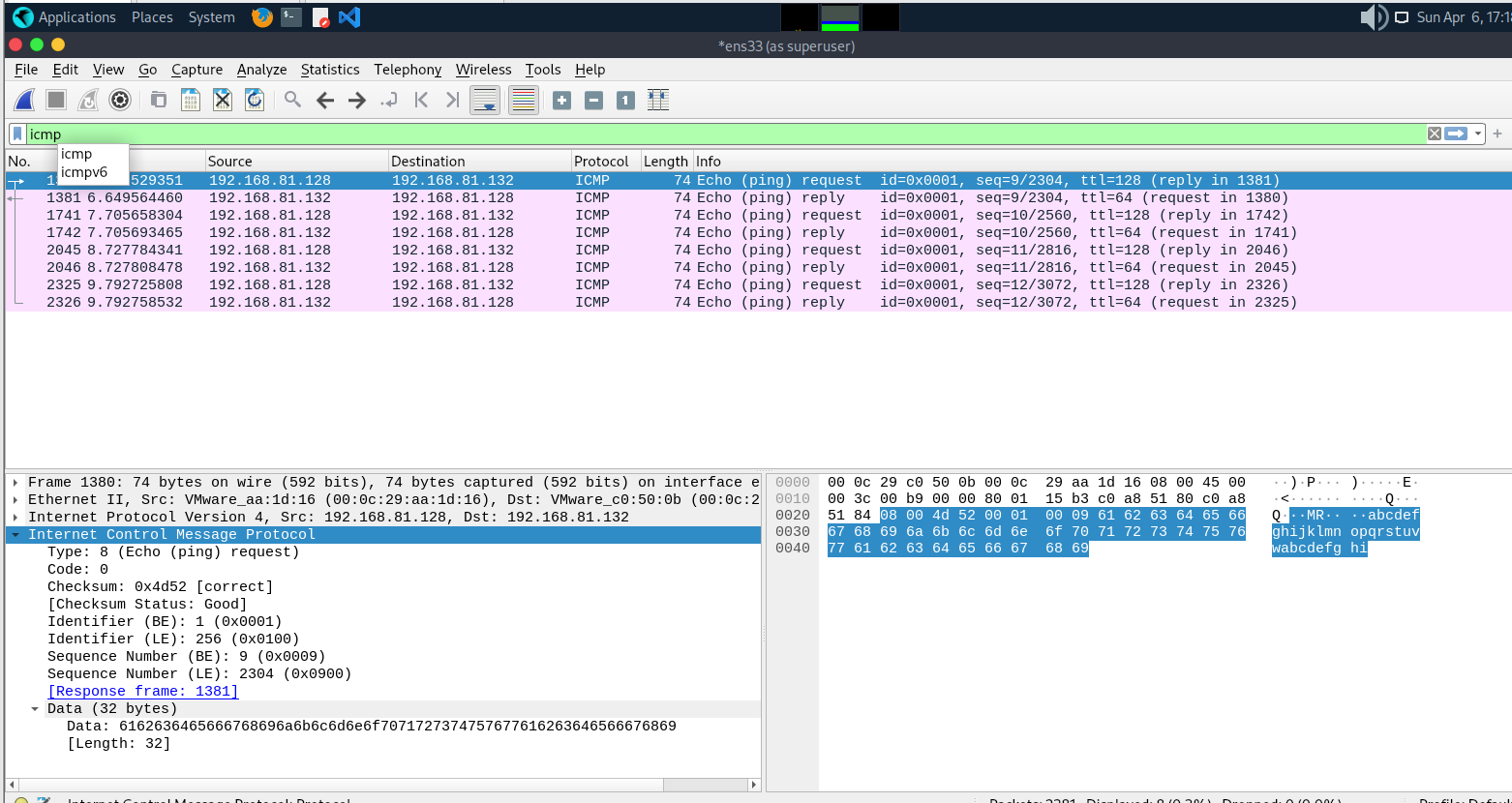
1. **Traffic Generation**: Executed ping 192.168.81.132 -n 4 from the Windows VM to send four ICMP Echo Requests to the Parrot OS VM.
2. **Capture**: Launched Wireshark on Parrot OS (sudo wireshark &), selected the active interface (e.g., ens33), captured traffic, and saved it as day1\_ping.pcap.
3. **Analysis**: Applied an icmp filter in Wireshark to isolate ping packets and inspected key fields.

**Observations**

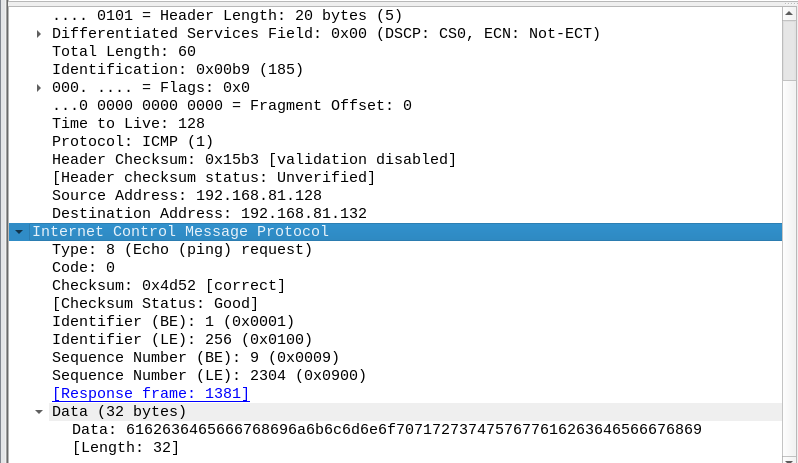
The day1\_ping.pcap file contained eight packets: four Echo Requests and four Echo Replies. Key findings include:



* **Packet Sequence**: Alternating requests (192.168.81.128 → 192.168.81.132) and replies (192.168.81.132 → 192.168.81.128).
* **ICMP Details**:
  + Requests: Type 8, Code 0 (Echo Request).
  + Replies: Type 0, Code 0 (Echo Reply).
  + Payload: Sequential ASCII (e.g., “abcd…”).



* **IP Headers**: TTL 128 (requests), 127 (replies, one hop decremented).
* **Timing**: Delta ~1-5ms between request-reply pairs.



* **Packet Size**: ~74 bytes per packet (IP/ICMP headers + data).

This analysis highlights ICMP’s role in verifying connectivity. The predictable request-reply pattern and consistent packet structure establish a baseline for normal traffic, a critical skill for identifying anomalies in cybersecurity roles like SOC Analyst.

**HTTP Traffic Analysis**.

**Objective**

On Day 1 of the Mohan Cyber Schedule, I expanded my network analysis skills by capturing and dissecting HTTP traffic in my VMware lab. This report covers the process of generating HTTP requests from a Windows 10 VM, capturing them with Wireshark on Parrot OS, and analyzing the resulting packets.

**Lab Setup**

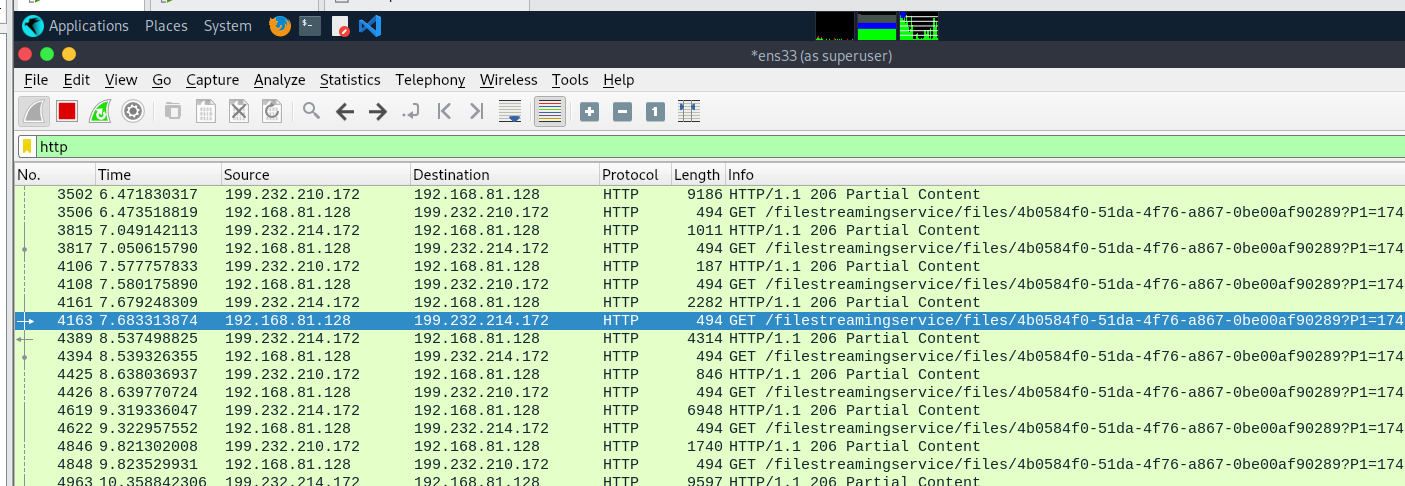
* **Environment**: VMware Workstation Player (free version).
* **Virtual Machines**:
  + Windows 10 Evaluation VM (IP: 192.168.81.128, 2GB RAM, NAT network).
  + Parrot OS Security Edition VM (IP: 192.168.81.132, 4GB RAM, NAT network).
* **Tool**: Wireshark (installed on Parrot OS via sudo apt install wireshark -y).

**Methodology**

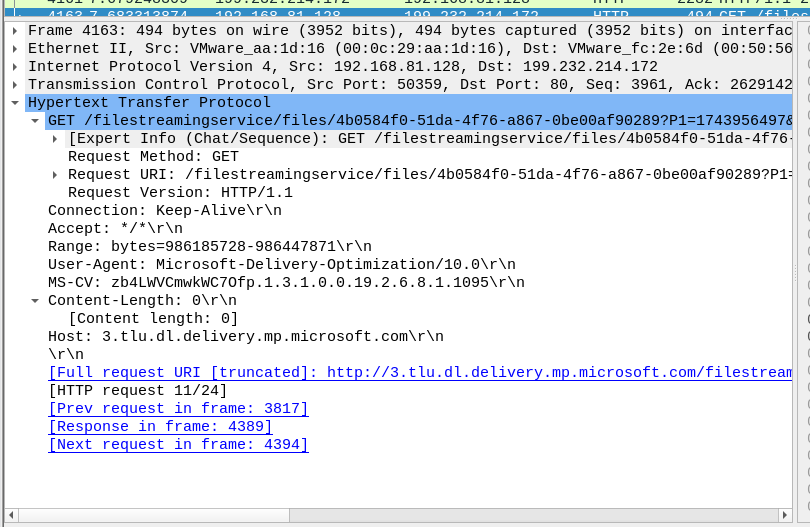
1. **Traffic Generation**: From the Windows VM, executed curl google.com in Command Prompt to generate an HTTP GET request to Google’s server.
2. **Capture**: Ran Wireshark on Parrot OS (sudo wireshark &), selected the active interface (e.g., ens33), captured traffic, and saved it as day1\_http.pcap.
3. **Analysis**: Applied an http filter in Wireshark, followed a TCP stream, and examined packet details.

**Observations**

The day1\_http.pcap file captured the HTTP transaction, including TCP handshake and data exchange. Key findings include:



* **TCP Handshake**: Three packets (SYN, SYN-ACK, ACK) between 192.168.81.128 and Google’s IP (e.g., 142.250.190.78).
* **HTTP Request**:
  + Method: GET, Host: google.com, sent from 192.168.81.128.
  + Port: Source (ephemeral, e.g., 49152), Destination (80).
* **HTTP Response**:
  + Status: 200 OK or 301 Moved (Google’s redirect).
  + Payload: HTML content (e.g., redirect to [www.google.com](http://www.google.com)).



* **Timing**: Delta ~20-50ms for response, depending on network latency.
* **Packet Size**: Request ~300 bytes, Response ~500-1000 bytes (varies with content).

This exercise demonstrated HTTP’s reliance on TCP for reliable data transfer, contrasting with ICMP’s simplicity. Following a TCP stream revealed the client-server interaction, a key skill for analyzing web-based threats in cybersecurity.