NITTE MEENAKSHI INSTITUTE OF TECHNOLOGY
(AN AUTONOMOUS INSTITUTION, AFFILIATED TO VISVESVARAYA TECHNOLOGICAL UNIVERSITY, BELGAUM, APPROVED BY AICTE & GOVT.OF KARNATAKA

COMPUTER NETWORKS PROJECT

ON

"NETWORK TRAFFIC MONITORING USING WIRESHARK"

Submitted in partial fulfillment of the requirement for the award of Degree of Bachelor of Engineering

in

Computer Science and Engineering

Submitted by:

AKANKSHA KASHYAP	1NT21CS021
AKRUTI SARANGI	1NT21CS024
ANUSHREE L	1NT21CS039
HIMANSHI YADAV	1NT21CS073

Under the Guidance of
Mrs. Deepthi Shetty
Assistant Professor, Dept. of CSE, NMIT

Department of Computer Science and Engineering
(Accredited by NBA Tier-1)

2023-2024

NITTE MEENAKSHI INSTITUTE OF TECHNOLOGY
(AN AUTONOMOUS INSTITUTION, AFFILIATED TO VISVESVARAYA TECHNOLOGICAL UNIVERSITY, BELGAUM
, APPROVED BY AICTE & GOVT.OF KARNATAKA)
Department of Computer Science and Engineering
(Accredited by NBA Tier-1)

CERTIFICATE

This is to certify that the **Network Traffic Monitoring using Wireshark** is an authentic work carried out by **Akanksha Kashyap (1NT21CS021)**, **Akruti Sarangi (1NT21CS024), Anushree L (1NT21CS039), Himanshi Yadav (1NT21CS073)** and bona fide students of **Nitte Meenakshi Institute of Technology**, Bangalore in partial fulfillment for the award of the degree of ***Bachelor of Engineering*** in COMPUTER SCIENCE AND ENGINEERING of Visvesvaraya Technological University, Belagavi during the academic year ***2023-2024.*** It is certified that all corrections and suggestions indicated during the internal assessment has been incorporated in the report. This project has been approved as it satisfies the academic requirement in respect of project work presented for the said degree.

Internal Guide	**Signature of the HOD**	**Signature of Principal**
_____	_____	_____
Mrs. Deepthi Shetty Assistant Professor, Dept. CSE, NMIT Bangalore	Dr. Vijaya Shetty S. Professor, Head, Dept. CSE, NMIT Bangalore	Dr. H. C. Nagaraj Principal, NMIT Bangalore

DECLARATION

We hereby declare that Learning activity project work

(i) The project work is our original work

(ii) This Project work has not been submitted for the award of any degree or examination at any other university/College/Institute.

(iii) This Project Work does not contain other persons' data, pictures, graphs, or other information, unless specifically acknowledged as being sourced from other persons.

(iv) This Project Work does not contain other persons' writing, unless specifically acknowledged as being sourced from other researchers. Where other written sources have been quoted, then:

a) their words have been re-written, but the general information attributed to them has been referenced.

b) where their exact words have been used, their writing has been placed inside quotation marks, and referenced.

(v) This Project Work does not contain text, graphics or tables copied and pasted from the Internet, unless specifically acknowledged, and the source being detailed in the thesis and in the References sections.

NAME	**USN**	**SIGNATURE**
AKANKSHA KASHYAP	1NT21CS021	
AKRUTI SARANGI	1NT21CS024	
ANUSHREE L	1NT21CS039	
HIMANSHI YADAV	1NT21CS073	

Date: 7/01/2024

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of the people who made it possible, whose constant guidance and encouragement crowned our effort with success. We express my sincere gratitude to our Principal **Dr. H. C. Nagaraj**, Nitte Meenakshi Institute of Technology for providing facilities.

We wish to thank our HOD**, Dr. Vijaya Shetty S.** for the excellent environment created to further educational growth in our college. We also thank her for the invaluable guidance provided which has helped in the creation of a better project.

We hereby like to thank **Mrs. Deepthi Shetty**, Assistant Professor, Department of Computer Science & Engineering, on her periodic inspection, time to time evaluation of the project, and help to bring the project to the present form.

ABSTRACT

This report provides a succinct exploration of Wireshark, an open-source packet analyzer widely employed for network traffic analysis. It covers the configuration process, highlighting the tool's cross-platform compatibility and user-friendly interface. Core functionalities, such as packet capture, filtering, and protocol analysis, are outlined, showcasing Wireshark's ability to dissect intricate network communications.

The report delves into advanced features, includes analyzing multimedia streams, emphasizing Wireshark's pivotal role for network administrators and security analysts. Real-world case studies exemplify its practical applications in diagnosing connectivity issues and ensuring data transmission security.

Wireshark's versatility is underscored through various display and filtering options, allowing users to focus on specific network events. The report highlights the tool's significance in identifying anomalies, diagnosing performance issues, and ensuring data integrity. Aimed at network administrators, this report focuses on effectively leveraging Wireshark for gaining valuable insights into network dynamics.

TABLE OF CONTENTS

INTRODUCTION

Computer networks are the foundation of the linked society we live in today, where information travels easily between devices and continents thanks to the digital revolution. A computer network is a sophisticated infrastructure that facilitates communication and resource sharing by connecting computers, servers, and other devices.

Wireshark, an open-source network protocol analyzer, plays a pivotal role in the realm of network analysis. Formerly known as Ethereal, it is a go-to tool for network administrators, security professionals, and enthusiasts. Wireshark captures and examines real-time data packets within computer networks, offering valuable insights into network behavior. Its user-friendly interface facilitates both graphical and detailed packet-level analysis, supporting a diverse range of network protocols. Whether used for troubleshooting, security assessment, or protocol analysis, Wireshark remains an essential resource. In an era of advancing technology and evolving network complexities, Wireshark stands as a fundamental instrument for understanding, optimizing, and securing the intricate web of connections that define our digital world. This introduction sets the stage for delving into the functionalities and significance of Wireshark in the dynamic landscape of network diagnostics.

Objective:

The primary aim of this report is to conduct an in-depth exploration of Wireshark, a powerful network protocol analyzer. Through a systematic investigation, it is possible to seek and unravel the intricacies of Wireshark's features, functionalities, and applications, with a specific focus on its utility in network troubleshooting, security analysis, and protocol debugging. By delving into exploring Wireshark, we aim to provide a comprehensive understanding of Wireshark's potential impact on enhancing network performance, identifying security vulnerabilities.

IMPORTANCE

Wireshark holds significant importance in computer networks for various reasons, making it an invaluable tool for network administrators, security professionals, and anyone involved in managing and analyzing network traffic. Here are key reasons why Wireshark is crucial in computer networks:

1. Network Troubleshooting:

- Wireshark allows for real-time packet-level analysis, enabling the identification and resolution of network issues such as latency, packet loss, and connectivity problems.
- It provides detailed information about each packet, helping network administrators pinpoint the source of problems quickly.

2. Security Analysis:

- Wireshark is instrumental in detecting and analyzing security threats, intrusions, and malicious activities on the network.
- Security professionals can use Wireshark to examine traffic patterns, identify anomalies, and uncover potential vulnerabilities in the network.

3. Protocol Analysis:

- Wireshark supports the decoding and analysis of a wide range of network protocols, providing insights into how different devices and applications communicate.
- It helps ensure proper protocol implementation and aids in diagnosing issues related to protocol misconfigurations.

4. Performance Optimization:

- By capturing and analyzing network traffic, Wireshark helps optimize network performance by identifying bottlenecks, optimizing bandwidth usage, and streamlining communication patterns.
- Administrators can make informed decisions to enhance the overall efficiency of the network.

5. Network Monitoring:

- Wireshark serves as a powerful network monitoring tool, offering a comprehensive view of the data flowing through the network.
- It enables administrators to observe patterns, trends, and usage statistics, facilitating effective capacity planning and resource allocation.

6. Education and Training:

- Wireshark is widely used for educational purposes, providing students and professionals with hands-on experience in understanding how networks function.
- It serves as a valuable training tool for individuals learning about networking protocols, packet structures, and network analysis techniques.

7. Troubleshooting Security Incidents:

- In the event of a security incident, Wireshark helps with forensic analysis by capturing and analyzing network traffic during the incident.
- This aids in understanding the nature of the attack, identifying compromised systems, and implementing necessary security measures.

8. Compliance and Auditing:

- Wireshark assists in ensuring network compliance with industry regulations and standards by allowing organizations to monitor and analyze network traffic.
- It can be used to generate reports and logs for auditing purposes.

9. Proactive Threat Identification:

- Wireshark facilitates proactive threat identification by allowing administrators to set up alerts and triggers based on specific network behavior patterns.
- This helps in identifying potential security threats before they escalate.

10. Open Source and Community Support:

- Wireshark is an open-source tool with a large and active community. This ensures continuous development, regular updates, and a wealth of community-contributed plugins and resources.

In summary, Wireshark is a versatile and powerful tool that plays a crucial role in enhancing the reliability, security, and efficiency of computer networks. Its ability to provide detailed insights into network behavior, troubleshoot issues, and contribute to security measures makes it an indispensable asset for network professionals.

KEY FEATURES

- **Live Packet Capture:**
  Allows real-time capture and analysis of network traffic, providing instant visibility into ongoing communication.

- **Customizable Display Filters:**
  Enables users to focus on specific packets or types of traffic, streamlining the analysis process and making it easier to identify relevant information.

- **Graphical User Interface (GUI):**
  Intuitive and user-friendly interface that displays captured data in an organized and visually accessible manner.

- **Packet-Level Details:**
  Provides in-depth information about each packet, including headers, payloads, source/destination addresses, and timing details.

- **Protocol Hierarchy:**
  Presents network traffic in a hierarchical structure, allowing users to quickly grasp the relationships between different protocols in a communication session.

- **Conversation Tracking:**
  Facilitates the analysis of conversations between network hosts, offering insights into communication patterns.

- **Colorization and Visualization:**
  Uses colorization to visually differentiate between different types of packets, enhancing the interpretability of captured data.

- **Statistics and Expert Information:**
  Offers detailed statistics and expert system information to help identify issues, anomalies, and potential security threats.

- **Cross-Platform Compatibility:**
  Available on multiple operating systems, ensuring accessibility for users on Windows, macOS, and various Linux distributions.

- **Export Options:**
  Allows users to export captured data to different file formats, facilitating further analysis or sharing of findings.

- **Filtering and Searching:**
  Advanced filtering options and search functionality to quickly locate specific packets or information within the captured data.

- **Open Source and Community Support:**
  Being open-source, Wireshark benefits from a robust community, ensuring continuous development, support, and the availability of additional resources.

Question 1:

How can you filter and display only HTTP traffic in Wireshark?

Filtering and displaying only HTTP traffic in Wireshark is crucial for quickly isolating and examining web-related data. This focused view simplifies troubleshooting, performance optimization, and security monitoring, providing valuable insights into HTTP requests and responses without overwhelming network analysis with unnecessary details.

Steps to filter and display only HTTP traffic in Wireshark:

1. **Start Capturing Traffic:**

Before applying any filters, start capturing network traffic by selecting an appropriate network interface. Click on the interface from the list and press the "Start" or "Go" button. Let Wireshark capture packets for a short duration.

2. **Stop Capturing:**

After capturing some packets, click the "Stop" button to halt the packet capture.

3. **Apply HTTP Display Filter:**

In the Wireshark main window, locate the "Display Filter" field at the top of the screen. Type the following filter expression to display only HTTP traffic:

4. **Analyze Filtered Packets:**

Wireshark will now display only the packets that are related to HTTP. You'll see details like source and destination IP addresses, ports, HTTP methods, and more.

5. **Inspect HTTP Packets:**

You can click on individual packets to inspect their details in the middle panel. The lower panel will provide detailed information about the selected packet.

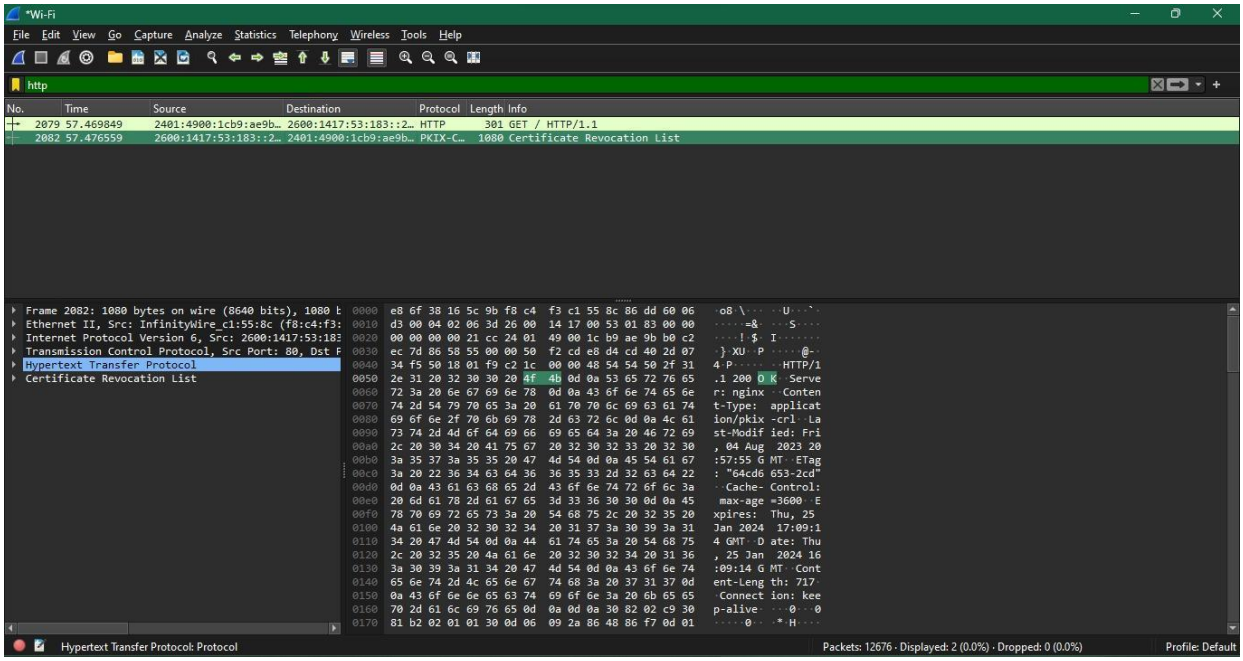

Fig. 1. Steps to filter and display only HTTP traffic

Question 2:

What is the method to find the total number of packets captured in a Wireshark session?

This information of total number of packets captured is essential for assessing the scale of network activity, diagnosing issues, and understanding the volume of data traversing the network. It provides a quick quantitative overview, aiding in efficient network analysis and troubleshooting.

Steps to find the total number of packets captured in Wireshark:

1. **Select Network Interface:**

In the main Wireshark window, you'll see a list of available network interfaces. Choose the one you want to capture traffic from and click on it.

2. **Start Capturing:**

Click the "Start" or "Go" button to begin capturing network packets. Wireshark will start displaying packets in real-time.

3. **Capture Packets:**

Allow Wireshark to capture packets for the desired duration or until you've captured enough data for analysis.

4. **Stop Capturing:**

Click the "Stop" button to halt the packet capture when you're ready to analyze the collected data.

5. **View Summary Information:**

In the main Wireshark window, look for the summary section at the bottom. It provides essential statistics, including the total number of packets captured.

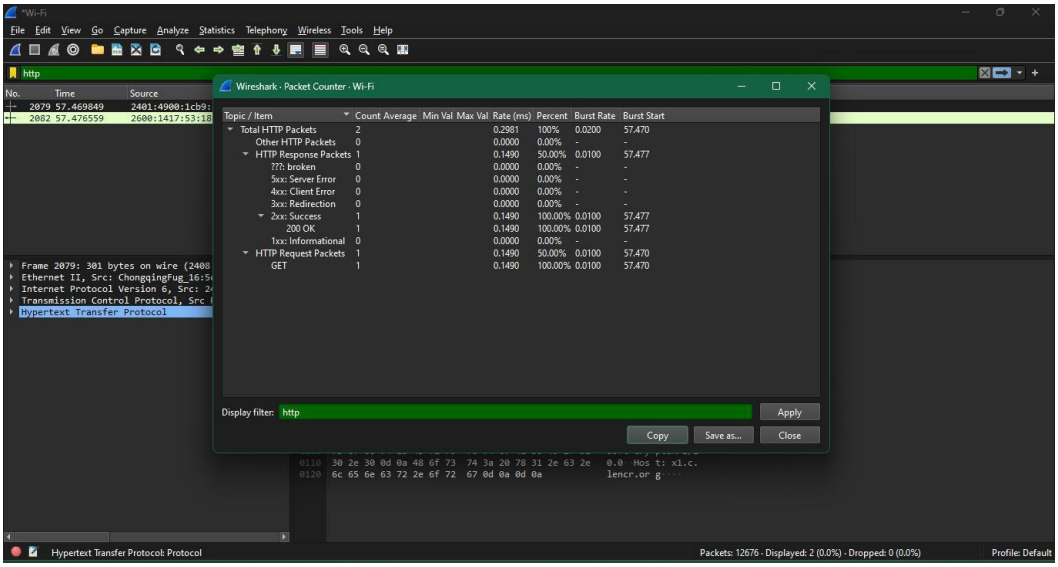

Fig. 2. HTTP packet counter

6. Select Capture File Properties:

In the "Statistics" menu, choose "Capture File Properties." A new window will appear displaying various statistics about the captured data.

7. View Packet Count:

Look for the "Number of packets" field in the "Capture File Properties" window. This field indicates the total number of packets captured during the Wireshark session.

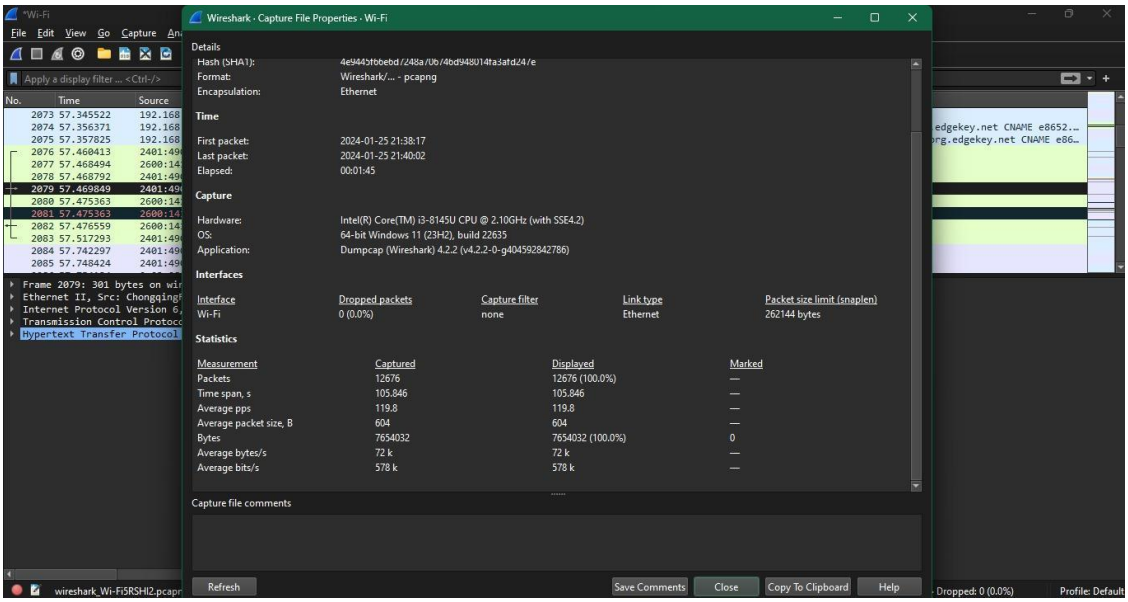

Fig. 3. Total number of packets captured (with HTTP filter)

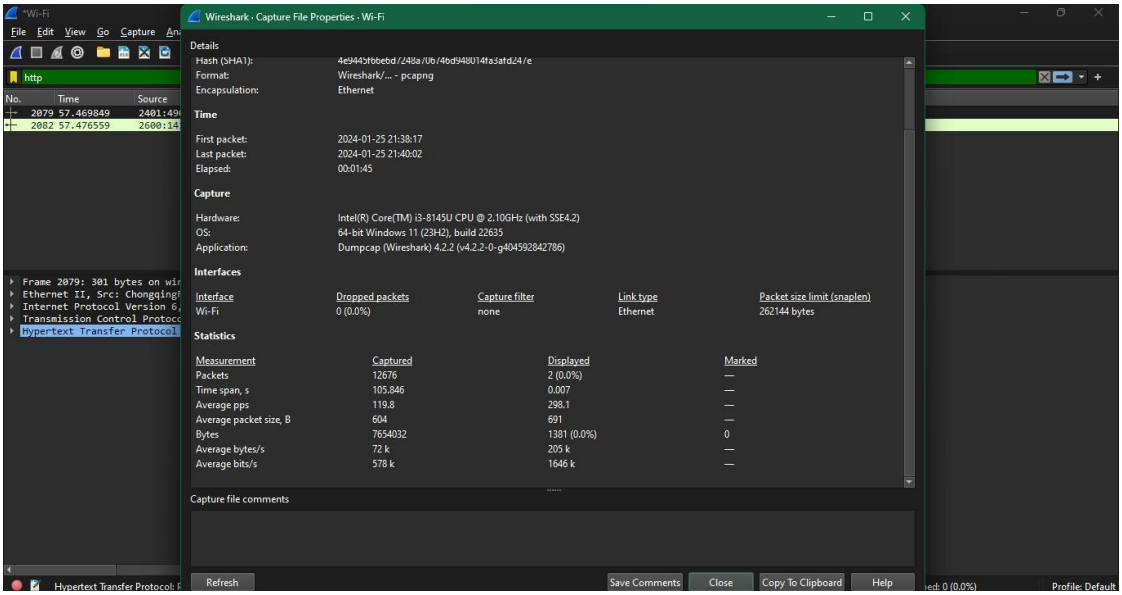

Fig. 4. Total number of packets captured (without filter)

Question 3:

How do you view conversations in Wireshark, and what information does it provide?

This view of conversation is crucial for understanding network dynamics, identifying top talkers, and pinpointing communication patterns efficiently.

Steps to view conversations in Wireshark:

1. **Capture Packets:**

Allow Wireshark to capture packets for the desired duration or until you've collected enough data.

2. **Stop Capturing:**

Click the "Stop" button to halt the packet capture when you're ready to analyze the collected data.

3. **Navigate to "Statistics" Menu:**

In the Wireshark menu bar, navigate to the "Statistics" menu.

4. **Select "Conversations":**

In the "Statistics" menu, choose "Conversations." This will open a new window displaying various conversation types.

5. **Choose Conversation Type:**

In the "Conversations" window, you can select the type of conversation you are interested in. For example, "IP," "TCP," or "UDP" conversations.

6. **View Conversation Details:**

After selecting a conversation type, Wireshark will list all the conversations of that type. You can see details such as source and destination IP addresses, source and destination ports, the number of packets, and the total data exchanged.

7. Select a conversation:

Click on a specific conversation to view detailed information about that communication. The lower panel will display packet details for the selected conversation.

Viewing conversations in Wireshark provides a structured and organized way to analyze network interactions. It helps identify communication patterns, troubleshoot issues, and gain insights into the dynamics of the network traffic.

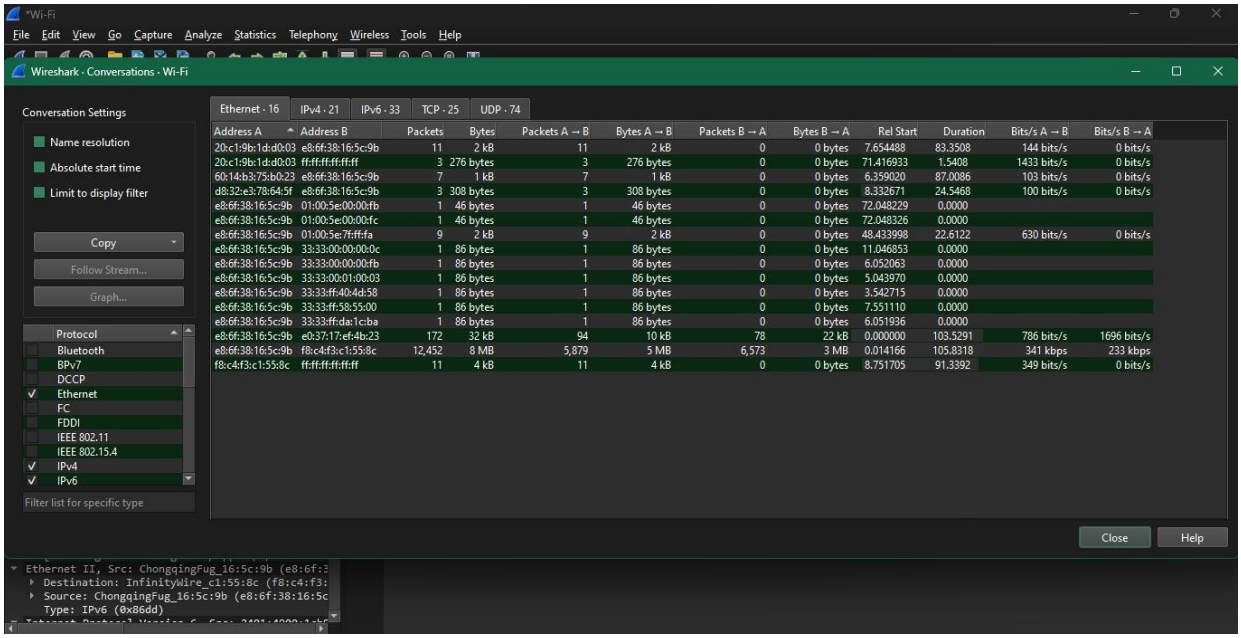

Fig. 5. Steps to view conversations in Wireshark.

Question 4:

Explain how to follow a TCP stream in Wireshark and why it is useful.

Following a TCP stream in Wireshark allows us to reconstruct and view the entire conversation between two endpoints using the TCP protocol. This feature is particularly useful for understanding the content and context of the communication, which can be valuable for troubleshooting, analysis, and security investigations.

1. **Open the Capture File:**
   Open Wireshark and load the capture file that contains the TCP traffic you want to analyze. You can do this by selecting "File" > "Open" and choosing the relevant capture file.

2. **Filter the Display:**
   Use a display filter to narrow down the displayed packets to the specific TCP conversation you are interested in. For example, if you want to follow the TCP stream between IP addresses A.B.C.D and E.F.G.H,

   Use the filter: ip.addr == A.B.C.D &&ip.addr == E.F.G.H.

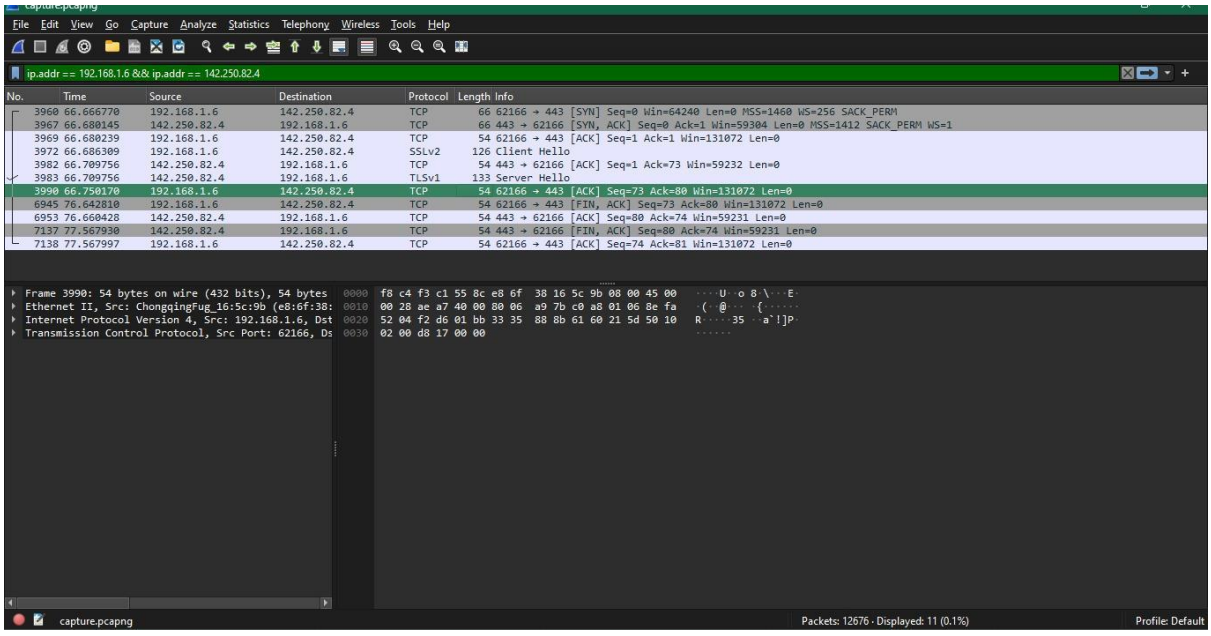

Fig. 6. Applying filter to the required IP addresses.

3. **Locate the TCP Packet:**

   Scroll through the packet list to find a packet that belongs to the TCP conversation you want to follow. Right-click on the packet and select "Follow" > "TCP Stream" from the context menu.

4. **Follow TCP Stream Window:**

   A new window will open, displaying the entire TCP stream between the selected source and destination IP addresses. The window is divided into two panes: one showing the data in ASCII format and the other in raw hexadecimal format.

5. **ASCII and Hexadecimal Views:**

   The ASCII view shows the content of the TCP stream in a human-readable format, while the hexadecimal view displays the raw binary data. This allows you to inspect both the textual and binary aspects of the communication.

6. **Scroll and Analyze:**

   Scroll through the TCP stream to analyze the entire conversation. The displayed data includes requests, responses, and any other information exchanged between the two endpoints.

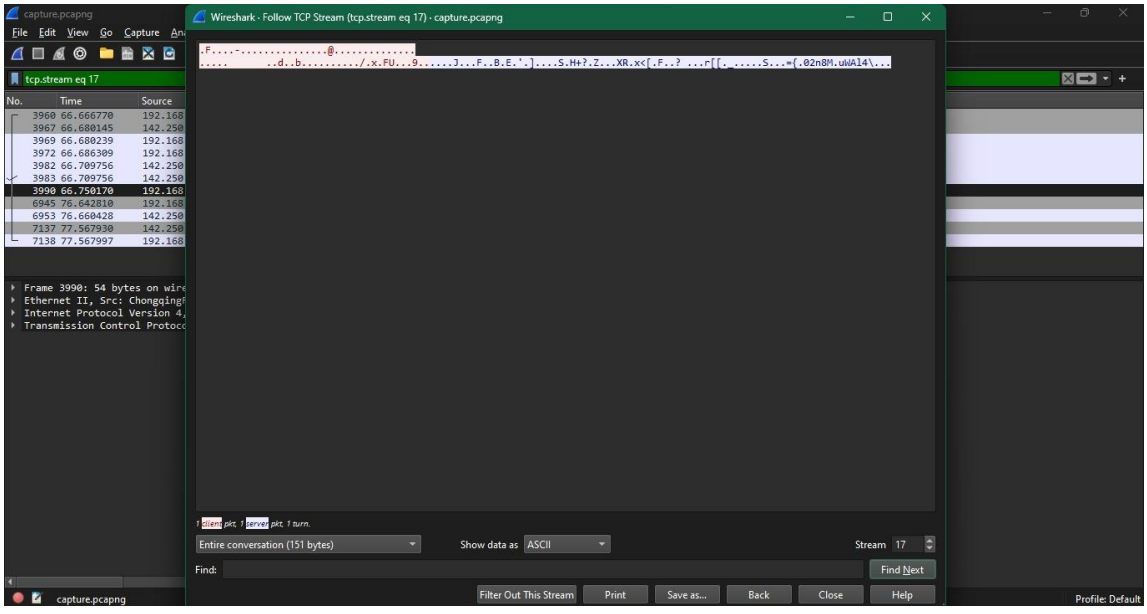

Fig. 7. TCP Stream.

7. **Save or Export Data (Optional):**

   If needed, you can save or export the content of the TCP stream. This can be useful for documentation, further analysis, or sharing information with other team members.

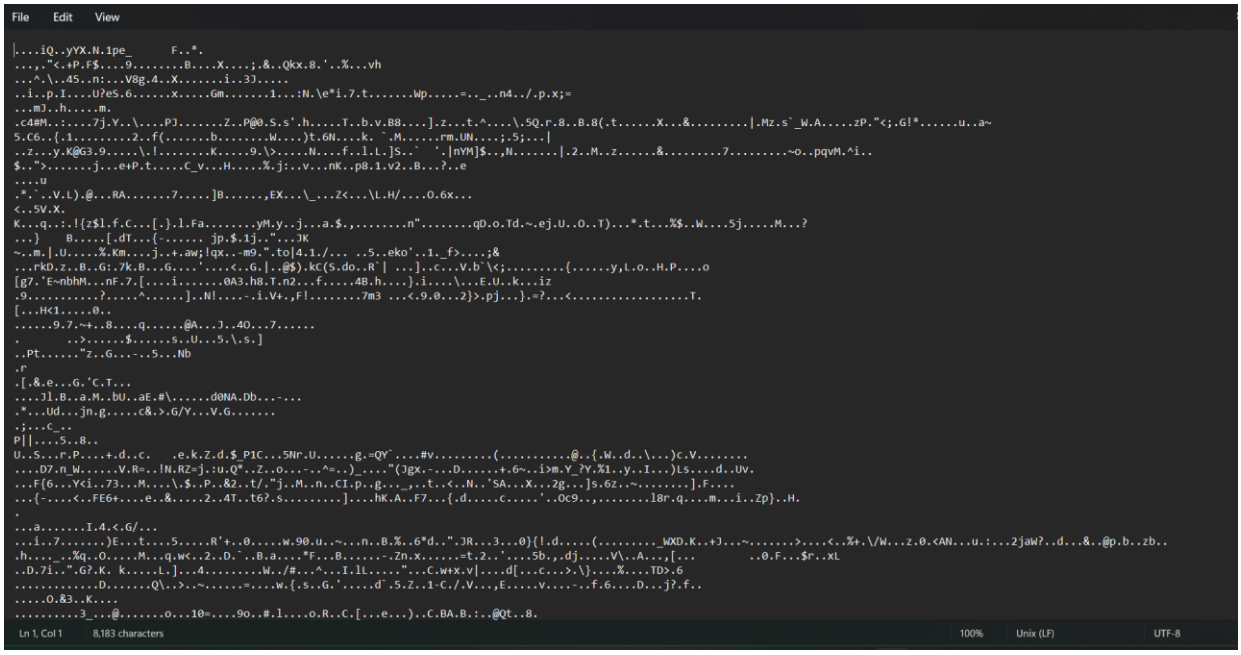

Fig. 8. Exported content of TCP Stream

GEO-MAPPING

In Wireshark, the process of geolocating IP addresses to physical locations is typically achieved through the use of GeoIP databases. These databases map IP addresses to geographical locations such as countries, cities, or coordinates. Wireshark itself does not provide native geolocation features; however, third-party plugins or external tools can be integrated to accomplish this.

To enable geolocation in Wireshark, you would typically need to install a GeoIP database and configure Wireshark to utilize it. One popular GeoIP database provider is MaxMind, which offers both free and paid versions of their GeoIP database. After obtaining a GeoIP database, you would need to specify its location in Wireshark's preferences.

Once configured, Wireshark can display geographical information alongside captured network traffic. This information is typically visible in the packet details pane, where you can see the source and destination IP addresses. Wireshark can then look up these IP addresses in the GeoIP database and display the corresponding geographical information.

It's important to note that geolocation accuracy can vary depending on the quality and freshness of the GeoIP database being used. Additionally, IP addresses may not always map directly to specific physical locations, especially in the case of dynamic IP assignments or the use of proxies and VPNs.

Overall, geolocation in Wireshark can be a useful feature for understanding the geographic distribution of network traffic, but it should be used alongside other analysis techniques for a comprehensive understanding of network behavior.

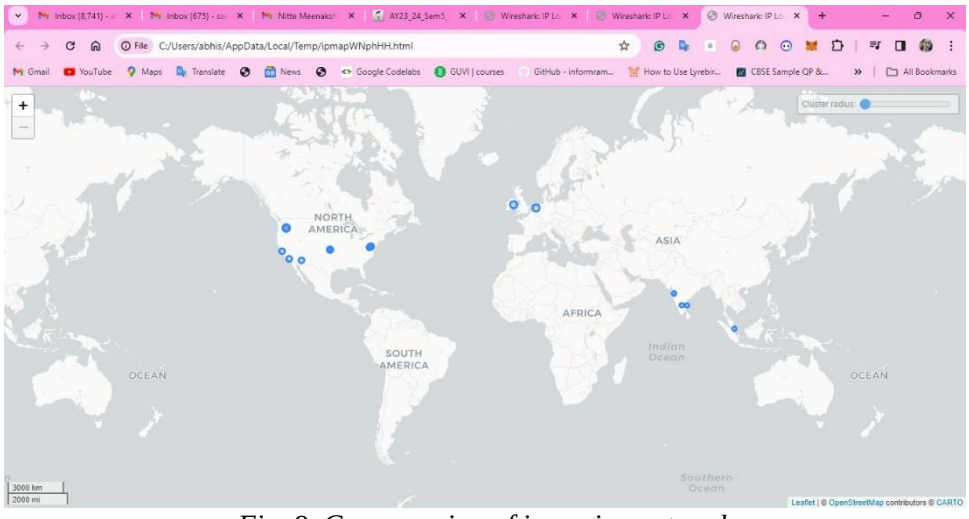

Fig. 9. Geo-mapping of incoming network

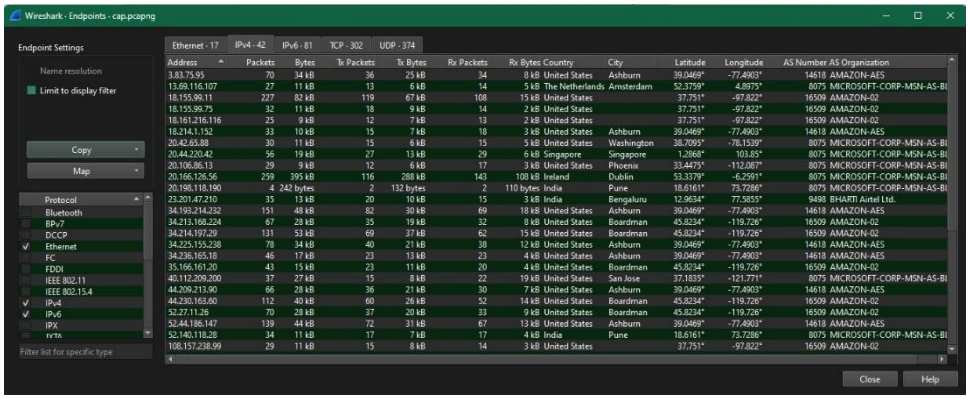

Fig. 10. End-point communication between 2 IP addresses

MALWARE DETECTION

Malware detection in Wireshark involves analyzing network traffic for signs of malicious activity or indicators of compromise (IOCs). While Wireshark is not specifically designed for malware detection, it can be used as part of a broader security strategy to identify suspicious behavior and potential threats. Here's how to perform malware detection using Wireshark:

1. Capture Network Traffic:

Start by capturing network traffic using Wireshark. You can capture traffic from a live network interface or open a previously captured packet capture (PCAP) file.

2. Identify Suspicious Traffic:

Look for anomalies or suspicious patterns in the captured traffic that may indicate malware activity. Some common indicators to watch for include:

- **Unusual outbound connections:** Look for connections to known malicious IP addresses or domains.
- **Unexpected protocols:** Identify protocols commonly associated with malware, such as IRC, peer-to-peer, or suspicious HTTP traffic.
- **Large data transfers:** Malware may exfiltrate data in large volumes, so look for unusually large packets or high data transfer rates.
- **Beaconing behavior:** Malware often exhibits periodic communication patterns known as beaconing. Look for regular, repetitive traffic patterns.
- **Suspicious DNS queries:** Analyze DNS traffic for requests to suspicious or randomly generated domain names.

3. Inspect Packet Contents:

Examine the contents of suspicious packets for signs of malicious activity. This may include:

- **Payload analysis:** Look for unusual or encoded payloads within packets that could indicate malware communication or data exfiltration.
- **Command and control (C2) communication:** Identify communication patterns that resemble C2 traffic, such as command requests or data exfiltration commands.

- **Malicious payloads:** Look for known malware signatures or file hashes within packet payloads.

4. Utilize Wireshark Filters:

Wireshark offers powerful filtering capabilities to focus on specific types of traffic or packets of interest. Use display filters to narrow down the captured traffic based on criteria such as IP addresses, ports, protocols, or packet contents. This can help isolate suspicious traffic for further analysis.

6. **GeoIP:**

GeoIP is a technology that maps IP addresses to physical locations, providing information about the country, city, or region associated with each IP address. It is used for purposes such as website localization, targeted advertising, security, and network analysis.

7. Correlate with Endpoint Data:

If possible, correlate the findings from Wireshark with endpoint data from host-based security tools. Analyzing both network and endpoint data can provide a more comprehensive view of potential malware activity and help confirm the presence of an infection.

By following these steps and leveraging Wireshark's capabilities for packet analysis and traffic inspection, we can enhance your ability to detect malware and malicious activity within network traffic. Additionally, integrating Wireshark with other security tools and practices, such as intrusion detection systems (IDS), endpoint security solutions, and threat intelligence platforms, can further strengthen your malware detection capabilities.

The primary focus of the analysis involves HTTP traffic, revealing a concerning pattern of file downloads by the system DESKTOP-U54AJ8K from foodsgoodforliver.com. After the discovery of potential malware in these downloaded files, the investigation intensifies with the submission of the suspect files to VirusTotal for comprehensive malware scanning and analysis.

The VirusTotal results provide critical insights into the nature of the detected malware, including information about its characteristics, associated threat indicators, and prevalence

across various antivirus engines. This collaborative approach, combining Wireshark traffic analysis with VirusTotal's malware scanning capabilities, offers a holistic understanding of the security risks posed by the network traffic associated with foodsgoodforliver.com.

The case study not only emphasizes the importance of integrating multiple tools for a thorough threat assessment but also provides a practical example of how collaboration between network traffic analysis and malware detection tools can enhance cybersecurity efforts. The insights gained from this study empower cybersecurity professionals to make informed decisions in mitigating potential risks and securing network integrity against evolving threats.

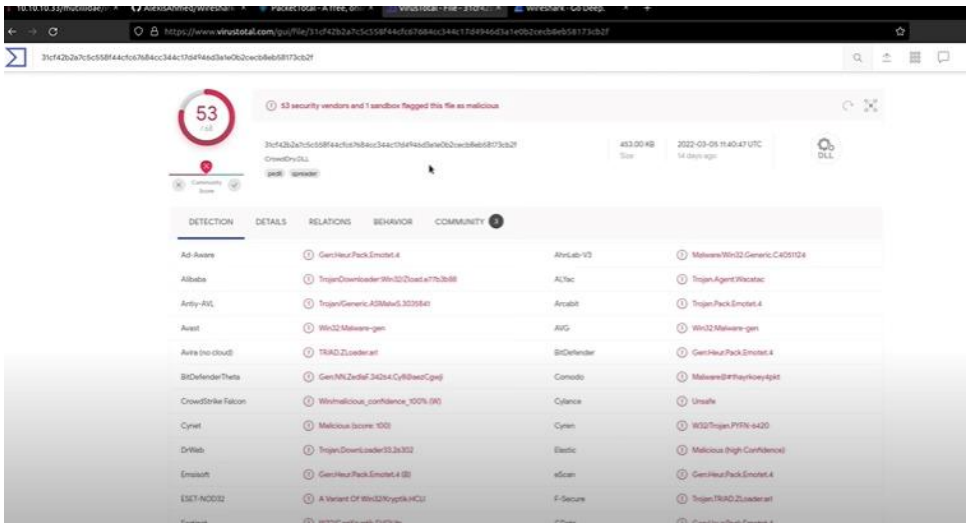

Fig. 11. Malware detected

WEB SERVER CREATION

A web server is a software application or a hardware device that serves content, typically web pages, to clients over the internet or a local network. It utilizes the Hypertext Transfer Protocol (HTTP) to communicate with clients, such as web browsers, and fulfill their requests for web resources.

Steps:

1. **Create a Resource Group (optional):** Organize your Azure resources by creating a new resource group or selecting an existing one.

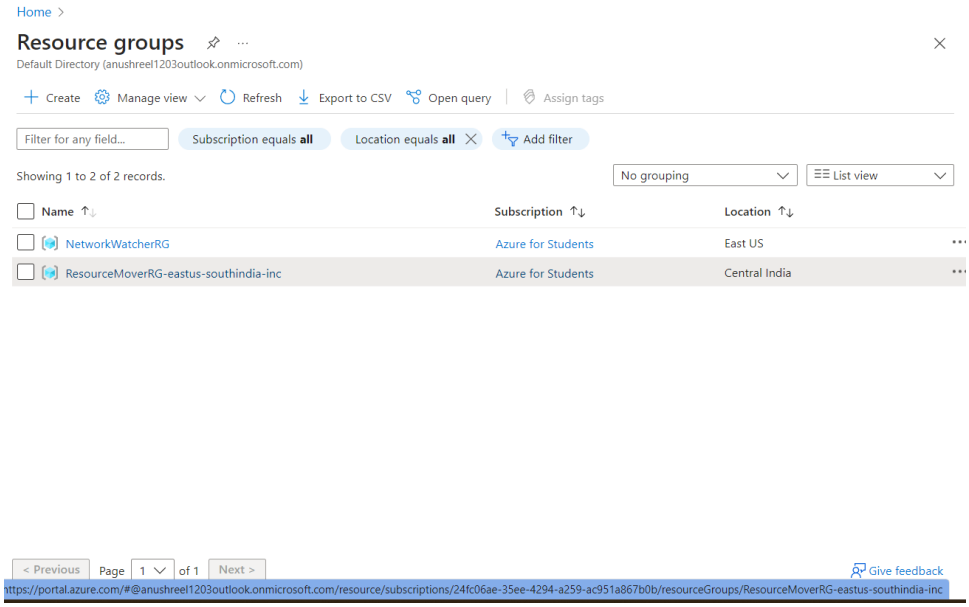

Fig. 12. Creation of resource group in Azure.

2. **Navigate to Create a Virtual Machine:** Click on "Create a resource" in the Azure Portal, search for "Virtual Machine", and select "Virtual Machine" from the search results.
3. **Configure Basics:** Enter basic details such as name, region, availability options, and resource group for your virtual machine.
4. **Choose Virtual Machine Size:** Select an appropriate virtual machine size based on your requirements, considering factors like CPU, memory, and storage.
5. **Configure Settings:** Set up networking configurations such as virtual network, subnet, public IP address, and network security group. Under Authentication type, choose "Password" instead of "SSH public key".
6. **Configure Disks:** Choose the operating system disk type and size. Optionally, add data disks if additional storage is needed.

7. **Configure Username and Password:** Provide a username and password that you will use to access the virtual machine. Ensure the password meets Azure's complexity requirements.
8. **Review + Create:** Review all configurations to ensure they meet your requirements, then click on "Create" to provision the virtual machine.

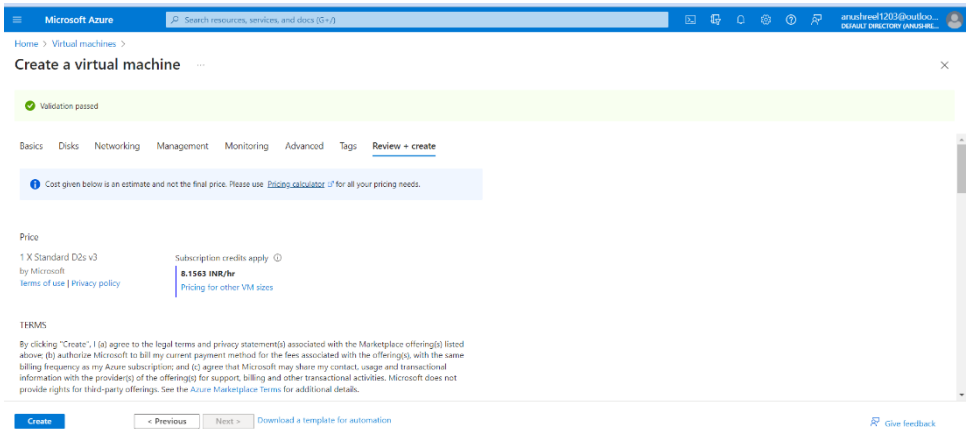

Fig. 13. Creation of virtual machine in Azure.

9. **Access the Virtual Machine via PuTTY:** Once the virtual machine is provisioned, obtain its public IP address from the Azure Portal.
10. **Download PuTTY:** If you haven't already, download PuTTY from the official website: https://www.putty.org/.
11. **Connect to Virtual Machine via PuTTY:** Open PuTTY and enter the VM's public IP address in the "Host Name (or IP address)" field. Make sure the "Connection type" is set to "SSH". Enter the username and password you specified during VM creation. Then, click "Open" to establish the SSH connection.

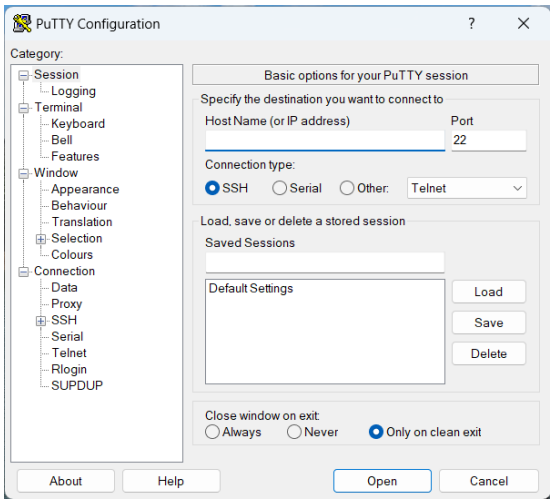

Fig. 14. Connectio to virtual machine through Putty

12. **Install Web Server Software:** Once connected to the virtual machine via PuTTY, install your preferred web server software such as Apache, Nginx, or IIS, depending on your requirements.

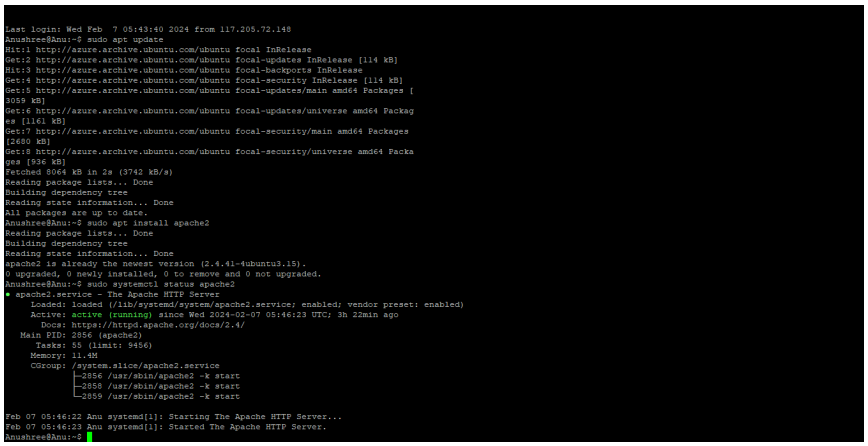

Fig. 15. Installing Apache web server software

13. **Navigate to Desired Directory:** Once connected, navigate to the directory where you want to create the file. You can use the cd command to change directories. The command ’cd /var/ww/html/’ is used.
14. **Create and Edit File:** Add the code in the file that configures the webpage.

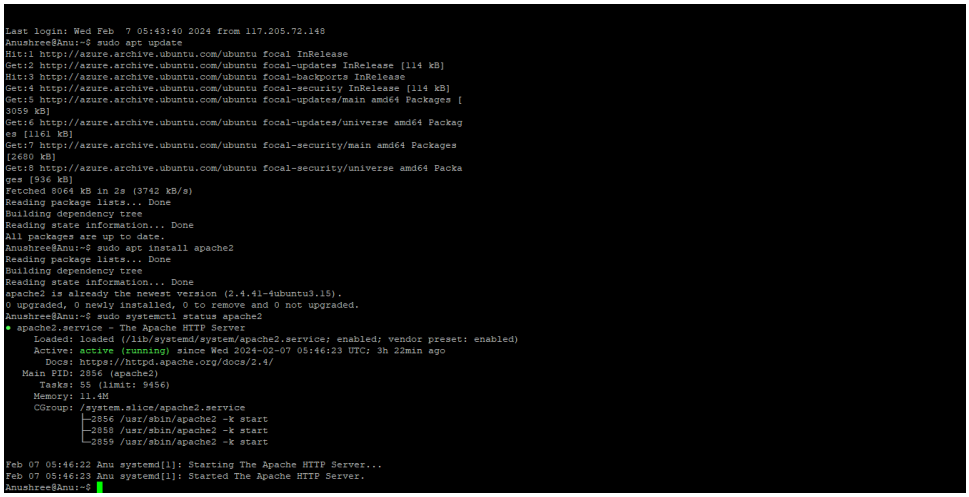

Fig. 16. Creating basic html file

15. **Deploy Your Website:** Upload your website files to the virtual machine using SCP (Secure Copy Protocol) or any other preferred method. Configure the web server software to serve these files.

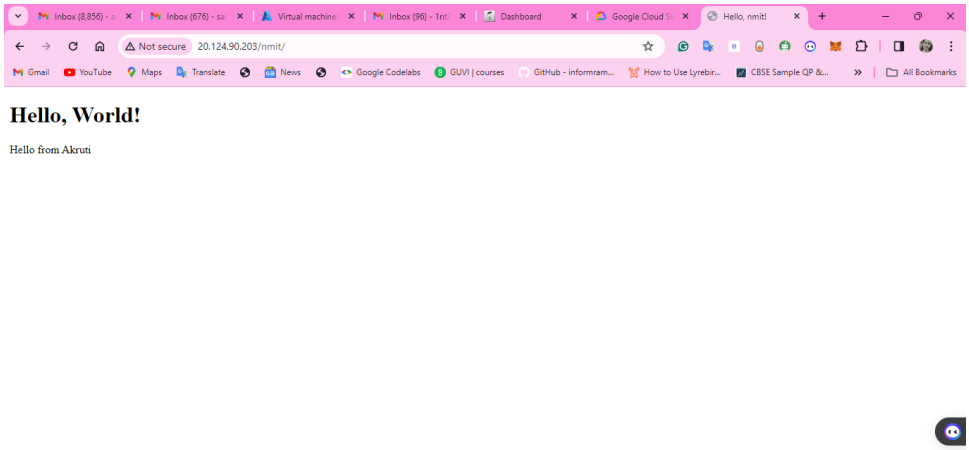

Fig. 17. Web page

16. **Configure Firewall Rules (optional):** Ensure that the necessary firewall rules are configured in Azure to allow inbound traffic to your web server on the appropriate ports (e.g., port 80 for HTTP, port 443 for HTTPS).
17. **Monitor and Manage:** Regularly monitor the performance of your web server using Azure monitoring tools and manage it as needed.

Following these steps will enable you to create and manage a web server on Azure and access it via PuTTY.

CONCLUSION

In conclusion, Wireshark stands as an indispensable tool for comprehensively unraveling the complexities of computer networks. This report has delved into the fundamental aspects of Wireshark, emphasizing its role in capturing, analyzing, and interpreting packet-level data. By providing a user-friendly interface, cross-platform compatibility, and a diverse range of functionalities, Wireshark empowers network administrators, security analysts, and IT professionals to gain profound insights into their network infrastructure.

The exploration of Wireshark's features, including packet capture, filtering, protocol analysis, and advanced capabilities such as decrypting encrypted traffic, showcases its versatility in troubleshooting connectivity issues and ensuring the security of data transmissions. Real-world case studies have demonstrated practical applications, illustrating how Wireshark is a vital asset in diagnosing anomalies and optimizing network performance.

As networks continue to evolve, the importance of tools like Wireshark becomes increasingly apparent. Its ability to unveil the intricacies of network communication equips professionals with the means to proactively manage and secure their network environments. In the ever-expanding digital landscape, Wireshark remains an invaluable ally, fostering a deeper understanding of network dynamics and facilitating proactive measures for enhanced performance, security, and reliability.

BIBLIOGRAPHY

1. https://www.wireshark.org/
2. https://www.comptia.org/content/articles/what-is-wireshark-and-how-to-use-it
3. https://www.techtarget.com/whatis/definition/Wireshark
4. https://www.wireshark.org/faq.html
5. Wireshark User's Guide Version 4.3.0 Richard Sharpe, Ed Warnicke, Ulf Lamping