Real Time Network Traffic Analysis and Visualization Using Wireshark and Google Maps

Swara S Gingade
Computer Science and Engineering
(Data Science)
RV College of Engineering®
Bengaluru, India
swarasgingade.cd22@rvce.edu.in

Nagashree B
Department of Biotechnology
RV College of Engineering®
Bengaluru, India
nagashreeb.bt22@rvce.edu.in

Rishika Mohan V
Department of Biotechnology
RV College of Engineering®
Bengaluru, India
rishikamohanv.bt22@rvce.edu.in

Mohana
Computer Science and Engineering
(Cyber Security)
RV College of Engineering®
Bengaluru, India
mohana@rvce.edu.in

***Abstract*— This paper presents Network Tracker, an innovative solution designed to address the challenge of gaining live insights into network events and better mapping their geographical coverage. In an era where technological advances have raised security and defense systems to new heights, the proliferation of hackers and fraudsters poses a growing threat. Network Tracker uses Wireshark's robust packet analysis capabilities and integrates with Google Maps-based location visualization. The main objective is to empower users, both businesses and ordinary individuals, to manage their networks in a transparent and proactive manner, highlighting the growing importance of networks, but also increasing the risks associated with them. Further highlights the urgent need for traffic internalization of detailed analytics to gain data stream insights, detect traffic risk and enhance overall network performance increased capabilities Google Maps combined with integration providing detailed, packet-level insights into data streams at a scalable level. Through websites flagged as potentially malicious activity to monitor and trace host connections to these servers, this solution provides users with an effective tool to detect and control address security concerns.**

Keywords—Network Tracker, Wireshark, Geographic Visualization, Network Traffic Analysis, IP Geolocation, Geospatial Data Visualization, Packet Sniffing, Google Maps Integration.

I. INTRODUCTION

Professionals and common users miss out on ways to gain live insight into the events occurring within their network because they don't possess the means for seeing how the events relate to geographic locations. Safety and security procedures had come much ahead of what we used to have years ago, similarly with advancement in technology has grown hackers and scammers. The proposed work aims at handling these challenges through building Network Tracker which leverages the capabilities of Wireshark (a potent packet analysis toolkit) and merges its location-based visualizations through integration with Google-Maps. The objective is to bring forth the real use cases of technology, and help individuals take charge of their networks for actual change, in ways transparent and controllable. Nowadays, with the connected world we live in, networks are important to communicate and share data. Since the emergence of the Internet, a large increase in data exchange is observed, which grows exponentially as time goes on and is roughly equivalent to 328.77 exabytes of data being produced globally every day in different forms like texts, files, media and numerous others. The bigger the growth of the network, the higher the risk factor. There's a large amount of malicious content and activities that take place on the Internet, including many attempts to use someone's data and violate their privacy across the Internet ecosystem. Traffic analysis is a vital activity to attain insight on the flow of data, detect risks and problems related to traffic and performance. The goal is to monitor websites which are flagged for suspicious activity and check the connection of the host to said servers or websites. This paper combines the powerful features offered by Wireshark, a proven network protocol analyzer, and maps the coordinates using Google Maps to provide comprehensive packet level insight into the overall flow of data at scale. This allows the user to gain awareness and understand and rectify security concerns much more effectively.

II. LITERATURE SURVEY

Mohammed AL Fawar *et al* [1] presented the significance of traffic analysis for optimizing performance, identifying network anomalies and securing network, while tools such as Wireshark are referred for traffic analysis, network, penetration testing, etc., It's about appreciating the significance of network analysis and the security problems that are associated with it. Kovstur Maxim *et al* [2] presented a method for analyzing wireless network traffic which was written in python programming language and used panda library. In order to make wireless networks available and fault-tolerant, constant monitoring is needed, in that, the actual target is about making better tracking of anomalous traffic during analysis. Laura Chappell *et al* [3] proposed a distributed network analysis in which traffic is inspected at multiple places on the network. The conventional way of doing this is to use a full-blown network Analyzer, which is

quite expensive. Exported packet record files are imported to real-time network analysis tool TOPAS and analyzed using the open-source network analyzer Wireshark. Banerjee, Usha *et al.* [4] described Wireshark as a sniffing tool in the networks which is shown by an experimental setup where effectiveness of malicious packet detection in any network is shown. Inferences have been derived and the capabilities have been made evident which indicates that it could be a potential subject for future development into a robust intrusion detection system. A Dabir *et al* [5] discusses the bottlenecks of commodity hardware-based packet capturing for local area networks (LANs) without data loss. Tests were run with a wireshark packet sniffer to capture packets directly from disk in Fast Ethernet networking with different setups. These were experiments that created large packets near line rate. Also play around with different kernel-level buffer sizes associated with the packet capturing socket. Vixens Ndatinya Zhifeng Xiao *et al* [6] presented Network Forensics Analysis Using Wireshark. As the variety and volume of networked computing systems have been increasing, network security has become an increasingly crucial field due to the increasing number of attacks. Now the network administrators have to be able to monitor and analyze the network traffic to identify the event and the network administrator must be able to respond immediately to any event. Piyush Goyal *et al* [7] Started a Comparison study of two of the best packets sniffing tools —Tcpdump and Wireshark. The increase in the sphere of the Internet has also broadened the spectrum of Networking, data transfer, and data security. They have become the preferred tools of the hackers to be exploited to scan for specific networks and eavesdrop on unencrypted information. These tools allow White Hat hackers to thwart the criminals by filtering out malicious packets and their source. Samer Hamdani *et al* [8] reported research on a COAP vs. MQTT comparison study. IOT technology contains continuous data emitters or sensor data, predominantly transmitted by internet connection, generating homogeneous and massive data transmission. A low-cost method for distributed network analysis was proposed by Gerhard Munz *et al* [9]. In contrast, instead of having very expensive network analyzers at every observation point, the authors recommend using packet data exports from network devices supporting PSAMP and Flexible NetFlow. The idea is to collect packet dumps from these devices and export them to TOPAS, an open-source real-time network analysis framework, which then uses the network analyzer Wireshark to do in-depth analysis. With a Monitor Manager in place, only the necessary data for desired analysis purpose is exported. Consequently, it provides a practical approach that makes use of existing infrastructure and open-source tools to provide scalable and cost-effective distributed network analysis, while highlighting its merits and drawbacks. Laura Chappell [10] founder of Wireshark University has put together a detailed guide to mastering the world's most famous network analyzer tool, Wireshark. This new edition of the best-selling book presents, in step-by-step, easy-to-follow instructions, essential functions and features of Wireshark, including how to filter, create custom columns, identify delayed sources, capture traffic in unattended mode, filter by keyword 46 Hands-on labs make it perfect for both Beginners and Experienced Wireshark users, it will provide you with a lot of insights and experience of network analysis. Sudha Arvind *et al.* [11] presented a novel traffic virtualization approach using Wireshark and Google Maps. This research aims to enhance the abilities of security analysts to identify different types of network traffic, discover anomalous traffic patterns, and identify the culprit network nodes. The researchers used Wireshark's packet sniffing and capture features to build a network tracker which captures data packets, analyzes them, and translates the originating and terminated IP address to latitude and longitude values. This unique approach allows you to see network traffic on Google Maps, allowing network activity to be tracked and understood. Pallavi Asrodia *et al* [12] described the importance of network traffic analysis in the context of the increasing growth and complexity of computer networks. They laid emphasis on packet sniffers, so that they could efficiently manage, operate and monitor these networks and maintain their smoothness, ensuring their economic efficiency. Packet sniffing is a fundamental tool in network monitoring, troubleshooting, and traffic logging, for wired as well as wireless networks. It also goes into the basics about packet sniffers and how they work to analyze network traffic. S Sandhya *et al* [13] emphasizes the importance of penetration testing for discovering weakness in data sharing systems on the web. It specifically identifies Wireshark as a strong tool for finding security holes, especially in user authentication. The paper outlines the benefits of this approach and highlights that it follows the standards. It covers different penetration testing tools and includes an example of using Wireshark as a fundamental penetration testing tool to combat security threats in networks that can be vulnerable to attacks by adversaries. In another work, Resul Das *et al.* [14] aimed to dispel the myth of online anonymity by stressing the need to secure personal details on dangerous networks such as the internet. In a study, it was possible to capture and analyze Wireshark network packets and reveal how personal and location information can be partially obtained from IP cameras to Wireshark. This research highlights the risks involved in transferring data over an unreliable network as it can be used for crime.

III. DESIGN AND IMPLEMENTATION

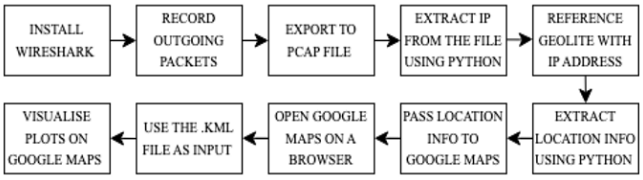

Fig.1. Block diagram and implementation methodology of network traffic analysis

Fig.1. shows the block diagram and implementation methodology of network traffic analysis methodology. Wireshark is a tool that captures packets (A packet contains essential communication information such as IP, target address, and security details) and is used in communication from a network and provides various visualization tools. This system uses Wireshark to capture the packets from the network. These packets are then captured and analyzed for the valuable network data. The packets captured are stored in a PCAP file from where the IP address is retrieved. Python is popularly used for analysis of network traffic, due to the flexibility and availability of a multitude of libraries and tools. Packet Capture analysis can be performed with libraries such

as ‘dpkt’ and capture or dissect networks to read, analyze or tamper data, allowing us to read the packet data for packet sniffing and analysis. A python script is used to perform network scan and enumeration along with pygeoip helping to find hosts, open ports, and gain information about the network service. It is used to show topology and traffic flow. Python scripts generate reports and alerts based on the network traffic analysis results, making it easier for networks to ping and detect malicious activity. This will be used to talk to the API using python scripts and it will have a backend of the development. In a script they use a GeoLiteCity database, which includes the IP addresses and its corresponding coordinates. Google Maps’ visualization capabilities of geospatial data, when combined with Wireshark’s detailed packet-level analysis, provide a full picture of the network traffic. A .kml file is created to store IP addresses as geographical coordinates, and geographical coordinates can be viewed as on geo-location services like Google Earth. The proposed application uses Google Maps to provide mapping functionality that offers geospatial visualization capabilities. It enables applications to show geographic data, including latitude and longitude coordinates. To plot network traffic on a map, this system has to translate from IP addresses (source and destination) to latitude and longitude. This translation can be accomplished through a number of means, including IP geolocation services or databases that map IPs to physical addresses. The system then applies markers or overlays to a Google Map when the .kml file is uploaded to provide the translated latitude and longitude values. Network traffic source and destination IPs are encoded in these markers. Users can then look at these markers on the map to see the flow of network traffic and also change the color of the path, zoom in to track the location of the server etc.

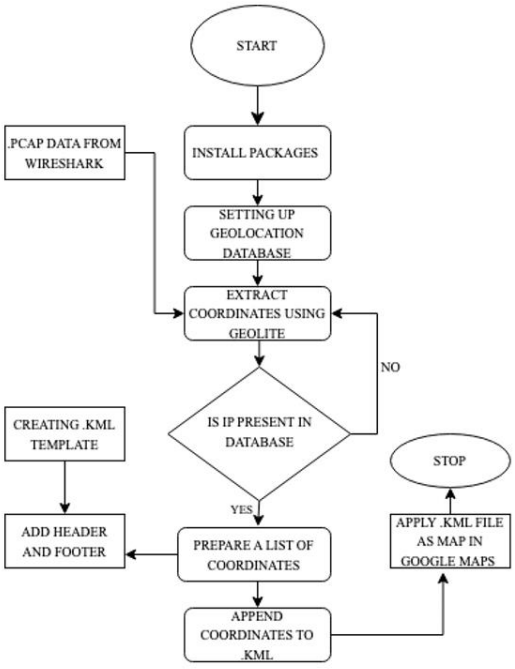

Fig.2. Detailed flowchart of Network traffic analysis

Used wireshark, google maps, python and GeoliteCity database, and then we can combine all of these tools to create a network tracker. In order to track IP addresses on a map, we are using a packet tracer in wireshark to capture data packets. This will allow users to display network traffic on a Google map and view it. Wireshark is set up to monitor the local Wi-Fi network connected to the device and sniff packets to capture a .pcap file on the device. A python script is created to read the .pcap file and generate KML (Keyhole Markup Language) for displaying the geographical path between two IP addresses of network traffic. The python module ‘dpkt’ allows users to parse packets with simple definitions of TCP/IP connections, thereby supporting reading and writing files to a PCAP file used for storing network traffic captures. It also provides the tools to parse network protocols within packets like identifying HTTP requests or DNS queries in the captured traffic. Socket library is a low-level network communication functional library which allows the user to create network sockets, such as TCP/UDP, establish network connections and perform various network operations such as conversion of binary and human-readable representations of IP addresses, and allow simple programs to send and receive data via network connections. The ‘pygeoip’ package is used to mapping IP addresses to geographical locations such as countries, cities, and coordinates (latitude and longitude) via querying a GeoIP database containing geocoded data (where IP addresses map to city, state, country).

Fig.3. shows a detailed flowchart of network traffic analysis design and the implementation from coding a python script to create a .kml file to using it as an input in google maps for visualization purposes. To load the GeoIP database file 'GeoLiteCity.dat', we use the ‘pygeoip.GeoIP’ constructor and create a variable ‘gi’, which contains the resulting object that can be used to cross verify the IP addresses. This database maps IP addresses to physical locations. An ‘retKML’ function is created that accepts a pair of IP addresses (‘dstip’ and ‘srcip’) as parameters. ‘dstip’ is the IP address to receive packets and ‘srcip’ is the IP source and the endpoint time (packet transmission) of packet packets. It employs geolocation by using the ‘pygeoip’ library to look up the country, city, latitude, longitude, associated with ‘dstip’ and the host IP address for ‘srcip’. The lat/long of source and destination IPs is taken from the GeoIP database. A .kml string is created, with the extracted values as the coordinates for the source and destination IPs that will be used to create a line between them. In this process, an empty string is returned if any exceptions occur. A plotIPs() function is created that takes the .pcap as input. It starts an empty string variable called ‘kmlPs’ which is to store the KML data, and it iterates over each packet in the .pcap file to extract the source and destination IP addresses. For each packet, it makes a call to the ‘retKML’ function, which returns KML data for source and destination IP of the packet. The ‘kmlPs’ string is updated with the addition of KML data. Any packets that fail during processing are caught here and ignored. The ‘main’ function is the primary entry point of the script. It reads a binary .pcap file. It starts with the KML header which carries KML Version Info and a style definition for the lines on the map. It then calls the plotIPs function to process the PCAP file and generate KML data. A KML document (‘kmldoc’) is a concatenation of the KML Header, the generated KML Data, and the KML Footer. The ‘kmldoc’ is written to an output file. Then finally, the script is executed — it executes when the script itself is run directly. In short, it reads network packets in pcap format, extracts source and destination IP addresses

and performs IP geo location on these IP addresses to generate a KML file which can be used to visualize the communication path on a map. This could be valuable when monitoring and analyzing network traffic and when the geographic aspect of network traffic is pertinent. Then, upload the .kml file with the coordinates of the packets captured in it into Google Maps to show the data path of the packets.

IV. RESULTS AND ANALYSIS

This section describes the detailed results and analysis.

Tools and libraries used – Wireshark, Google Maps API, Python. *Dpkt* - used to extract IP addresses from packet. *Socket*- is used to set the protocol used for packet transfer. *Pygeoip* - is used to convert IP to coordinates.

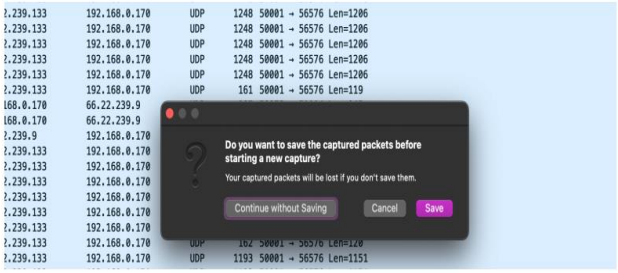

Fig.3. Packets capture using Wireshark.

```
def retKML(dstip, srcip):
    dst = gi.record_by_name(dstip)
    src = gi.record_by_name('49.207.244.191')
    try:
        dstlongitude = dst['longitude']
        dstlatitude = dst['latitude']
        srclongitude = src['longitude']
        srclatitude = src['latitude']
        kml = (
            '<Placemark>\n'
            '<name>%s</name>\n'
            '<extrude>1</extrude>\n'
            '<tessellate>1</tessellate>\n'
            '<styleUrl>#transBluePoly</styleUrl>\n'
            '<LineString>\n'
            '<coordinates>%6f,%6f\n%6f,%6f</coordinates>\n'
            '</LineString>\n'
            '</Placemark>\n'
        ) % (dstip, dstlongitude, dstlatitude, srclongitude, srclatitude)
        return kml
    except:
        return ''
```

Fig.4. Extraction of longitudes and latitudes using .dat file

```
def main():
    with open('wire.pcap', 'rb') as f:
        pcap = dpkt.pcap.Reader(f)
        kmlheader = '<?xml version="1.0" encoding="UTF-8"?> \n<kml xmlns="http://www.opengis.net/kml/2.2">\n<Document>\n'\
            '<Style id="transBluePoly">' \
                    '<lineStyle>' \
                    'width>1.5</width>' \
                    '<color>501400E6</color>' \
                    '</lineStyle>' \
                    '</Style>'
        kmlfooter = '</Document>\n</kml>\n'
        kmldoc = kmlheader+plotIPs(pcap)+kmlfooter
        with open('cord.kml', 'w') as k:
            k.write(kmldoc)
```

Fig.5. Creation of header file to highlight the path.

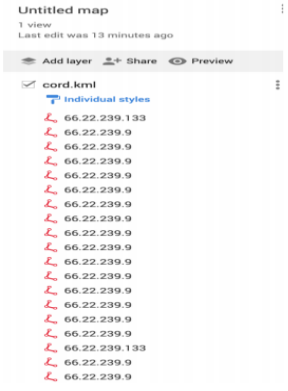

Fig.6. Data path of packets in coordinates.

Fig.7. Visual representation of data path of packets.

Figure 3 to 7 shows the implementation steps of real time network traffic analysis and visualization. Adding header and footer information to the KML file will allow the users to visualize the data on Google Maps. For the simulation, Wireshark was allowed to run for 10 minutes to obtain a sample packet list. The simulations show the servers through which the data is being communicated. All the main servers from where the global communication is made through the South Indian region are quickly communicated to the major servers such as the United States, Alaska, Canada etc. From this we can infer that the data path that follows is secure and is not being rerouted to any undisclosed locations or to places which are tagged under suspicion. Note that this simulation does not show secured packages such as HTTPS, SSL and so on.

V. REAL TIME IMPLEMENTATION CHALLENGES

Network analysis framework implementation ran into a number of severe difficulties when it came to tracking encrypted packets. By design, encrypted packets use a 'secure' protocol that enables untraceable transfers, making them resistant to common monitoring strategies used by network analysis tools like Wireshark. We were severely constrained in our capacity to thoroughly inspect and analyze network traffic because of this tremendous barrier. One of the primary complications arising from encrypted packets is the inherent inability of Wireshark to effectively monitor and decipher their contents. This deficiency is primarily due to the cryptographic measures embedded within encrypted packets, which protect the confidentiality and integrity of the transmitted data. Consequently, Wireshark, and similar tools, are rendered ineffectual in decrypting and examining the payload of these packets, which hampers our ability to gain insights into the nature of the traffic and its potential security implications. Furthermore, efforts to monitor specific websites and their associated packet exchanges faced additional challenges rooted in legal and ethical considerations. Certain websites and their corresponding network activities may be subject to privacy regulations, data protection laws, or ethical guidelines that limit the extent to which they can be scrutinized. In compliance with these legal and ethical constraints, we refrained from monitoring certain websites and their packet exchanges. As a result, this deliberate exclusion introduced what we refer to as 'blind spots' in our network analysis. These 'blind spots' represent areas within our network analysis where we were unable to collect data or gain insights due to the legal and ethical limitations imposed on our research. These constraints underscore the importance of striking a delicate balance

between conducting rigorous network analysis and adhering to the legal and ethical boundaries governing the surveillance of network traffic. In summary, the challenges posed by encrypted packets and the need for compliance with legal and ethical standards have introduced notable limitations in network analysis framework. These hurdles are critical to acknowledge when interpreting the findings and implications of research, as they underscore the evolving complexities in the field of network monitoring and analysis, particularly in the context of encryption and privacy regulations.

VI. APPLICATIONS

Network Traffic Monitoring and Analysis: This is a complete network traffic monitoring and analysis tool. It constantly monitors the data packets flowing through a network. This information enables Network administrators to obtain an understanding about traffic characteristics, such as the protocols being employed, bandwidth usage, and where the traffic is coming from and going to. They can configure alerting for suspicious usage patterns so that capacity is utilized effectively, and anomalies are caught early. Security Monitoring: Security professionals derive value from it, as it helps them detect security threats and anomalies within the network. When geospatially visualizing network traffic it is easier to notice patterns that might indicate nefarious behavior. For instance, a sudden increase in incoming traffic from an unexpected country could point to a cyberattack. It helps to detect intrusion and gives a clear view of holes that need to be patched. Troubleshooting: This system is also useful for IT teams that are experiencing network issues and need to identify the source of the problem quickly. The geographical mapping of network traffic reveals where there is congestion, disruption or outages. This visualization helps IT staff to isolate the issues quickly (hardware failures, bottlenecks, misconfigurations) so that these problems can be fixed swiftly, and the downtime is minimized. Geospatial Data Visualization: The geospatial visualization aspect of the system can be used by industries beyond its network-related applications. For example, a logistics company can use it to monitor the real-time whereabouts of delivery vehicles on a map and optimize routes for best performance. Retailers can use sales data to segment by region, inform inventory and marketing decisions. Real-time Tracking: For situations where immediate action is required, real-time tracking is critical. For example, organizers of a live streaming event can use the system to track the dissemination of video data across their network. Anything sudden in performance can be resolved in real-time to give a smooth viewing experience. Historical Data Analysis: Historical data analysis gives organizations an insight into network performance from a historical perspective. They can use it to forecast long-term trends, plan for network expansion or upgrades and effectively allocate resources. For example, a telecommunications company can use historical data to anticipate the need to increase network capacity in areas where data volume is on the rise. Educational Purposes: This system can be used as a useful classroom teaching tool in educational institutions. It's something students can interact with to learn difficult networking concepts like packet routing and network protocols. They can also simulate scenarios and get immediate feedback on network traffic, improving their sense of network behavior. Customized Network Monitoring: In which it is possible for users to customize the monitoring to suit their needs. For instance, an enterprise could monitor the use of a key application to ensure it is performing adequately. Such flexibility enables the system to be configurable for different industries and application scenarios. IP Geolocation: In several contexts it is valuable that the system may map IP addresses to physical locations. E-commerce firms can offer region-specific offers to customers, while security teams can relate IP addresses with geo-locations in order to detect suspicious activities or fraud, to provide some examples. Python Scripting: Python serves as the backend language and allows for extension and integration. Developers can write scripts for automating tasks or for integrating with other software and can also add new features to the system, as desired. As a result, the system itself is capable of adapting towards new network requirements.

VI. CONCLUSION

In this paper real time network traffic analysis is implemented using Wireshark and Google Maps. The users would be able to effectively use Wireshark, a powerful network packet analyzer, to get IP addresses and other information from network traffic. This proficiency was identified as important for network troubleshooting and security analysis. Users were pretending to understand how to convert IP addresses into geographical locations using tools like GeoLiteCity, to plot the physical location of network activity. This could be applicable to tracking and geographic analysis. Understanding the KML files and how to interact with Google Maps, thus enabling participants to see the network communications on a map, in a very clear and intuitive way and by projecting onto a map, users would be able to visualize the route and the physical locations where information passed i.e., It was suggested that it may assist in detecting network problems as well as enhancing data transfer. Finally, users were expected to know how to monitor network traffic to identify suspicious or malicious packets, and it was particularly important for network security and ability to locate, react to threats in real time.

Some of the important issues that are planned to be tackled in the future is the dissection of secure protocols such as HTTPS to uncover malicious, yet encrypted data which can damage single or multiple systems and servers. Further ongoing research has a promising scope in developing a compact and portable software tool through developing and enhancing further understanding of the intricacies of cyber security so as to allow clients and users to understand, monitor and maintain network security and prevent cyber-attacks, ransomware and many such malicious activities.

REFERENCES

[1] Alfawareh Muhamed *et al* "A deeper Look into Network Traffic Analysis Using Wireshark" *Academia.edu*, 2015.

[2] K. Maxim *et al* "Research of wireless network traffic analysis using big data processing technology," *International Congress on Ultra Modern Telecommunications and Control Systems and Workshops* (ICUMT), 2021, pp. 115-121.

[3] Chappell *et al* "Wireshark network analysis: the official Wireshark certified network analyst study guide", 2nd Edition, ISBN: 978-1-893939-94-3,*Chappell University*, 2010.

[4] Usha Banerjee *et al* "Evaluation of the Capabilities of WireShark as a tool for Intrusion Detection" *International Journal of computer applications*, 2010, pp.1-5.

[5] A. Dabir *et al* "Bottleneck Analysis of Traffic Monitoring using Wireshark," *Innovations in Information Technologies (IIT)*, Dubai, United Arab Emirates, 2007, pp. 158-162.

[6] Ndatinya, Vivens *et al.* "Network forensics analysis using Wireshark" *International Journal of Security and Networks*, 2015, pp.91-106.

[7] P. Goyal *et al* "Comparative study of two most popular packet sniffing tools-Tcpdump and Wireshark," *International Conference on Computational Intelligence and Communication Networks (CICN)*, 2017, pp. 77-81.

[8] S. Hamdani *et al* "A Comparative study of COAP and MQTT communication protocols," *International Symposium on Digital Forensics and Security (ISDFS)*, 2019, pp. 1-5.

[9] G. Munz *et al* "Distributed Network Analysis Using TOPAS and Wireshark," *IEEE Network Operations and Management Symposium Workshops*, 2008, pp. 161-164.

[10] Chappell, Laura *et al* "Wireshark 101: Essential skills for network analysis-wireshark solution series" *Laura Chappell University*, 2017.

[11] S. Arvind *et al* "Network Traffic Virtualization Using Wireshark and Google Maps" *International Conference on Distributed Computing and Electrical Circuits and Electronics (ICDCECE)*, 2023, pp. 1-6.

[12] Asrodia, Pallavi *et al* "Network traffic analysis using packet sniffer" *International journal of engineering research and applications*, 2012, pp. 854-856.

[13] S. Sandhya et al "Assessment of website security by penetration testing using Wireshark," *International Conference on Advanced Computing and Communication Systems (ICACCS)*, 2017, pp. 1-4.

[14] R. Das *et al* "Packet tracing and analysis of network cameras with Wireshark" *International Symposium on Digital Forensic and Security (ISDFS)*,2017, pp. 1-6.