**Network Traffic Classification**

**Team members**

Jeevan Rai

Steven Sells

Aqsa Siddiqui

**Video Link**

1. **Background (Aqsa)**

In today's digitally driven landscape, organizations rely heavily on robust and intelligent network infrastructure to support operations, collaboration, and innovation. As businesses expand their digital footprints, the complexity and volume of network traffic increase exponentially, often resulting in performance bottlenecks, bandwidth congestion, and challenges in traffic monitoring. Traditional monitoring solutions are often reactive and insufficient for real-time classification or detection of traffic anomalies. To address these limitations, our team embarked on a project to explore how simulated network environments and machine learning can be integrated to classify and analyze traffic patterns more effectively.

This project centers around the design and deployment of a simulated enterprise network using **Graphical Network Simulator 3 (GNS3)**, mimicking a small manufacturing company with Engineering, Finance, and Sales departments. Each department's devices were configured and connected to a core router and internet access via NAT. Traffic data was generated and captured through **Wireshark**, with simulations reflecting real-world behaviors like web browsing, streaming, DNS queries, and diagnostics.

The collected traffic data was exported and analyzed using **JupyterLab** with **Pandas** and **machine learning models** including Decision Tree, SVM, KNN, Logistic Regression, Random Forest, and Neural Network. This approach allowed for the classification of network traffic types and the evaluation of each model’s predictive accuracy. The goal was not only to identify high-usage activities and potential inefficiencies but also to demonstrate how intelligent traffic classification can support proactive network optimization.

Through this project, we aimed to showcase a scalable and adaptable framework for **data-driven network traffic analysis**, providing network administrators with actionable insights to improve performance, security, and planning.

1. **Process Flow (Jeevan)**

Using various tools (mentioned in section 4) and knowledge of Computer Networking along with Machine Learning, following steps were taken to achieve the project objective.

1. Designed network topology for the small manufacturing company in Graphical Network

Simulator-3 (GNS3) with the following features.

* Engineering Department: 2 PCs (VPCS) and 3 laptops (simulated using VPCS) connected to an ethernet switch (Switch 1), the switch is connected to a c7200 router
* Finance Department: 3 PCs (VPCS) and 1 laptop (simulated using VPCS) connected to an ethernet switch (Switch 2), the switch is connected to a c7200 router
* Sales Department: 3 PCs (VPCS) connected to an ethernet switch (Switch 3), the switch is connected to a c7200 router
* The router is connected to Network Address Translation (NAT)

1. Devices were ran on virtual machine called GNS3 VM using Oracle Virtual Box Manager
2. Devices were assigned with IP addresses as shown in section 4.1
3. Connection between the router and NAT node was set up with IP address as “DHCP” and routed to NAT with necessary configuration. Interfaces at the router from the switches were configured as “nat inside” while the interface from NAT was configured as “nat outside”
4. To verify successful network connections, the router was pinged from each of the connected devices using respective IP addresses. The devices’ IP addresses were also pinged from the router. To check internet connection, NAT was pinged as “8.8.8.8” from the router. Same process repeated from each of the connected devices.
5. To capture network traffic data, the data was captured using a PC on the GNS3 environment. To do so, the connection between the PC and the respective switch has to be selected and start the capture session. This automatically initiated the Wireshark application. The capture session ran for a couple of minutes, and was exported as a “csv” file called “network\_traffic\_data.csv”.
6. The file was read as a pandas dataframe called “data” in the Jupyter lab. Data properties were examined, and some basic data preprocessing steps were performed to build the final dataset called “dataset”
7. The final dataset was split into training and testing datasets based on the proportions of the protocol types in the final dataset.
8. Prepared the dataset for training and testing by defining the training features and predicting target variables.
9. Using the training dataset, all of the following 6 models were trained.

* Decision Tree
* Support Vector Machine (SVM)
* K-Nearest Neighbor (KNN)
* Logistic Regression
* Random Forest
* Neural Network

1. Using the testing dataset, the accuracy of each trained model was analyzed. Additionally, the confusion matrix was used to analyze the performance of the prediction in greater detail.
2. The accuracy of each model was compared using visualization.
3. **Motivation (Aqsa)**

In the digital age, reliable and high-performing network infrastructure is essential to the daily operations and long-term success of modern businesses. With growing reliance on communication platforms, remote access, and cloud-based applications, organizations demand consistent, high-speed connectivity. However, as network architectures become increasingly complex and traffic volumes rise, identifying issues such as bandwidth congestion, performance bottlenecks, or unauthorized access becomes significantly more challenging. Traditional monitoring methods often fall short—offering limited visibility and lacking the ability to classify traffic behavior in real time—leaving administrators reactive rather than proactive.

This project was driven by the need for a smarter, more responsive approach to network traffic analysis. Our objective was to emulate a realistic enterprise network environment using GNS3 and collect traffic data through Wireshark. The captured traffic, generated through a variety of simulated activities, was then analyzed using machine learning models to classify network behavior into categories such as streaming, web browsing, DNS queries, and diagnostics. By combining simulation with AI-powered analysis, we aimed to uncover actionable insights such as usage trends, peak traffic periods, and potential vulnerabilities—details that static monitoring tools typically overlook.

Ultimately, this project aspires to empower network administrators with intelligent tools that offer deeper visibility and control over network operations. Machine learning–based traffic classification enables organizations to identify inefficiencies, make data-driven decisions, and implement optimizations—ranging from bandwidth allocation and load balancing to hardware upgrades. Our motivation stems from the belief that intelligent, adaptive network analysis will be essential for building resilient, scalable, and future-ready systems capable of meeting the evolving demands of a connected world.

1. **Tools Used (Jeevan)**

**4.1 Graphical Network Simulator -3 (GNS3)**

GNS-3 is an open-source network simulation software used to design, build, and test virtual and real network environments. It's widely used by network engineers, students, and IT professionals preparing for certifications like Cisco CCNA, CCNP, and beyond. This was used to create the network topology for a small-sized manufacturing company as shown below.

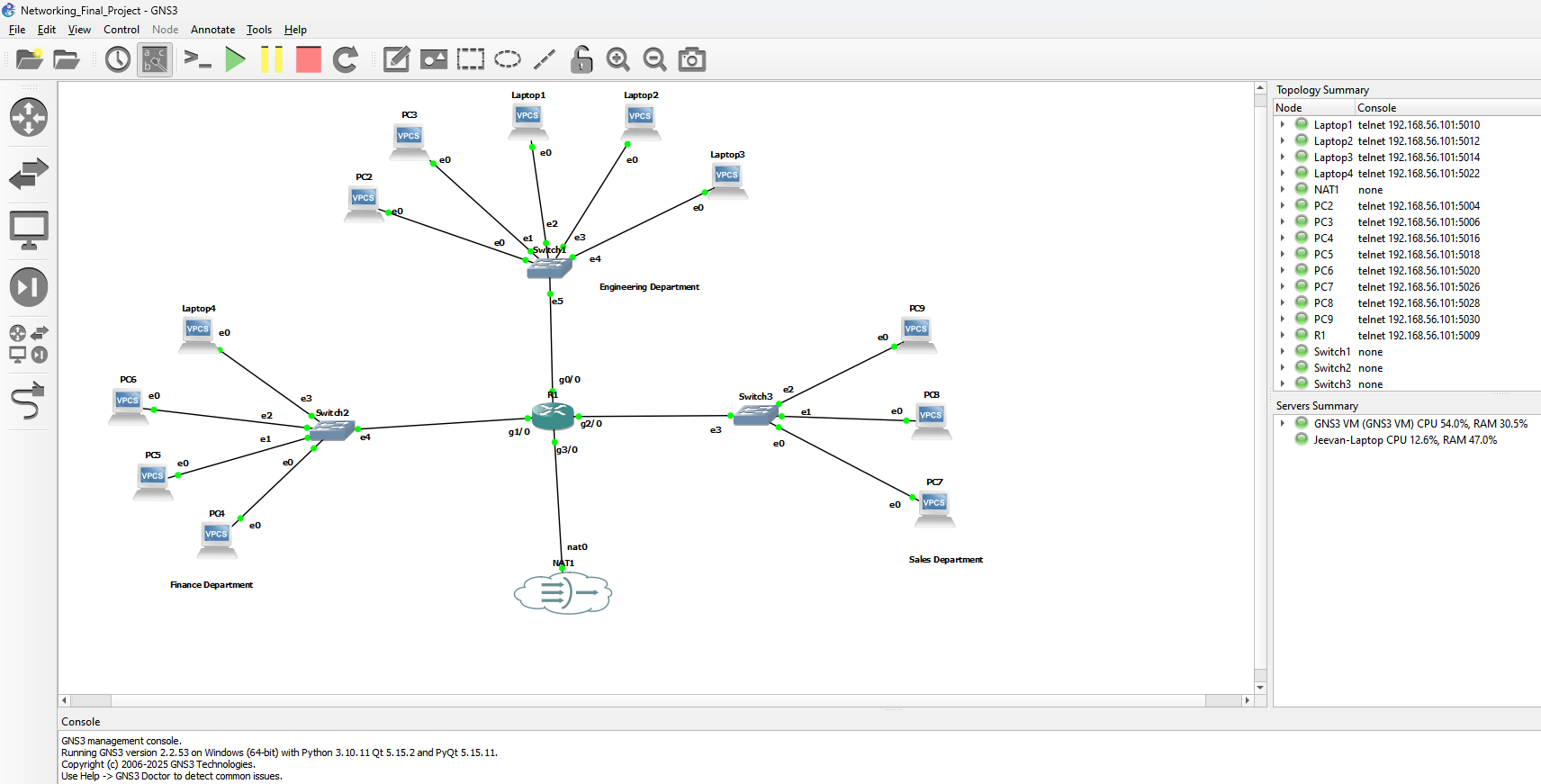


Fig 1. Network topology

The network topology shows all the devices used in the finance, sales, and engineering departments of a small manufacturing company. All of the devices are connected to a switch which is connected to a c7200 router (R1). The router is then connected to a Network Address Translation (NAT) to allow the devices to access the internet through the host machine’s network connection. It bridges the gap between the GNS3 internal network and the external network similar to the internet that we have in our homes. Since GNS3 does not have a device like “laptop” similar to Cisco Packet Tracer, VPCS (virtual PC) was used to simulate a laptop. Following are the IP addresses assigned to the devices.

**Engineering Department**

Laptop 1: 192.168.1.5

Laptop 2: 192.168.1.6

Laptop 3: 192.168.1.7

PC 2: 192.168.1.3

PC 3: 192.168.1.4

Switch to Router: 192.168.1.1

**Finance Department**

Laptop 4: 192.168.2.7

PC 4: 192.168.2.4

PC 5: 192.168.2.5

PC 6: 192.168.2.6

Switch to Router: 192.168.2.1

**Sales Department**

PC 7: 192.168.3.7

PC 8: 192.168.3.8

PC 9: 192.168.3.9

Switch to Router: 192.168.3.1

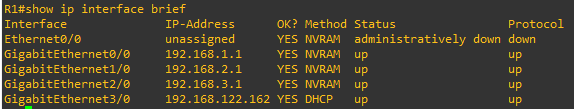


Fig 2. IP interface for the network

Shown above is the IP interface information for the connections between switches and router. The router interface (connected to NAT) is configured for DHCP with IP address of 192.168.122.162 enabling the router and connected devices to reach the internet.

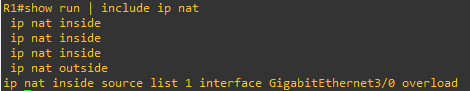


Fig 3. NAT configuration

Shown above is the NAT configuration on the router. The configurations for connection to the router are represented by “ip nat inside”, and for the connection to NAT is represented by “ip nat outside” because the interface facing the internet is outside.

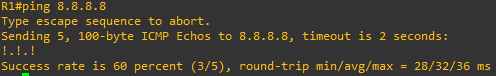


Fig. Ping check to internet

The above figure shows a ping check from router to NAT to check internet connection. The same can be done from any of the devices connected to the router. It shows a successful connection.

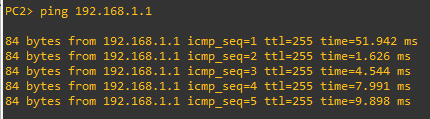


Fig. Ping check to router

Shown above is a ping check from PC2 (engineering department) to the router. It shows a successful connection. The same ping checks were performed from all the connected devices to the router, and found to have successful connections.

**4.2 Oracle Virtual Box Manager**

To run a virtual machine in GNS3 to simulate end devices, GNS VM was set with VirtualBox as virtualization engine. To manage this virtual machine, Oracle virtual box manager was used as shown below.

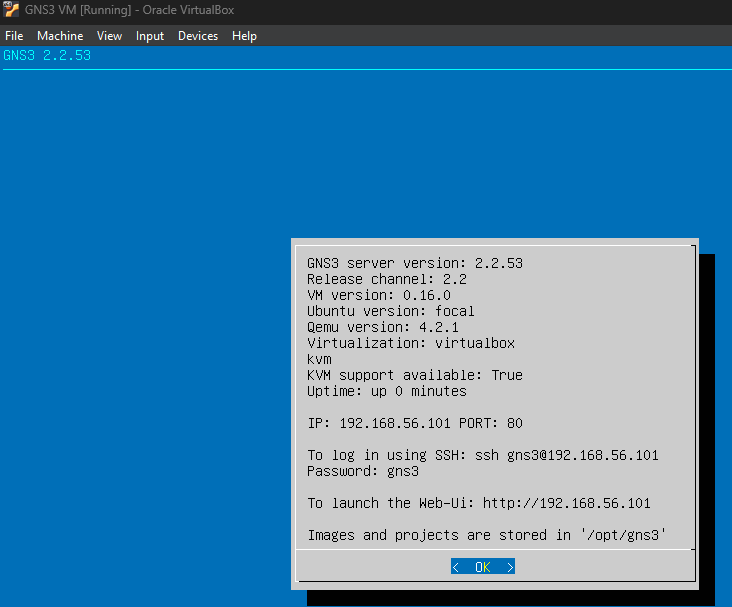
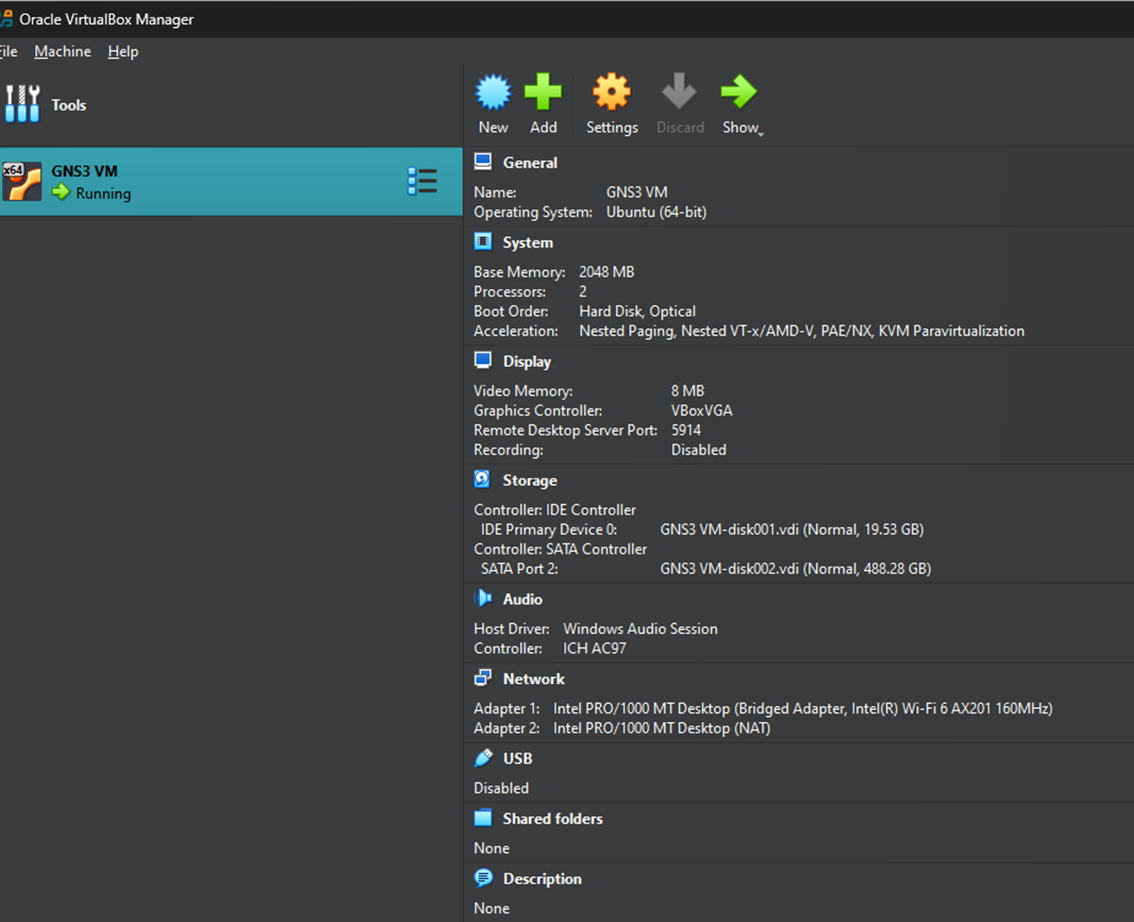


Fig 4. Virtual Box enabled

The image above shows a running virtual machine called GNS3 VM. Image on the left shows “KVM support available: True” and IP address of 192.168.56.101 which is an indication that the virtual machine is running successfully.

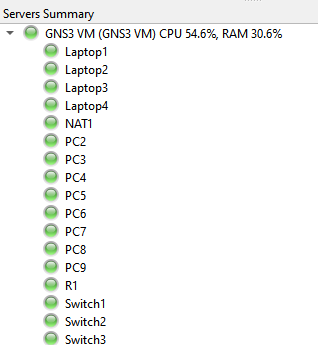
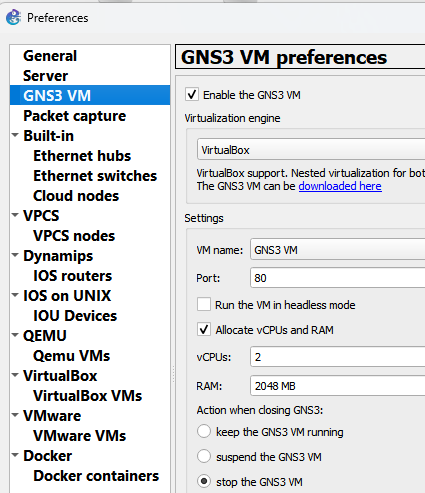
 

Fig 5. End devices running in virtual machine

In addition to previous images, the above image (left) shows that all of the end devices are running successfully through the GNS3 VM server. The image on right shows the preference configured to use the virtual machine as the virtualization engine.

**4.3 Wireshark**

In order to capture network traffic data between a device and NAT (similar to the internet),

Wireshark was activated after initiating a capture from a device as shown below.

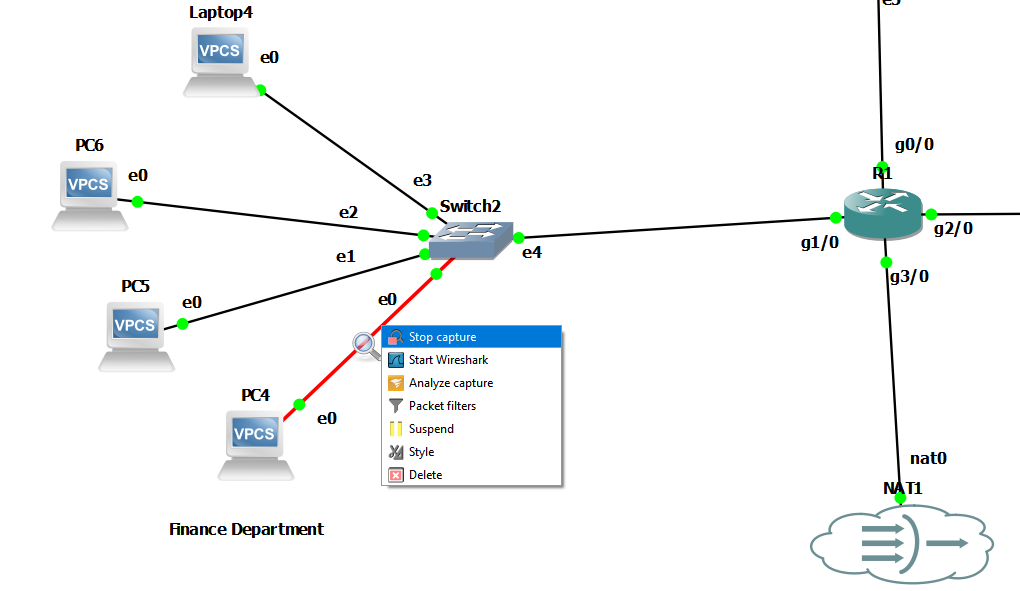


Fig 6. Network data capture.

The image above shows capture of network traffic data from PC4. A snip of the captured network traffic data is shown below.

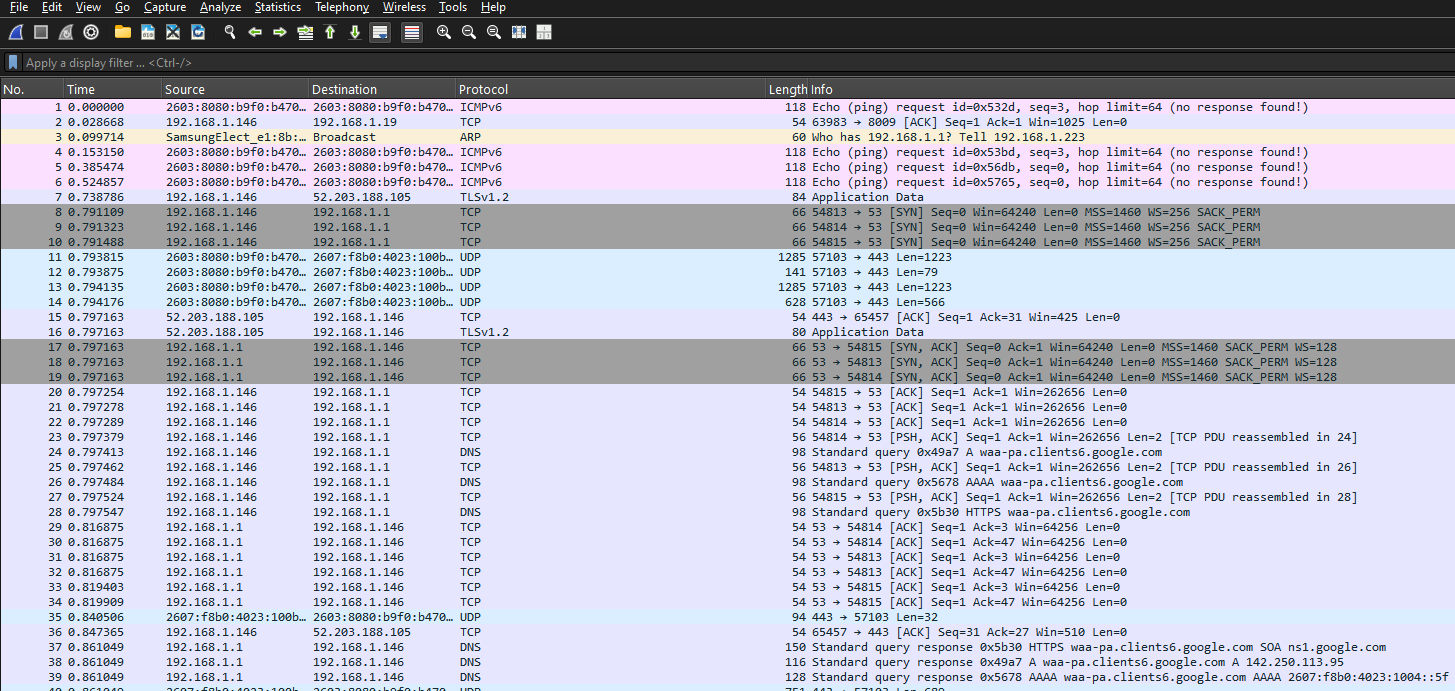


Fig 7. Network traffic data capture using Wireshark

Here we can see Wireshark interface capturing network traffic data. Once the capture is initiated from a device or router, an interface similar to this automatically initiates in Wireshark. Shown below is a quick overview of the different columns seen in the image.

* No. : The sequential packet number in the capture file. Helps in identifying the order of packet arrival.
* Time: The timestamp showing when the packet was captured. It's typically relative to the start of the capture unless set to absolute time.
* Source: The origin IP address (or hostname) of the packet — where the packet came from.
* Destination: The target IP address (or hostname) — where the packet is going.
* Protocol: The protocol used by the packet (e.g., TCP, UDP, HTTP, DNS, ICMP, ARP). This helps us to understand what type of communication is happening.
* Length: The size of the packet in bytes, including headers and payload.
* Info: A summary of the packet's contents. It includes high-level details like request/response, sequence numbers, ports, flags, etc.

**4.4 Jupyterlab (Jeevan)**

After capturing network traffic data (captured as csv file) from Wireshark, further analysis of the captured data was done on Jupyterlab. Following are the primary analysis performed:

* Overview of the different properties of the data
* Distribution of the network data protocols (Streaming, Web Traffic, DNS Traffic, and Network Diagnostic)
* Classification of the network data protocols based on machine learning models (Decision Tree, Support Vector Machine, K-Nearest Neighbors, Logistic Regression, Random Forest, and Neural Network)
* Comparison of the classification performance of each model

1. **Research Questions (Steven)**

* How can network traffic data be captured in a simulated enterprise environment to reflect realistic usage patterns across departments?
* Which preprocessing methods are required to ready raw traffic data for machine learning classification?
* Which machine learning algorithms excel in accurately categorizing various types of network traffic?
* Are the classification results useful for identifying bandwidth-intensive activities and performance bottlenecks in the network?
* In what ways can classified traffic data insights enhance strategies for network optimization and future planning?
* When choosing a model for practical deployment, what are the trade-offs among accuracy, speed, and interpretability?
* Is it possible to use time-series patterns in network traffic to forecast future increases in traffic volume or recognize recurring trends?
* What effect do encrypted traffic flows (such as HTTPS) have on the ability to classify types of traffic using only metadata?

1. **Data Collection (Steven)**

Using Graphical Network Simulator 3 (GNS3), we designed a simulated enterprise network, which initiated the data collection process. This design enabled us to establish a virtual environment that emulated a small manufacturing company comprising three departments: Engineering, Finance, and Sales. Each department comprised several end devices linked via Ethernet switches, which were subsequently connected to a central c7200 router. This router was linked to a Network Address Translation (NAT) device that offered simulated internet access. This topology bears a close resemblance to actual enterprise networks, enabling us to observe realistic communication flows both within and between departments.

After the virtual network became operational, traffic was captured using Wireshark at the link between the router and the NAT device. This guaranteed the recording of both internal communication and internet-bound traffic. Simulated devices performed specific activities — including web browsing, file downloading, DNS lookups, and diagnostic pings — to create a variety of traffic types. The purpose of these actions was to generate unique traffic signatures (e.g., high-volume TCP packets from downloads as opposed to short, periodic DNS queries). Captures initiated from the GNS3 interface automatically opened Wireshark, gathering packets for several minutes in each simulation session.

The data obtained was exported from Wireshark in .csv format and included essential fields like packet number, timestamp, source and destination IP addresses, protocol type, packet length, and info. The structured data acted as the raw input for the machine learning analysis. Once the data was collected, it was imported into JupyterLab as a Pandas DataFrame from a CSV file. From that point, initial preprocessing actions were performed, including the elimination of duplicates, filtering out non-relevant protocols, and developing features for classification. Each row was assigned to one of four traffic categories — Streaming, Web, DNS, or Diagnostic — based on observed behaviors and protocol types through additional labeling. Thanks to this dataset, which was organized with care, we were able to train and assess multiple machine learning models in the subsequent phase of the project.

1. **Data Overview (Steven)**

Several thousand records representing individual packets traversing the simulated enterprise network were contained in the dataset created from the captured network traffic. Each record contained metadata fields like packet number, timestamp, source IP, destination IP, protocol, packet length, and a brief summary of the packet's contents (as displayed in Wireshark’s “Info” column). These characteristics offered a comprehensive array of features for analytical purposes as well as machine learning classification. The traffic mirrored a variety of simulated activities—file downloads, web browsing, DNS lookups, and ICMP pings—emanating from different departments with diverse traffic intensities and behaviors.

Once we imported the data into JupyterLab with Pandas, we started with exploratory data analysis (EDA) to comprehend the distribution of protocol types and pinpoint traffic patterns. Our observations indicated that the dataset was primarily composed of TCP and UDP protocols, with ICMP and DNS constituting a smaller but noteworthy fraction. For instance, the engineering department produced high-volume TCP flows that were consistent and indicative of video streaming or large file transfers, while DNS and ICMP traffic was more sporadic and low-volume, typically linked to name resolution and diagnostic commands. The behavioral distinctions served as a useful basis for traffic labeling and feature selection.

The subsequent step entailed cleaning and transforming the data. To ensure consistency, duplicate packets, broadcast frames, and any malformed entries were eliminated. Key features were identified, including packet size, protocol, source and destination ports (when available), and timing intervals between packets. A labeling scheme was developed manually to categorize traffic into four types: Streaming, Web Browsing, DNS, and Diagnostics. This label was used as the target variable for supervised learning. The cleaned dataset was divided into training and testing sets, employing stratified sampling to preserve balanced class distributions. To validate the variety and spread of traffic types within the dataset, visual aids like bar plots and pie charts were employed. Thanks to this thorough overview and preparation, our models were trained on high-quality, representative data that reflected actual network behavior.

1. **Machine Learning Model (Aqsa)**

To classify the different types of network traffic, we implemented a supervised machine learning approach using six distinct models: **Decision Tree, Support Vector Machine (SVM), K-Nearest Neighbors (KNN), Logistic Regression, Random Forest**, and a **Neural Network**. These models were selected to compare a range of algorithmic approaches—from interpretable linear models to complex, nonlinear classifiers.

The dataset obtained from Wireshark was first cleaned and labeled into four target classes: **Streaming, Web Browsing, DNS, and Diagnostics**. Features such as packet length, protocol type, source and destination IP addresses, and time intervals were used as inputs for model training. The labeled dataset was split into training and testing sets using stratified sampling to preserve class distribution balance.

Each model was trained using the training subset and evaluated on the testing set using metrics such as **accuracy, precision, recall, F1-score**, and **confusion matrices** to assess their performance in detail. Hyperparameters for some models (e.g., number of neighbors in KNN, depth in Decision Tree, etc.) were tuned to improve generalization and avoid overfitting.

Visualization tools were also used to compare performance across models, making it easier to interpret which algorithm was most effective for our classification goals. This multi-model approach allowed us to analyze the trade-offs between accuracy, speed, and interpretability across algorithms.

1. **Results Analysis & Recommendations (Aqsa)**

Upon evaluating all six machine learning models, we found that **Random Forest and Neural Network classifiers delivered the highest accuracy**, with Random Forest slightly outperforming others due to its ability to handle complex feature interactions and its robustness against overfitting. The **Decision Tree and KNN** models also performed well but showed a slight decrease in accuracy when generalizing to unseen data. **Logistic Regression**, while simple and interpretable, struggled with the nonlinear nature of some traffic patterns. **SVM** achieved decent performance but required more computation time and tuning to optimize results.

The confusion matrices revealed that most misclassifications occurred between **Web Browsing and Streaming traffic**, likely due to overlapping packet characteristics such as protocol type and length. DNS and Diagnostic traffic were generally well-separated due to their unique frequency and size patterns.

Based on our findings, we recommend the **Random Forest model** for practical deployment in environments requiring both high accuracy and interpretability. For more resource-constrained environments or where model explainability is critical, **Decision Trees** can still offer a lightweight and understandable alternative.

Moving forward, enhancing the feature set with additional metadata (e.g., flow duration, packet inter-arrival time, or port numbers) could improve classification granularity. Integrating time-series forecasting or anomaly detection techniques could also enable proactive alerting for unusual traffic patterns or security threats. Lastly, testing this framework with **real-time streaming data** could validate its scalability for enterprise-level applications.