Final Year Design Project Report

VPN SpyGlass: VPN traffic analyzer

By

Hassan Abdullah	337275
Sameen Mubashar	346848

Supervisor:
Dr. Mehdi Hussain

Co-Supervisor:
Dr. Arsalan Ahmad

Bachelor of Science in Computer Science (2020-2024)

Department of Computing
School of Electrical Engineering and Computer Science
National University of Sciences & Technology

DECLARATION

We thus certify that the project report we turned into the School of Electrical Engineering and Computer Science (SEECS), titled "**VPN SpyGlass: VPN traffic analyzer**," is a record of our original work completed with supervision from Dr. Mehdi Hussain and co-supervision from Dr. Arsalan Ahmad, and that no material has been copied without proper citation. Additionally, this project work is turned in to partially meet the requirements for a bachelor's in computer science degree.

Team Members

Hassan Abdullah

Sameen Mubashar

Supervisor:

Dr. Mehdi Hussain

Co-Supervisor:

Dr. Arsalan Ahmad

Date:

10th May 2024

DEDICATION

To Allah the Almighty,

To our Parents and Faculty
&
To Everyone who we met along the way

ACKNOWLEDGEMENTS

We would like to express our gratitude to our respected advisor **Dr. Mehdi Hussain** and co-advisor **Dr. Arsalan Ahmad** for their unwavering support, guidance, and motivation throughout our FYP "VPN SpyGlass" in Software Engineering. Their expertise, advice, and dedication helped us achieve our goals, and we could not have completed this project without their constant encouragement and feedback.

We also extend our heartfelt thanks to all the faculty members of SEECS who provided us with their valuable feedback and support throughout our project. We are also grateful to our parents and friends who supported us and provided us with the necessary resources to carry out our research.

Finally, we would like to express our gratitude to all those who directly or indirectly contributed to our project's success. Thank you all for your support and guidance.

Table of Contents

Table of Figures

Table of Tables

Abstract

The "VPN SpyGlass" project is creating a VPN traffic analyser tool that may be used to identify and evaluate VPN activity on networks to improve network security. The programme distinguishes between VPN and ordinary network traffic using deep packet inspection techniques. It also supports real-time monitoring via a dynamic internet dashboard. The project tackles the obstacles that come with using VPNs, such as security threats, privacy concerns, and network performance issues, and offers a comprehensive solution for effective network security management. VPN Spyglass's user-friendly interface and thorough rating criteria enable consumers to make educated judgements about VPN services while successfully protecting their online privacy and security.

The VPN SpyGlass project was inspired by the challenges and worries that consumers have when navigating the VPN business. Its goal is to give users with clear, unbiased information and tools that will help them make educated VPN service selections. The project combines strong network analysis with simple interface design, allowing users to manage VPN usage, improve network security, and increase network visibility. VPN SpyGlass seeks to increase network administrators' capacity to detect illegal VPN connections, categorise VPN traffic, and reduce security risks by utilising cutting-edge technologies and deep packet inspection algorithms. This will eventually contribute to a safer, more secure digital environment.

Chapter 1

BACKGROUND

1.1 Scope

The VPN SpyGlass project aims to create a powerful and user-friendly open-source tool for monitoring and analyzing VPN activities within a network. This digital solution enables network managers to discover, categorize, and gain insights about VPN traffic on their networks. The program will identify between traditional network traffic and VPN traffic using cutting-edge deep packet inspection techniques, as well as categorize the observed VPN traffic based on the protocol or type being used. The goal is to improve administrators' ability to monitor network activity, identify unauthorized VPN connections, and reduce security hazards.

A dynamic online dashboard powered by ReactJS / NextJS, NodeJS, and Python will give users real-time access into network traffic, enabling them to make informed decisions like limiting VPN connections with PfSense Firewall. The VPN SpyGlass project provides an extensive platform for controlling VPN usage and enhancing network security to bridge the gap between sophisticated network analysis and intuitive interface design.

1.2 Motivation

The numerous obstacles and concerns that clients have while attempting to navigate the huge and rather opaque VPN industry fueled the VPN SpyGlass project. The demand for VPN services has grown because of concerns about online privacy, security risks, and geo-blocking, yet users sometimes struggle to determine which solutions are reliable and acceptable. Many of the current platforms and assessments for VPN services lack extensive investigation, are opaque, or are based on biassed opinions. Consequently, consumers may receive incorrect information, make poor decisions, or suffer security threats. The range of VPN service providers, each offering a unique set of features and guarantees, adds to the complexity and ambiguity of picking a VPN.

The VPN SpyGlass project attempts to address these concerns while also providing users with the knowledge and resources they need to make informed VPN service choices. The aim is to offer individuals confidence and equip them with the tools they need to effectively protect their online privacy and security by rigorous research, objective reviews, and User orientated design.

1.3 Significance of Project

The VPN SpyGlass system's goal is to provide advanced capabilities to network administrators for detecting, categorizing, and regulating VPN traffic within their network. The solution attempts to improve network security and visibility while also providing a user-friendly monitoring and management interface. The following are the product's key outcomes:

Benefits:

1. **Enhanced Network Security**: The solution assists managers in identifying unauthorized VPN usage, allowing them to handle any security breaches as soon as possible.
2. **Monitoring User VPN behavior:** Administrators can monitor user VPN behavior to obtain insight into possible hazards and misuse.
3. **Granular Control**: The solution allows you to selectively prohibit VPN usage, giving you more control over network traffic.

The VPN SpyGlass initiative adds to the broader conversation about cybersecurity, online freedom, and digital rights from a social standpoint. Through the initiative, users are equipped with the information and resources necessary to protect their online activities against censorship, monitoring, and data breaches, therefore promoting a more robust and democratic digital environment.

1.4 How Solution is Different?

The VPN SpyGlass solution distinguishes itself from other VPN evaluation platforms with its comprehensive approach, state-of-the-art technology, and user-focused design. First off, unlike a few other sites that only focus on user reviews or performance metrics, the VPN SpyGlass website offers a thorough evaluation system that considers a range of VPN service factors. Jurisdictional analysis, privacy assessments, security audits, user feedback aggregation, and performance testing are all included in this. The website offers users a comprehensive comparison of all VPN providers, assisting them in making decisions based on their unique goals and concerns.

Second, the VPN SpyGlass project uses cutting edge technologies like automation, data analytics, and machine learning to increase the scalability and efficiency of its inspections. The platform uses automated processes for data collection, analysis, and updating to ensure that its material is reliable, accurate, and current. The user experience and accessibility are highly valued on the VPN SpyGlass platform. Its flexible capabilities, feature-rich interfaces, and educational resources are designed to accommodate users with different levels of technical expertise. The website demystifies technological concepts, clarifies industry jargon, and provides useful insights to assist clients in making an informed and confident VPN choice.

1.5 Document Conventions

1.5.1 Document Conventions Used

This document uses the following conventions:

VPN	Virtual Private Network – A safe network connection that allows users to converse with one other via a public network connection much like if their computers were physically linked to a private network.
Firewall	A network security system that uses pre-established security rules to monitor and manage both incoming and outgoing network traffic.
DPI	Deep Packet Inspection
GUI	Graphical User Interface
API	Application Programming Interface
SRS	A document that specifies the functional and non-functional requirements of a software project.

MVC	A software architecture pattern that divides an application into three interconnected parts: the view (for user interaction and data), the controller (for managing user input and model changes), and the model (for data and business logic).
API calls	Requests made by one application to another through defined interfaces.
Dashboard	A visual representation of data, often providing real time insights and analytics.
Network Congestion	A condition in data networking where network is queued with so much data that it slows down or disrupts normal flow of traffic.
SRS	A document that specifies the functional and non-functional requirements of a software project.

Table 1: Document Conventions

1.5.2 Typographical Conventions

This document uses the following typographical conventions:

Font	Georgia
Italic	To draw attention to a particular point in text i.e., *Emphasis*
Underline	To enlist alternate words or to reference another part of text.
Bold	To highlight the **importance** of a Word/Phrase.

Table 2: Typographical Conventions

Chapter 2

LITERATURE REVIEW

In recent years, the proliferation of Virtual Private Networks (VPNs) has led to the development of various tools aimed at detecting and monitoring VPN network traffic. Machine learning approaches have been used in solutions such as those provided by Avnish Goel et al. and Gerard Draper-Gil et al. to analyses and categories VPN traffic patterns. But a lot of these current technologies are stand-alone programs that don't support real-time detection and need human input. On the other hand, VPN SpyGlass, our technology, is a breakthrough in VPN traffic analysis. In contrast to conventional programs that use database matching methods, VPN SpyGlass uses an advanced Python algorithm that has been trained on an ever-updating dataset of recognized VPN traffic patterns.

Compared to other similar tools in the market, our technology can offer zero-day protection against new VPN dangers thanks to this technique, which can detect such threats in real time even before they are categorized in databases that are already in place. Due to its dependence on static databases, other programs could find it difficult to identify more recent, unidentified VPN activity. In contrast, VPN SpyGlass is proactive and adaptable, always changing to counter new threats. VPN SpyGlass seeks to revolutionize VPN traffic monitoring by utilizing machine learning and real-time analysis to provide users with unmatched visibility and security in an ever-evolving digital environment.

2.1 Lightweight Anti DDoS Security Tool: Edge Level Filtering in SDN using P4.

The research paper titled "Lightweight Anti DDoS Security Tool: Edge Level Filtering in SDN using P4"[1] by Masumi Arafune, Bhargavi Goswami, Manasa Kulkarni, Nagarajan Venkatachalam, and Saleh Asadollahi focuses on addressing the challenge of packet injection attacks in Software Defined Networks (SDN). To provide edge-level filtering without relying on the control plane, the article presents a technique known as Lightweight Anti-DDoS Software (LADS). LADS seeks to provide a reliable and effective defense against malicious packet injection attacks by utilizing the P4 programming language to build filtering functions in edge switches.

The paper concludes that during packet injection assaults, LADS proved to be 100% successful in filtering malicious packets. The lightweight nature of the transition using LADS was demonstrated by the average 3% increase in CPU utilization. Measuring bandwidth revealed that LADS was able to keep genuine hosts' network performance intact while effectively isolating attackers. To improve efficiency, it is recommended that whitelisting be included for port blocking in future work. Provide a blocked port management feature to unblock hosts from being blocked indefinitely. Limitations include the quantity of hosts linked to a single switch and the kind of packet injection threats that LADS mitigates should be addressed.

2.2 Detection of VPN Network Traffic

"Detection of VPN Network Traffic" [2] is the title of a research paper written by Avnish Goel, Rochak Kaushik, Apoorv Kashyap, S. Nagasundari, B. Devesha Reddy, and Prasad B. Honnavali from PES University in Bangalore, India's CSE Department in 2022. In the digital era, the article addresses the significance of Virtual Private Networks (VPNs) in guaranteeing data security and privacy. It emphasises the necessity of VPN detection to stop dangerous actions that might take use of VPNs for anonymity, such ransomware assaults.

In this study, we develop a virtual private network (VPN) based on a widely used network security protocol and employ multiple machine learning models to identify VPN traffic. To train and evaluate their algorithms, the scientists employed a standardised dataset from the Canadian Institute for Cybersecurity. They used techniques like Random Forest and Multilayer Perceptron (MLP) for classification, and they were successful in identifying VPN traffic with high accuracy rates.

As part of the technique, an AWS VPN server was set up, network traffic data was converted from PCAP to CSV format, and machine learning algorithms were used to analyse the data. To get insights, the authors plotted the data properties pertaining to both VPN and non-VPN traffic. Additionally, real-time data analysis was done to verify the models' predictions.

2.3 Encrypted Network Traffic Analysis of Secure Instant Messaging Application: A Case Study of Signal Messenger App

The paper [3] dives into the analysis of encrypted network traffic from the Signal Messenger app, with a specific focus on forensic strategies to uncover artifacts from this encrypted data. The study aims to provide insights and methodologies that can aid forensic investigators in extracting valuable information about the app's behaviour from the encrypted traffic it generates. By leveraging network infrastructure and firewall capabilities, the research demonstrates a systematic approach to capturing and analysing the encrypted traffic, ultimately aiming to identify patterns and artifacts that can be crucial for forensic purposes.

One of the key highlights of the research is the emphasis on the security features of the Signal app, particularly its robust end-to-end encryption. While this encryption ensures a high level of security and privacy for users, it also poses significant challenges for forensic investigations due to the complexity of analysing encrypted data. The study seeks to address these challenges by proposing a strategy that can effectively navigate the encrypted network traffic to reveal important insights that can aid forensic analysts in their investigations.

2.4 A Survey on Encrypted Network Traffic Analysis Applications, Techniques, and Countermeasure

The research paper, written by Sotiris Ioannidis and Eva Papadogiannaki and published on July 13, 2021[3], addresses the difficulties that arise from the growing use of network traffic encryption. It also explains the operation of encrypted traffic inspection, with a particular emphasis on functions related to middleboxes, network analytics, security, and user privacy. Classifying encrypted communication,

identifying user behaviors, and examining quality of service/experience are all done with the use of methods like machine learning, deep learning, and neural networks. Scalability issues and the need to retrain models with new data present challenges. Machine learning is used for both malware and intrusion detection in encrypted networks and mobile devices. Evaluation criteria for machine learning include accuracy, precision, recall, and false-positive rates.

Further research into the scalability of machine learning models for classified encrypted traffic, the accuracy of these models, and the difficulties associated with malware detection on mobile devices and intrusion detection in encrypted networks are some possible directions for future work. Considering advancing encryption technologies and rising network traffic quantities, future research may concentrate on creating more effective and efficient methods for encrypted traffic analysis.

2.5 Characterization of Encrypted and VPN Traffic using Time-related Features

Written by [4] (Gerard Draper-Gil, Arash Habibi Lashkari, Mohammad Saiful Islam Mamun, and Ali A. Ghorbani, 2016), the research goes into the complexity of traffic categorization, concentrating specifically on encrypted and VPN data, and emphasises the importance of time-related factors in effectively categorising such information. The work uses machine learning algorithms such as C4.5 and KNN to provide a robust flow-based classification system that efficiently identifies VPN traffic and categorises encrypted traffic into various categories such as browsing, streaming, and VoIP. Using a dataset of encrypted communication with 14 distinct labels, the study illustrates the usefulness of time-related characteristics in traffic analysis, with accuracy rates over 80%. Notably, the study found that using lower flow timeout values improved classification accuracy, with C4.5 outperforming KNN.

The findings imply that time-related indicators can be used to reliably classify encrypted and VPN traffic. In the future, the researchers hope to broaden their study to include new applications and forms of encrypted communication, as well as investigate the use of time-based aspects in traffic categorization.

2.6 Results of Studies

The studies provide valuable insights into network security, traffic analysis, and encryption technologies. In addressing packet injection attacks in Software Defined Networks (SDN), the Lightweight Anti-DDoS Software (LADS) demonstrated a 100% success rate in filtering malicious packets, with minimal impact on CPU utilization. Future work should address limitations such as scalability and the types of threats mitigated. The study on VPN traffic detection showcased high accuracy rates using machine learning models, validating the practical applicability of the approach in identifying VPN traffic and ensuring network security.

A survey on encrypted network traffic analysis highlighted scalability challenges in machine learning models for encrypted traffic classification, emphasizing the need for continuous model retraining and future research into improving scalability and accuracy. By leveraging time-related features, the research on traffic categorization demonstrated the efficacy of machine learning algorithms in accurately classifying encrypted and VPN traffic. Future studies should explore new applications and forms of encrypted communication while refining time-based

aspects in traffic analysis.

Here's a table comparing the merits and demerits of each of the papers mentioned:

Research Paper	**Goal**	**Method Used**	**Limitation**
Characterization of Encrypted and VPN Traffic using Time-related Features [4]	The ability to identify VPN traffic and classify encrypted traffic into distinct groups based on the kind of traffic using flow-based time-related attributes.	C4.5 decision tree and KNN	1. Proposed set of time-related features achieving accuracy levels around 80%. 2. Require Labelled Dataset of previous Network Packet History
Detection of VPN Network Traffic [2]	Employing several machine learning techniques to identify and categorize network traffic as VPN or non-VPN across the standardized dataset.	Multilayer Perceptrons (MLPs) and Random Forest Model	1.The precision and recall for the same were 0.99907 and 0.92849 respectively. 2. Require Labelled Dataset of previous Network Packet History
Encrypted Network Traffic Analysis of Secure Instant Messaging Application: A Case Study of Signal Messenger App [3]	acilitate the extraction of the Signal Messenger app's behavior from encrypted network traffic for forensic investigations.	Use traditional deep packet inspection systems	Robust end-to-end encryption complicates the process of analyzing the encrypted data for forensic investigations.
A Survey on Encrypted Network Traffic Analysis Applications, Techniques, and Countermeasures [3]	Review the literature on network traffic inspection and analysis following the rise of encryption in communication channels.	Use traditional deep packet inspection systems	Only for Normal traffic. Not for VPN.
Lightweight Anti DoS Security Tool: Edge Level Filtering in SDN using P4 [1]	Defense mechanism by enabling edge-level filtering without involving the control plane.	SDN switches may become the first line of defense against packet injection attacks by integrating filtering capabilities in edge switches, which can offer an efficient and effective defense layer in SDN network systems.	1. Only for DDoS Attack 2. Isolates Data Plan from control Plane

Table 3: Literature Summary

Chapter 3

PROBLEM DEFINITION

"The proliferation of VPNs poses a challenge to network admin, as these tools can bypass security measures and access restricted content pose security risks, violate agreements, and slow network performance. Moreover, Existing methods often struggle to identify VPN traffic due to encryption and masking techniques employed by VPN services."

3.1 The Challenge of VPN Security

By encrypting internet traffic and hiding customers' IP addresses, VPN services are frequently used to improve online privacy and security. Nonetheless, several security and dependability problems beset the VPN sector, such as:

- Data breaches: Sensitive user information was made public by several VPN companies due to data breaches.
- Insufficient encryption: Several VPN providers have poor encryption implementations or inadequate encryption standards, which exposes customers to hackers and spying.
- Logging practices: Even with purportedly zero-logs practices, certain VPN service providers could gather and retain customer data, jeopardising their privacy.
- Phishing and malware: Users run the risk of connecting to VPN servers that are unintentionally used for phishing attempts or infected with malware, endangering their devices and data.

3.2 Limitations of Existing VPN Evaluation Tools

The shortcomings of the VPN service evaluation methods now in use make it challenging for them to provide comprehensive and impartial assessments. These limitations include:

- Lack of transparency: A lot of VPN evaluation tools rely on outdated or erroneous data, which makes it difficult for users to make informed decisions.
- Bias: Certain VPN review systems may be influenced by financial incentives or affiliations with VPN providers, leading to recommendations that are skewed.
- Restriction on criteria: Many VPN evaluation tools focus just on speed, server coverage, or pricing, ignoring important factors like privacy policies, security features, and jurisdiction.

3.3 Proposed Solution

To address the drawbacks of the available VPN assessment tools, we provide VPN SpyGlass, a comprehensive platform for evaluating and contrasting VPN services. VPN SpyGlass aims to provide users with accurate, unbiased, and up-to-date information so they may choose the best VPN for their needs. Important features of VPN SpyGlass include:

- Diverse evaluation: VPN SpyGlass considers a wide range of characteristics when evaluating VPN services, including server design, jurisdiction, logging policies, encryption methods, and independent security audits.
- Vulnerability and objectivity: VPN SpyGlass runs without interference from VPN providers and turns down sponsorships and other financial incentives that may influence its evaluations. Offering consumers accurate and fair information, the website is committed to transparency and objectivity.

- Using VPN to empower users SpyGlass helps users make informed decisions by providing them with detailed information about the benefits and drawbacks of every VPN provider. To choose the VPN that best meets their requirements, customers may assess VPN providers based on their personal preferences and priorities.

3.4 Real-Time Monitoring and VPN Database

VPN Real-time VPN service monitoring is provided by SpyGlass to spot any new security or reliability issues. The portal regularly updates its vast traffic packets of VPN services with new information on encryption methods, server locations, logging policies, and other relevant aspects. Customers may access a comprehensive profile of each VPN service, which includes user ratings, reviews, and recommendations.

3.5 Leveraging Advanced Technologies

VPN SpyGlass uses state-of-the-art technologies to enhance its evaluation capabilities. It can identify patterns and trends and assess massive traffic packets. By employing technology, VPN SpyGlass provides administrator with unparalleled insights into the quality and reliability of VPN services.

3.6 Conclusion

A revolutionary platform that addresses problems with the available VPN assessment tools is VPN SpyGlass. With VPN SpyGlass, users can make informed judgements regarding their online safety and privacy since it provides rapid, unbiased, and transparent evaluations of VPN providers. VPN Spyglass's comprehensive approach to VPN assessment and state-of-the-art technology set new standards for transparency, objectivity, and user empowerment in the VPN industry.

.

Chapter 4

METHODOLOGY

3.1 Method and Approach

The core of our technique is Deep Packet Inspection (DPI), a sophisticated packet analysis tool that allows us to examine network traffic at the individual packet level. DPI enables us to obtain comprehensive information from packet payloads, such as application-specific content and header data. DPI allows for the differentiation of VPN traffic from other network traffic in several network data sets, which aids in classification and analysis.

We prioritise responsiveness and flexibility throughout the development process by using an Agile methodology. This iterative process enabled us to abandon our first idea, which required using switches and the P4 language, in favour of a more reliable solution that takes use of the PfSense firewall. We also updated to Django for our application framework and migrated to MongoDB for our database. In the rapidly evolving world of VPN assessment tools, this strategic decision allows us to adapt more swiftly to shifting needs and technical improvements, keeping VPN SpyGlass innovative and flexible.

3.1.1 Web Dashboard

Our Web dashboard has a clean and simple layout based on ReactJS and Vite, allowing for seamless display of real-time network traffic metrics. The dashboard has detailed graphs that display how much VPN is utilised. Furthermore, it categorises the traffic into multiple VPNs that are in operation, giving clients extensive visibility into their network activity and supporting effective VPN traffic management.

3.1.2 Cloud Database

We chose the MongoDB Atlas Cloud platform as our database option to ensure scalability and dependability. MongoDB's adaptive document-based approach allows us to effectively store and retrieve network traffic data. Every packet's information is recorded as JSON objects, which contain the length, protocols, and category within the VPN in use. This structure simplifies and speeds up data retrieval. Furthermore, MongoDB has comprehensive security features that effectively secure the confidentiality and integrity of our data, hence increasing the reliability of our platform.

3.1.3 Hosting

Vercel's Platform, we were able to host our web application online and benefit from Vite's smooth deployment process for React.js applications. Vercel stands out with powerful features like SSL encryption, continuous deployment, and global CDN, which provide our web dashboard the best speed and security available. One of Vercel's main advantages is its ability to set up distinct environment variables for increased security. By adopting Vercel, we can guarantee excellent uptime for our online dashboard without having to pay for hosting.

3.1.4 Categorization Algorithm

Our algorithms classify VPN traffic based on features including VPN protocols, IP addresses, port numbers, payloads, and traffic behaviour by utilising Deep Packet Inspection (DPI) techniques. Our algorithms generate rules based on these characteristics and analyse the contents of the packets to distinguish between different types of VPN communication. Subsequently, they assign the VPN to which

each network traffic belongs. This fine-grained segmentation facilitates resource management and informed decision-making by enabling administrators to effectively monitor and regulate VPN usage on the network.

3.2 Deep Packet Inspection Algorithm

Deep packet inspection (DPI), our project's major component, enables us to thoroughly scrutinise data packets as they travel over the network. Unlike older approaches that just scan packet headers, DPI investigates packet contents in depth and gives a full insight of network traffic patterns. DPI is an essential component of our design, allowing us to recognise and categorise packets based on critical information such as IP addresses, protocols, and port numbers. By meticulously studying VPN traffic traces, we may detect specific patterns unique to each VPN service. These patterns are derived from IP addresses, port numbers, and packet payloads and serve as the foundation for developing robust rules that correctly identify VPN activity from conventional network traffic.

In our project, we employ DPI to give network managers the ability to accurately categorise and regulate VPN usage across the network. This innovative method ensures the efficient use of network resources while also increasing security and resource allocation. Our approach improves the overall efficacy and security of the network infrastructure by enabling proactive management of VPN traffic and informed decision-making with DPI.

3.3 Tech Stack

Component	Technologies Used
Web Dashboard	React Js Vite Typescript Servers
Cloud Database	Mongodb Atlas PostMan Node Js Express Js
Hosting	Vercel app
Categorization Algorithms	DPI Python
Network Control	Python PfSense Firewall

Table 4: Tech Stack

Chapter 5

SOFTWARE REQUIREMENTS SPECIFICATION

5.1 Requirements Elicitation

5.1.1 Intended Audience and Reading Specifications

The SRS is intended for programmers, project managers, testers, and reviewers of documentation.

The project, its goal, and its scope are introduced in the opening section of the text. The project is described in detail in the second section of the document, which also includes a walkthrough of the fundamental procedures to implement the functionalities, a description of the modes in which the application can be used, and functional and non-functional requirements and constraints.

- To ensure appropriate implementation, developers will use it as a fundamental guide to comprehend the specific functional and non-functional needs of the product.
- To coordinate their testing efforts with the anticipated results listed in the specification, testers will use the SRS to create thorough test cases.
- To track project progress, manage resource allocation, and make sure the development process complies with the specified criteria, project managers will consult the SRS.
- Users should have a fundamental awareness of the application's purpose and a quick grasp of how to utilize it.
- To understand the sequence of changes that have occurred, documentation writers should study earlier versions of the documentation.

5.1.2 Product Perspective

The "VPN SpyGlass" project is a complete network monitoring and management solution for detecting and categorizing VPN activities within a network. It functions within the context of network security and management, supporting network administrators in monitoring and managing VPN usage.

5.1.2.1 System Architecture

The system follows a *client-server model*, consisting of two main components (Fig 1):

1. **Packet Analysis**: Deep packet inspection and VPN traffic identification are performed by this server-side component. It looks at network packets and uses features like IP addresses, port numbers, and packet durations to distinguish between regular and VPN traffic. Depending on the VPN protocol being utilised, detected VPN traffic is categorised.
2. **Admin Dashboard**: Via web browsers, the client-side component offers a graphical user interface for real-time monitoring and control. It establishes a connection with the server-side Packet Analysis Engine to retrieve updates and command instructions.

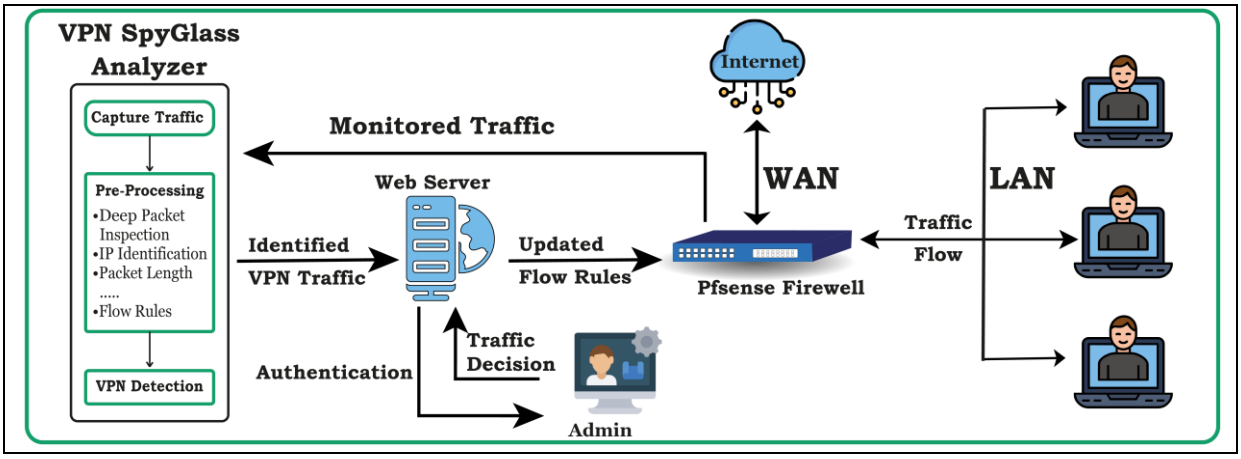


Figure 1: System Architecture

5.1.2.2 *Interaction between Components*

The Packet Analysis Engine continuously examines the incoming network packets. The Engine performs API calls to deliver relevant data to the Web Dashboard when VPN activity is detected. The Web Dashboard is used to visualize this data and indicates any VPN activity that has been found, along with real-time traffic fluctuations. The Dashboard may be used by administrators to initiate further procedures, such as blocking VPN traffic using firewall rules.

5.1.3 Product Functions

Our product automates the following functions for VPN traffic detection and monitoring:

5.1.3.1 *Packet Capture and Inspection*

- Capture network packets using Wireshark for analysis.
- Perform deep packet inspection to extract relevant packet attributes.

5.1.3.2 *VPN Traffic Identifications*

- Analyze packet attributes including IP addresses, port numbers, and packet lengths.
- Detect and identify VPN traffic based on predefined patterns.

5.1.3.3 *VPN Protocol Classification*

- Further classify detected VPN traffic based on the specific VPN protocol or type being used.

5.1.3.4 *VPN Traffic Blocking*

- Allow authorized administrators to block specific VPN traffic.
- Utilize OpenFlow/Firewall rules to implement traffic blocking on demand.

5.1.3.5 *User Authentication and Access Control*

- Implement user authentication to control access to the web dashboard.
- Ensure that only authorized users can view and control the system.

5.1.3.6 *Graphical Representation of Insights*

- Present data analytics and VPN traffic insights graphically in the web interface.

- Provide visualizations such as charts and graphs to represent network traffic trends.

5.1.3.7 Real-time Update of Insights

- Update graphical representations and visualizations dynamically as new VPN traffic is detected.
- Ensure that administrators have access to the most recent data and insights.

5.1.4 User Classes and Characteristics

5.1.4.1 Network Administrator

Responsible for managing and securing the network infrastructure. Have knowledge of network protocols, security measures, and traffic analysis techniques. Proficient in using technical tools for network monitoring and analysis.

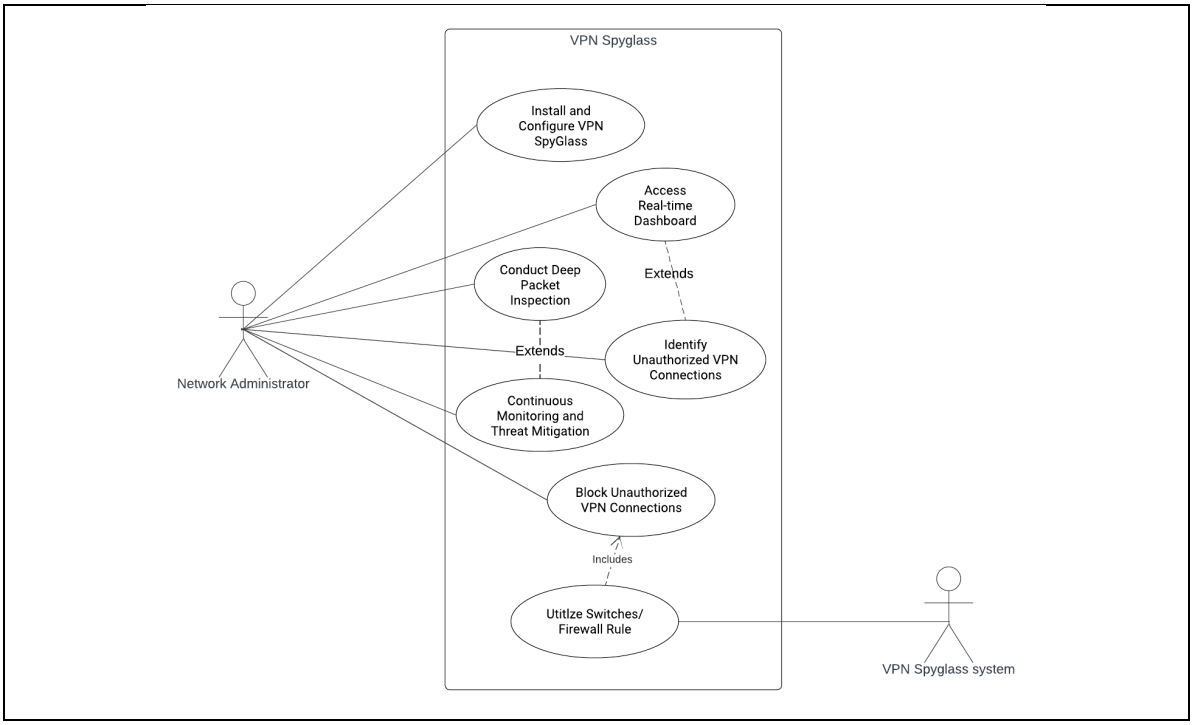


Figure 2: Use Case Diagram

Use the VPN SpyGlass dashboard to monitor and analyze network traffic in real-time (Fig 2). Configure settings and rules to detect and categorize VPN traffic accurately. Utilize the dashboard to identify unauthorized VPN connections and potential security risks. Have the ability to block specific VPN traffic if required.

5.1.4.2 Software Developers

Program developers are responsible for designing, developing, and maintaining the VPN SpyGlass application. They possess strong programming skills and expertise in relevant technologies. Developers ensure the system's functionality, reliability, and adherence to design specifications.

Developers design and implement the deep packet inspection algorithm and the packet analysis engine responsible for VPN traffic detection and categorization. They create the web dashboard using ReactJS / NextJS, NodeJS, and Python to provide a user-friendly interface for administrators. Developers integrate the different backend components, such using PfSense Firewall, to ensure real-time data transfer and control. They rigorously test the application's accuracy, performance, and security, identifying and resolving any issues.

5.1.5 Operating Environment

The program is an online application that can be accessed with any web browser that supports the **Secure Hyper Text Transfer Protocol (HTTPs).** The program will be installed on a cloud server. The website will be hosted **by Vercel**. The client-side web app will be accessible through web browser on all devices. All operating systems, including Windows, MAC OS, Linux, and Android, will use the same version of the program.
Basic Browser- Latest Compatible Versions are as:

Google Chrome -110	*Opera - 92*
Mozilla Firefox - 106	*Microsoft Edge – 103*

Table 5: Compatibility

The backend implementation involves use of following:

Linux Ubuntu 22.04	Nodejs v20.12.2	Wireshark
PfSense	Express	Python

Table 6: Dependencies

5.1.6 Design and Implementation Constraints

The design and implementation constraints will include hardware, software, and security-usage constraints.

5.1.6.1 Timing Constraints

The overall size of all webpages should not exceed 5 Megabytes and the load time of a website should be kept under 3 seconds, so that page loads quickly on 97% of client devices.

Moreover, ensure that the tool's detection and analysis processes don't introduce significant delays, especially for real-time monitoring purposes. The web dashboard should provide real-time updates and responses, especially when administrators interact with it to block VPN traffic.

5.1.6.2 Design Constraints

The development process will be constrained by a number of implementation restrictions. The web dashboard should be designed to be responsive, ensuring that it works and displays properly on various screen sizes and devices, including desktops, tablets, and smartphones.

The user interface should be intuitive and user-friendly to facilitate easy navigation and interaction for network administrators. Maintain a consistent design and user experience across all sections of the web dashboard to avoid confusion. To promote readability and minimize congestion in the physical design, font sizes are fixed between 18 and 30.

5.1.6.3 Programming Constraint

To ensure that the website is accessible on all devices and is always updated, and the language doesn't get outdated too soon, therefore, the website is designed using JavaScript, HTML, and CSS. **The React JS library** is one of the most updated programming libraries that can be used as React is a cross platform and platform independent language.

Moreover, code should be written clean and well-documented to ensure that

the project remains maintainable and understandable as it evolves. The code should be optimized for efficiency, especially when dealing with packet analysis and real-time monitoring to prevent performance bottlenecks.

5.1.6.4 Memory Constraints

Cookies will be stored for each individual user, storing login information as well as the user's consultation history. If the user has not been active in the past three months, the cookies will be erased.

For using PfSense firewall, the system will need minimum of 1 GB ram and 2 GB of Hard Drive to keep the system running.

Furthermore, space is necessary on the system to construct the virtual version of the PfSense Firewall and Website Server. A minimum of 50 GB of space is necessary to keep the system running.

5.1.6.5 Hardware Constraints

The project requires a P4 programmable switch with Intel Tofino 2 chip. However, it is not available in the country and costs a lot therefore, virtual environment will be preferred instead of actual hardware.

Moreover, we can use firewalls like PfSense to implement it. As a software-based firewall, it can also be implemented using a PC or Laptop solely used for this purpose.

5.1.7 User Documentation

While launching the product, the user will be provided with guidelines which will help system administrators to use the product in the most efficient way possible.

5.1.7.1 Video Tutorial

Upon logging in or signing up for the VPN Spyglass website, you'll have access to a comprehensive video tutorial that covers all aspects of the tool. This tutorial aims to provide step-by-step guidance on how to use VPN Spyglass, ensuring that you can quickly grasp its functionality and features.
The video tutorial will cover the following topics:

- Introduction to VPN Spyglass
- Navigating the Dashboard
- Packet Analysis and Detection
- Real-time Monitoring
- Blocking VPN Traffic
- Using the Question Sections
- Troubleshooting Tips

5.1.7.2 FAQs

To further support users, VPN Spyglass features a dedicated section where you can find solutions to the most frequently asked questions. If you're facing a common issue or looking for specific information, the Question Sections can provide you with quick answers and solutions.

Some of the common topics covered in the Question Sections include:

- Getting Started with VPN Spyglass
- Troubleshooting and Error Resolution
- Understanding Packet Analysis and Detection
- Using Real-time Monitoring and Traffic Blocking
- Tips for Effective Network Management

By referring to the Question Sections, you can quickly find answers to queries that others have encountered, saving you time, and ensuring a smooth experience while using VPN Spyglass.

5.1.8 Assumptions and Dependencies

5.1.8.1 Dependencies

The dependencies include the following:

- **Packet Capturing Tool**: The system depends on a packet capturing tool like Wireshark or a similar utility to collect network traffic data.
- **Operating System Compatibility**: The tool's packet capturing component may have dependencies on specific operating systems (e.g., Windows, Linux) or versions of those operating systems.
- **Web Server**: The web dashboard component relies on a web server (e.g., Apache, Nginx) to host and serve the dashboard application.
- **Real-time Data Processing**: Real-time data processing and forwarding to the web dashboard may depend on a messaging system (e.g., WebSocket) or event handling framework.
- **Specific Versions of Software**: The product requires the system to have specific versions [1] of software to work properly.

5.1.1.1 Assumptions

The assumptions include the following:

- **User's Network Configuration**: It's assumed that the user has the necessary network permissions to capture and analyze network traffic, as well as to make changes to network rules if blocking VPN traffic is implemented.
- **Web Browser Familiarity**: Users are assumed to have a basic familiarity with using web browsers, including navigation, accessing web pages, and interacting with web elements.
- **Networking Knowledge**: Users are assumed to have a fundamental understanding of networking concepts, such as IP addresses, ports, and protocols, to interpret the information presented by VPN Spyglass accurately.
- **Web Dashboard Interaction**: Users are assumed to know how to interact with the web dashboard, including using its features, buttons, and controls effectively.
- **Internet Connectivity**: Users are assumed to have an internet connection to access the web dashboard and receive updates or alerts from VPN Spyglass.

5.2 External Interface Requirements

5.2.1 User Interfaces

The user interface for the VPN SpyGlass software shall consist of a web dashboard accessible through modern web browsers, including but not limited to Internet Explorer, Mozilla Firefox, Google Chrome, and Safari. The interface shall provide the following functionalities:

- Live Traffic Monitoring – Display real-time network traffic with color-coded indicators for regular and VPN traffic. Highlight and differentiate detected VPN traffic for easy identification.

[1] 5.1.5 Operating Environment - backend implementation

- VPN Traffic Details – Show detailed information about identified VPN traffic, including IP addresses, port numbers, and protocols.
- VPN Blocking – Provide an option to block specific VPN traffic. Allow administrators to define OpenFlow/Firewall rules for traffic blocking.
- User Authentication – Require user authentication through a secure login process before granting access to the dashboard.
- Dashboard Configuration – Allow administrators to customize the dashboard layout, views, and preferences.

The user interface shall be developed using ReactJS / NextJS for front-end interactivity and NodeJS for backend interactions.

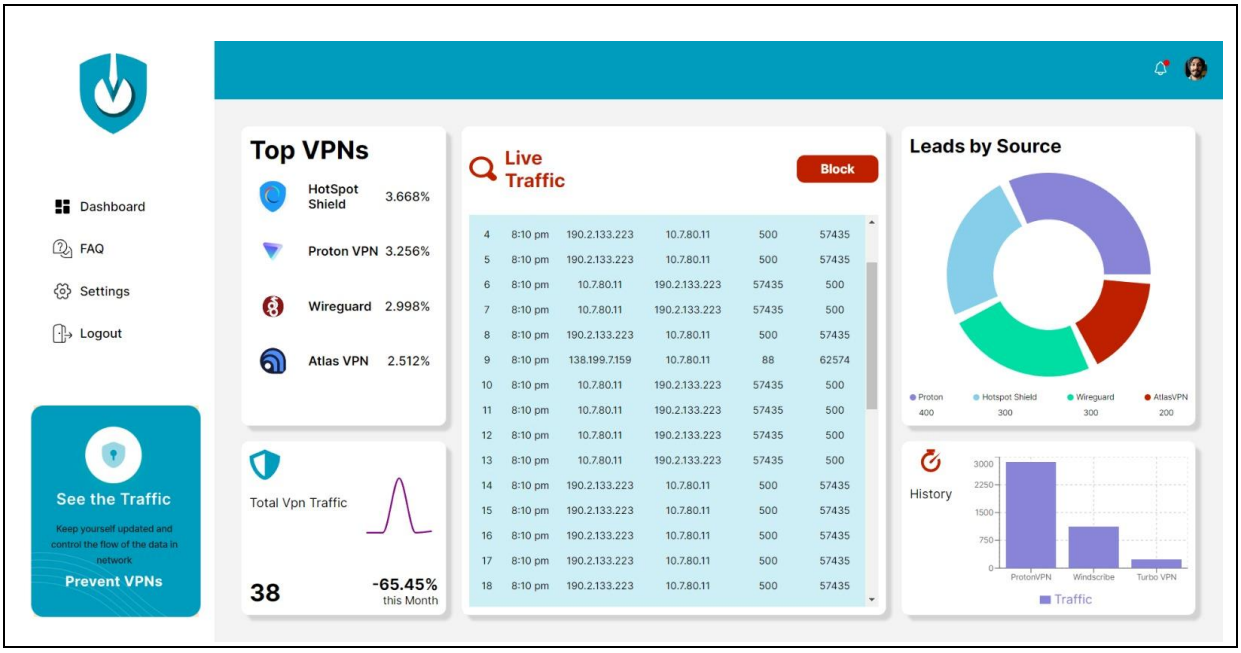


Figure 3: GUI VPN SpyGlass

5.2.2 Hardware Interfaces

- **Network Hardware**: The system requires network devices (routers, switches, ethernet wire etc.) to capture and analyze network traffic.
- **Server Hardware**: A dedicated server or cloud infrastructure is needed to host the web dashboard and backend components.
- **Client Devices**: Users will access the web dashboard using devices such as laptops, desktops, tablets, and smartphones.

5.2.3 Software Interfaces

- **Operating System**: The system is compatible with multiple operating systems including Windows, macOS, and Linux.
- **Web Browsers**: The web dashboard supports modern web browsers such as Google Chrome, Mozilla Firefox, Safari, and Microsoft Edge.
- **Wireshark**: The packet analysis engine utilizes Wireshark for capturing and inspecting network packets.
- **ReactJS / NextJS**: The frontend of the web dashboard is built using the ReactJS / NextJS.

- **NodeJS**: The backend of the web dashboard is developed using the NodeJS framework.
- **Python**: The packet analysis engine and backend components are developed using the Python programming language.
- **PfSense**: Software based Firewall used to control the flow of traffic inside a network.

5.2.4 Communications Interfaces

- **HTTP/HTTPS**: Communication between clients and the web dashboard occurs over HTTP/HTTPS protocols.
- **API Endpoints**: The web dashboard communicates with backend components using RESTful API endpoints.
- **Wi-Fi/Ethernet**: The system captures network traffic through Wi-Fi or Ethernet interfaces depending on the network setup.
- **SSH**: To communicate between PfSense and the web dashboard, SSH commands and port communication will be used.
- **Firewall Rules Protocol**: Communication between the web dashboard and the PfSense occurs using SSH and Web GUI for controlling the firewall's behavior.
- **Interactive Forms**: The web dashboard uses interactive forms to display real-time traffic updates, VPN detection results, and user controls for VPN blocking.

5.3 System Features

5.3.1 Network Administrator

5.3.1.1 User Authentication

Description: Network administrators can log in to the VPN SpyGlass dashboard using their credentials.
Priority: High
Stimulus/Response Sequences:

- The network administrator accesses the VPN SpyGlass dashboard.
- The system prompts the administrator to enter their login credentials.
- Upon successful authentication, the administrator gains access to the dashboard.

5.3.1.2 Real-Time Traffic Monitoring

Description: Network administrators can monitor real-time network traffic using the dashboard.
Priority: High
Stimulus/Response Sequences:

- The network administrator logs in to the dashboard.
- The dashboard displays real-time traffic visualization.
- VPN traffic is highlighted for easy identification.

5.3.1.3 VPN Traffic Categorization

Description: The system automatically identifies and categorizes VPN traffic based on deep packet inspection.

Priority: High
Stimulus/Response Sequences:

- The network administrator observes the dashboard.
- VPN traffic is categorized and labeled by the system based on analysis results.

5.3.1.4 Unauthorized VPN Detection

Description: The dashboard helps network administrators identify unauthorized VPN connections.
Priority: High
Stimulus/Response Sequences:

- The network administrator reviews the dashboard.
- Unauthorized VPN connections are flagged, and alerts are generated.

5.3.1.5 VPN Traffic Blocking

Description: Network administrators can block specific VPN traffic using OpenFlow rules through the dashboard.
Priority: High
Stimulus/Response Sequences:

- The network administrator accesses the dashboard.
- In cases of security concerns, the administrator selects specific VPN traffic to block.
- OpenFlow rules are applied to block the selected traffic.

Action	Response
User Authentication	• The network administrator accesses the VPN SpyGlass dashboard. • The system prompts the administrator to enter their login credentials. • Upon successful authentication, the administrator gains access to the dashboard.
Real-Time Traffic Monitoring	• The network administrator logs in to the dashboard. • The dashboard displays real-time traffic visualization. • VPN traffic is highlighted for easy identification.
VPN Traffic Categorization	• The network administrator observes the dashboard. • VPN traffic is categorized and labeled by the system based on analysis results.
Unauthorized VPN Detection	• The network administrator reviews the dashboard. • Unauthorized VPN connections are flagged, and alerts are generated.
VPN Traffic Blocking	• The network administrator accesses the dashboard. • In cases of security concerns, the administrator selects specific VPN traffic to block.

	• OpenFlow rules are applied to block the selected traffic.

Table 7: Network Administrator Features

5.3.2 Software Developers

5.3.2.1 DPI Algorithm Implementation

Description: Software developers design and implement the deep packet inspection (DPI) algorithm for identifying VPN traffic.
Priority: High
Stimulus/Response Sequences:

- Developers work on implementing the DPI algorithm.
- DPI algorithm analyzes network packets for VPN traffic patterns.

5.3.2.2 Web Dashboard Development

Description: Developers create the web dashboard using ReactJS with NextJS, NodeJS, and Python.
Priority: High
Stimulus/Response Sequences:

- Developers design and develop user-friendly web dashboard.
- The dashboard provides real-time traffic monitoring and control features.

5.3.2.3 Backend Integration

Description: Developers integrate backend components, including using Firewall to enable real-time data transfer and control.
Priority: High
Stimulus/Response Sequences:

- Developers ensure seamless integration of backend components for data collection and control.
- OpenFlow / PfSense flow rules are implemented for traffic blocking.

5.3.2.4 System Testing

Description: Developers rigorously test the application for accuracy, performance, and security.
Priority: High
Stimulus/Response Sequences:

- Developers conduct thorough testing to identify and resolve any issues.
- Performance and security vulnerabilities are addressed during testing.

5.4 Other Nonfunctional Requirements

5.4.1 Performance Requirements

- The system shall process and categorize incoming packets for VPN detection in real-time with minimal latency.
- The web dashboard shall load and display real-time traffic updates within 2 seconds.
- The deep packet inspection algorithm shall process a minimum of 1000 packets per second.

- The tool shall handle up to 500 simultaneous users accessing the web dashboard without experiencing performance degradation.
- The system should provide accurate VPN detection with a false positive rate of less than 5%.
- The system must be capable of satisfying 99/100.

5.4.2 Security Requirements

- User authentication for the web dashboard shall use strong encryption (e.g., HTTPS) to protect login credentials.
- The system shall not log or store any sensitive user data or network traffic data.
- Access to the web dashboard shall be protected against unauthorized access.
- The system shall comply with relevant data protection and privacy regulations.
- The system shall not introduce vulnerabilities into the network it is monitoring.

5.4.3 Safety Requirements

- The system shall not interfere with the normal operation of the network it is monitoring.
- In the event of a system failure or crash, it shall recover and resume normal operation within 1 minute.

5.4.4 Software Quality Attributes

- The admin panel's user interface (UI) should be basic and straightforward, taking no more than two hours for new users to become acquainted with.
- The system will maintain track of logs for auditing purposes, ensuring that they are securely kept and only accessible by permitted users.
- The system should provide regular software upgrades and patches to address security vulnerabilities and improve functionality.
- The web dashboard must be responsive and compatible with all major browsers, including Chrome, Firefox, and Safari.
- The system must be scalable to easily expand to handle larger networks.

5.4.5 Business Rules

- The system must not restrict VPN traffic without administrator permission.
- Administrators must be able to customize and fine-tune VPN detection criteria to meet unique network needs.
- The system must keep an audit record of VPN blocking activities for accountability.
- Only authorized administrators have access to the VPN blocking capability.
- The system shall not ban VPNs used for legitimate business purposes unless the user explicitly consents.

Chapter 6

SOFTWARE DESIGN SPECIFICATIONS

6.1 Design Methodology and Software Process Model

In the development of our project, a meticulous choice of design methodologies has been made for the creation of an efficient system. The project being divided into two parts, the front-end and back-end required us to ponder on the most suitable approach for both while maintaining consistency in the project. The front-end is crafted using React, following procedural design methodology, which complements Reacts component-based architecture and declarative approach. Simultaneously, the backend is implemented in NodeJS using python, embracing procedural principles. This not only provides consistency throughout the project but also enables a systematic and structured development process providing clarity in both user interface design and data management.

In the subsequent sections, the choice of the procedural design methodology and process model to be used will be explored and justified accordingly and in greater detail.

6.1.1 Design Methodology

While `***Procedural Approach***` is not a common design methodology being used by the developers, the modern tool like React uses a declarative approach for the development of the client-side of the project. Procedural Approach is often a valid approach for the backend development using NodeJS.

The rationale for employing the procedural design methodology on both the client-side and server-side modules is expounded upon in the subsequent explanation.

6.1.1.1 Website based dashboard.

The utilization of a procedural design methodology in developing a React-based website dashboard is justified as for the following reasons:

- **Modular Component Composition:** React uses Component-based Architectural approach so procedural design facilitates the decomposition of the dashboard into modular components, enhancing the maintainability and clarity in Structuring UI elements.
- **Effective Event Handling:** Procedural design effectively handles user interactions and events, enabling the definition of specific procedures for known tasks related to event-driven functionalities.
- **Structured Control Flow:** The procedural paradigm aids the component-based Architecture of React as it allows for a systematic control flow, aiding in step-by-step execution of data processing tasks, contributing to a well-organized codebase within the dashboard components.

6.1.1.2 Integration of Firewall

The adoption of a procedural design methodology for integrating a Firewall into React-based application within the NodeJS Framework, rather than opting for an object-oriented approach is substantiated by:

- **Modularity and Code Separation:** Procedural design streamlines the process of integrating Firewall by separating concerns into distinct procedures, enhancing maintainability. In contrast, an object-oriented approach might

complicate modularization with a more interconnected class structure. The procedural approach's emphasis on discrete procedures aligns well with the segmented nature of Firewall operations, resulting in simpler and modular codebase.

- **Simplicity and Readability:** Procedural Design approach with its emphasis on functional programming approach, can often lead to more straightforward and readable code. Integration of Firewall allows to execute only specific operations and procedures so in this case procedural approach enhances code clarity while using OOP might end up into unnecessary creation of classes having complex relation with each other.
- **Practicality and Specific Use Cases:** For certain functional requirements such as configuring and managing Firewall rules through a request from client-side, a procedural approach may be more practical and directly aligned with the task's procedural nature. OOP, while powerful in other contexts, might introduce unnecessary complexity for this specific use case.

6.1.1.3 User Authentication

The utilization of a procedural design methodology for the development of a database storing the user data for authentication purposes is justified as for the following reasons:

- **Modular Authorization Logic:** Chosen Design approach enhances code modularity in NodeJS, allowing discrete steps for user authentication for maintainability and readability purposes. Using OOP may increase the complexity for straight forward authentication. It will add unnecessary abstraction layer to simple authentication logic.
- **Compatible with NodeJS' Class-Based Views.:** Procedural design smoothly interacts with NodeJS's class-based views, following framework rules. OOP, while competent, has a longer learning curve for developers who are new with the framework's conventions.

6.1.2 Software Process Model

Given the ever-changing world of network dynamics and security issues, ***'Agile (Iterative) Development'*** is the most appropriate software process paradigm. This software process architecture easily handles frequent project development changes. Agile allows you to discard or alter project functionality at any point in the Software Development Life Cycle (SDLC) without impacting subsequent phases. More advantages related to this process model will be discussed in further detail in the next section.

- **Iterative Development:** The dynamic nature of network security coincides with the agile iterative methodology, allowing for regular evaluation and adaptation to evolving needs. Its incremental feature delivery model is ideal for VPN Spyglass, allowing for the progressive development and delivery of functional needs depending on priority.
- **Collaboration and Adaptability:** The essential role of cross-functional teams in Agile is evident in VPN Spyglass, promoting collaboration between development of User Interface and its integration with database and Firewall. The iterative feedback loop in Agile ensures constant input, crucial for enhancing the services of our project.
- **Flexibility and Adaption:** It accommodates changes in the security landscape, enabling the team to address new threats and regulatory requirements in

subsequent iterations. The model's responsiveness ensures prompt updates to security measures and integration of new features of modifying the dynamics of built features.

6.2 System Designs and Overviews

VPN Spyglass is implemented using ReactJS, NodeJS and Python. ReactJS is a powerful JavaScript framework which will be used in the frontend or user interface of our Project. NodeJS provides a set of tools and conventional method making the data handling tasks very convenient, so it is being as backend language.

The main purpose of the system is to give meaningful analysis of the network traffic and clear categorization between normal traffic and VPN traffic.

6.2.1 System Operations and Requirements coverage

VPN Spyglass, our innovative project, provides an online portal for login/register. The authenticated users are provided with real-time visibility into network traffic dynamics. Through a web-based dashboard, users gain insights into the percentage of VPN traffic within a network, uncovering crucial information about the state of the network. The dashboard also highlights the top VPNs currently in use as well as ranks the most utilized VPNs over past week, empowering the network administrators to take timely decisions about network security and its optimization.

In this system, the database plays a crucial role in storing user information and related activities. Live network traffic is dynamically displayed through API calls. Users can proactively manage the network using the analytics provided by the dashboard, enabling actions such as blocking a specific VPN in response to network congestion.

6.2.2 Architectural Design

The system's modular structure is designed to achieve comprehensive functionality through the collaboration of distinct modules *(as shown in Figure 1)*. The high-level overview aims to provide a general understanding of the system's decomposition and the interplay between the individual modules.

6.2.2.1 Inter-Module Dependencies

6.2.2.1.1 User Interfaces

- User Interfaces represent the font-end layer.
- Communicates user inputs and requests to the corresponding module.

6.2.2.1.2 Controllers

- Serves as the intermediary between the UI components and the backend modules.
- Synchronize the flow of requests and corresponding actions to be taken.

6.2.2.1.3 Real-time Traffic Module

- Utilizes API calls for real-time traffic information.
- Collaborates with VPN categorization module to distinguish VPN traffic from regular network traffic.

6.2.2.1.4 VPN Categorization Module

- Collaborates with the Live Traffic module to categorize traffic and distinguish VPN connections.
- Feeds the distinguished data to the controller for further processing.

6.2.2.1.5 VPN Blocking Module

- Implements logic to monitor VPN traffic thresholds and dynamically block VPN connections.
- Also allows manual blocking of specific VPNs based on user or system decisions.

6.2.2.1.6 Backend Services

- Comprises multiple modules handling specific functionalities (e.g., data preprocessing, business logic or external integrations.
- Communicates with each other through well-defined interfaces.

6.2.2.1.7 Database Management

- Manages data storage and retrieval for the system.
- Connected to backend modular services for seamless data flow.

6.2.3 Box and Line Architecture Diagram

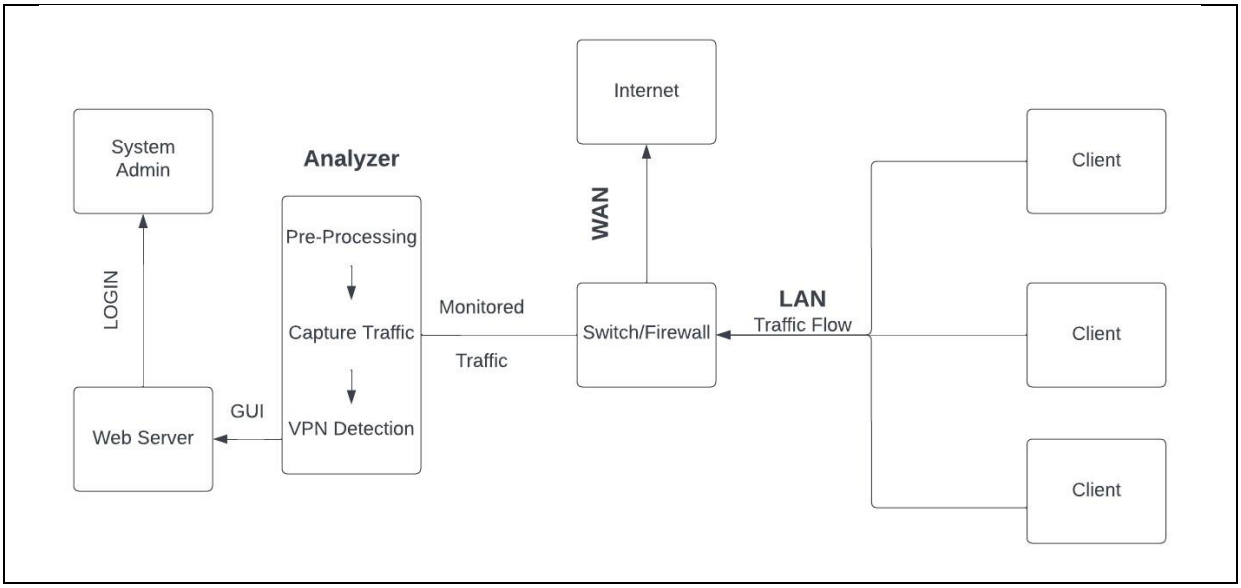


Figure 4: Box and Line Architecture

6.2.4 Three-tier Client-Server Architecture

The ***`three-tier client-server model`*** is a widely adopted architectural pattern that organizes software application into three distinct layers, each responsible for specific functions.

The utilization of this model is justified by its capability to enhance modularity, scalability, and maintainability by separating the user interface (presentation tier), application logic (application tier) and data management (data tier) *(as shown in Figure 5)*. This division of responsibilities simplifies development, maintenance, and scalability of software systems.

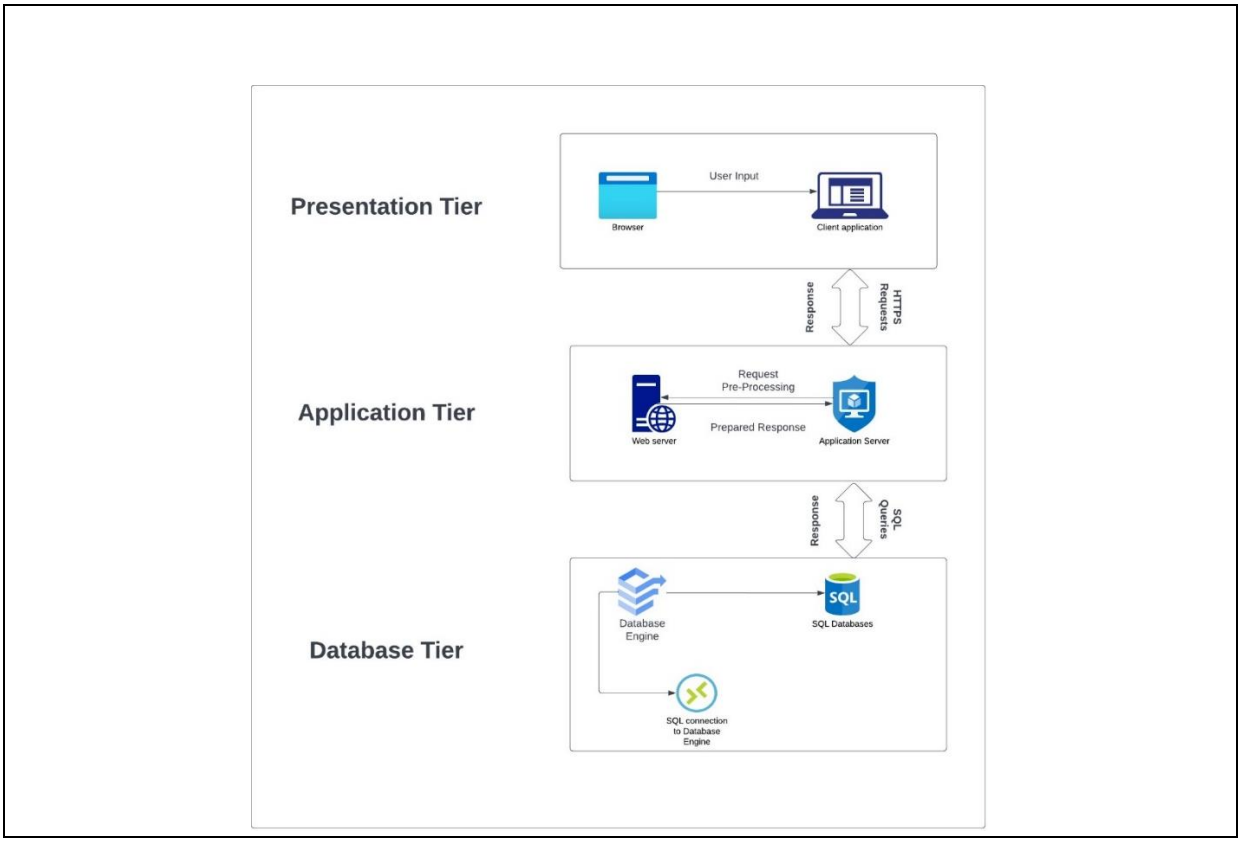


Figure 5: Client-Server Architecture

.

6.2.4.1 Project Modules in Three-Tier Model

Now let's map the modules/components of VPN Spyglass to the three-tier client-server model.

Presentation Tier: The web dashboard developed with ReactJS, NodeJS and python, resides in this tier. It allows us to interact with the application layer of VPN detection system.

Application Tier (Server): The backend components involving Python, NodeJS, ExpressJs and the web server, are part of the application tier. It manages the business logic, communication with the data tier, and overall functionality of the system.

Data Tier (Database): The data tier is Mongo database which will be used to store data for user authentication.

The interactions between the intermodular components and how they communicate with each other is shown below:

1. **Browser to Client-Side Application:** The user interacts with the client-side application in the presentation tier.
2. **Client-Side Application to Application Server:** Requests for data or services from the application tier.
3. **Application Server to Business Logic Layer:** Transfer requests from the presentation tier to the business logic layer for processing.
4. Application Server to Data Tier: Retrieves or updates data in the data tier based on requests from presentation tier.

6.3 Design Models

6.3.1 Activity Diagram

6.3.1.1 User Authentication Process

The User Authentication Process is an essential part of the VPN Spyglass system. It guarantees the system's security and integrity by authenticating people who seek to access it. This procedure is critical because it protects the system from unauthorized access, preserves sensitive data and resources, and ensures that only authorized staff may use the VPN Spyglass network monitoring and management capabilities.

The process of User Authentication can be expressed in following steps:

1. **Start**: The process begins.
2. **Enter Credentials**: User inputs username and password.
3. **Send Authentication Request**: Credentials are sent to the Authentication Server.
4. **Verify Credentials**: Server checks credentials against the database.
5. **Authentication** Successful?
    a. Yes: Proceed to "Access Granted."
    b. No: Go back to "Enter Credentials."
6. **Access Granted**: User gains access to the system.
7. **End**: The process concludes.

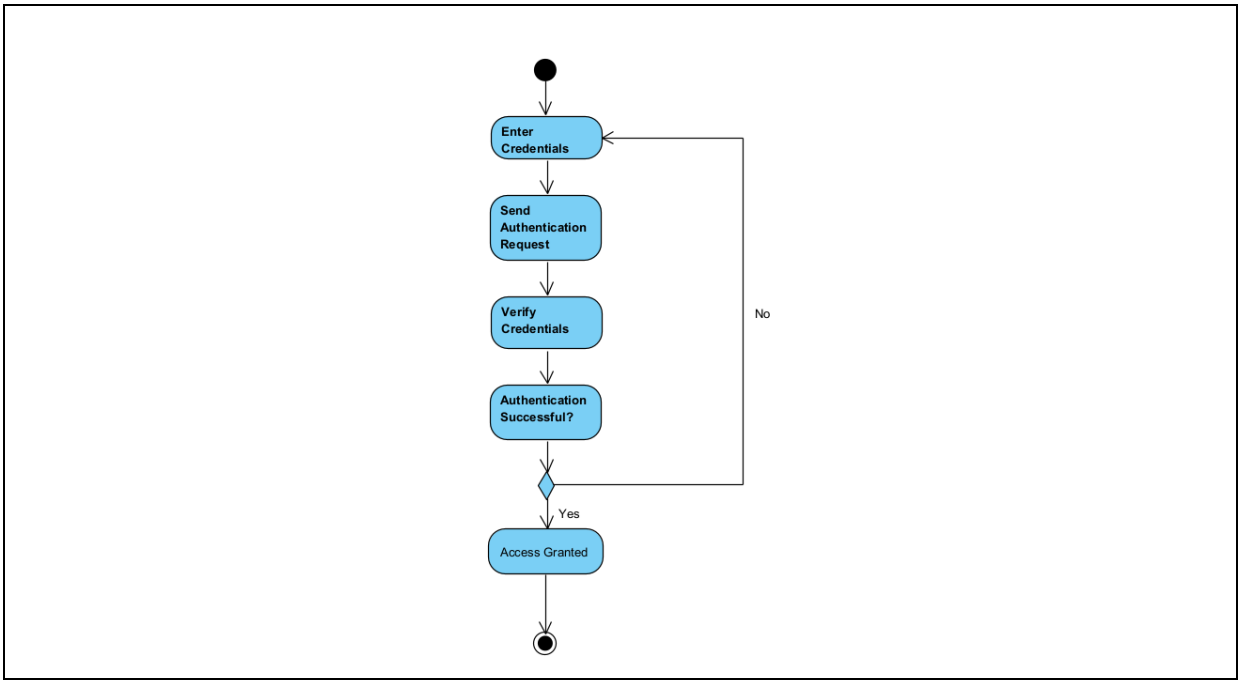


Figure 6: User Authentication Process

6.3.1.2 VPN Detection Process

The VPN Detection Process is a vital component of the VPN Spyglass system since it ensures network security. This procedure entails recording network traffic, analyzing packets using deep packet inspection, and detecting probable VPN activity.

This technique is critical because it allows administrators to proactively detect unauthorized VPN usage, adding an extra layer of protection to avoid potential data breaches and unauthorized network access. It provides administrators with the ability to successfully enforce network regulations and protect the integrity of their network architecture.

The process of VPN Detection can be expressed as follows:

1. **Start**: The process begins.
2. **Capture Network** Traffic: System captures packets from the network.
3. **Analyze Packets**: Deep Packet Inspection is performed.
4. **Detect VPN Traffic**: System identifies potential VPN traffic.
5. **Categorize VPN Traffic**: System classifies the VPN traffic by protocol or type.
6. **Display Results**: Results are shown on the dashboard.
7. **End**: The process concludes.

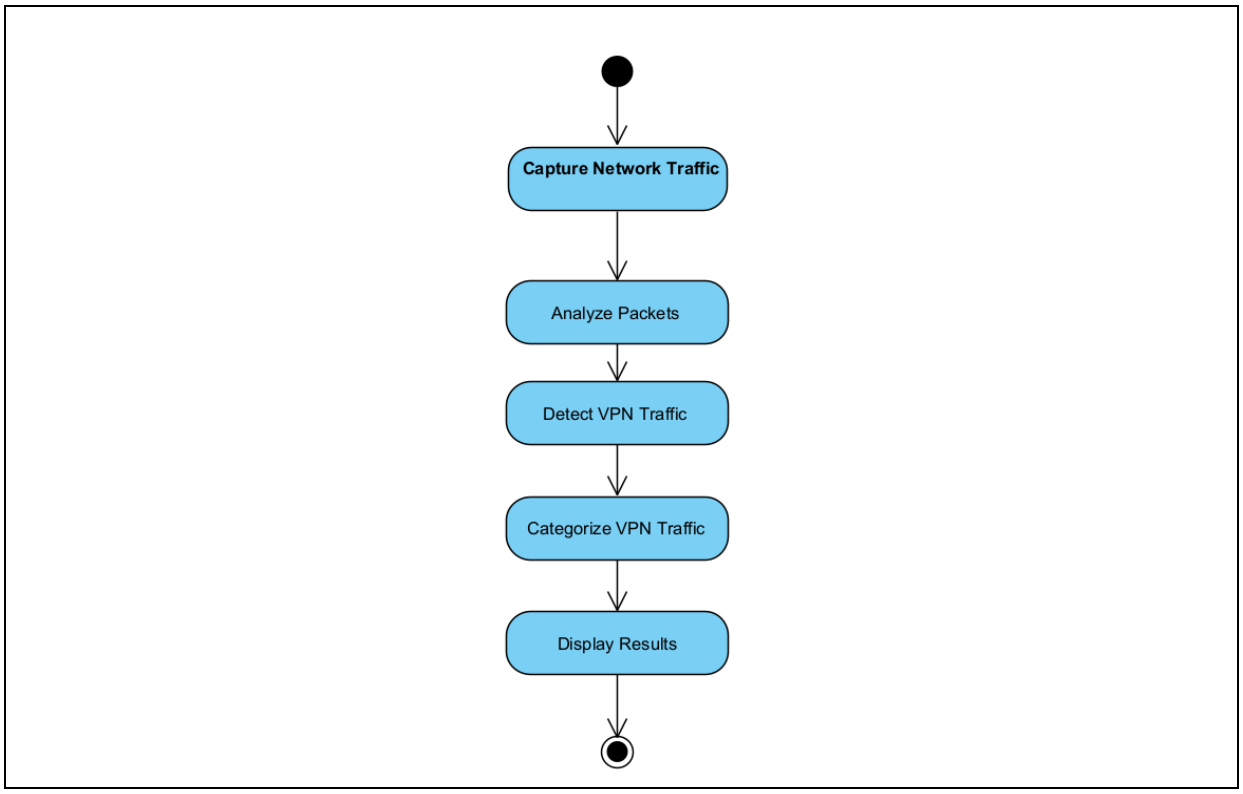


Figure 7: VPN Detection Process

6.3.1.3 VPN Blocking Process

The VPN Blocking Process is an important security technique in network management that includes detecting and restricting unauthorized VPN (Virtual Private Network) traffic within a network. This technique is critical because it allows network administrators to maintain control over their network resources, prevent possible security breaches, and enforce corporate rules by selectively limiting VPN connections that may violate security standards or access limits.

The process of VPN blocking can be expressed as follows:

1. **Start**: The process begins.
2. **Review VPN Traffic**: Administrator reviews identified VPN traffic.
3. **Want to block the traffic**?
    - Yes: Continue to "Select Traffic to Block"
    - No: End (The process concludes)
4. **Select Traffic** to Block: Administrator selects unauthorized VPN traffic.
5. **Send Block Request**: Request is sent to VPN Blocking Module.
6. **Update Rules**: VPN Blocking Module activates block rules.

7. **End**: The process concludes

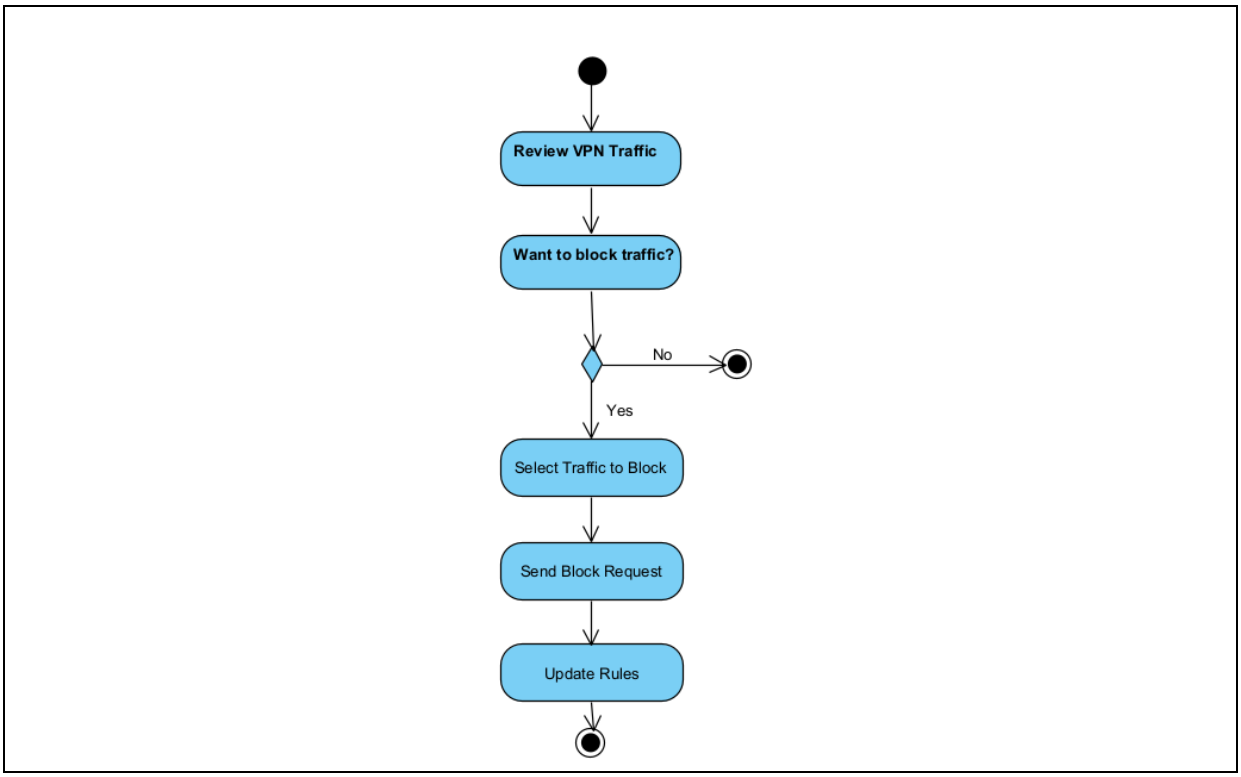


Figure 8: VPN Blocking Process

6.3.2 Data Flow Diagram

The VPN SpyGlass project's Data Flow Diagrams (DFDs) provide a detailed visual representation of data flow and processing within the system. A complex program called VPN SpyGlass is designed to monitor, investigate, and regulate VPN activities inside a network architecture. These DFDs are intended to give an organized and transparent view of the system's operation by breaking down its complex workings into simpler components.

The DFDs are organized into layers that provide varying levels of detail.

- **Level 0 DFD:** The context diagram (Level 0 DFD) is the most abstract level. It depicts how the VPN SpyGlass system interacts with external parties such as network administrators and network infrastructure as a unified procedure. Understanding the system's boundaries and external interfaces necessitates knowledge of this figure.
- **Level 1 DFD**: This level breaks down the major process from the Context Diagram into its various subprocesses. It delves into the key aspects of VPN SpyGlass, including data analysis, report writing, network traffic monitoring, and VPN traffic control. This level provides a more in-depth analysis of the system's network data processing and storage interactions.
- **Level 2 DFD**: Each Level 1 subprocess is broken down to reveal its underlying workings. This level provides additional information about the system's specific functions, such as rule enforcement, packet capture, signature matching, and report visualization.

Level 0 (Context Diagram)

Description

- **System**: VPN SpyGlass.
- **External Entities**: Web Dashboard, and Firewall
- **Data Flows:** User commands, Traffic data, Update Firewall Rules, and Simulated Traffic.

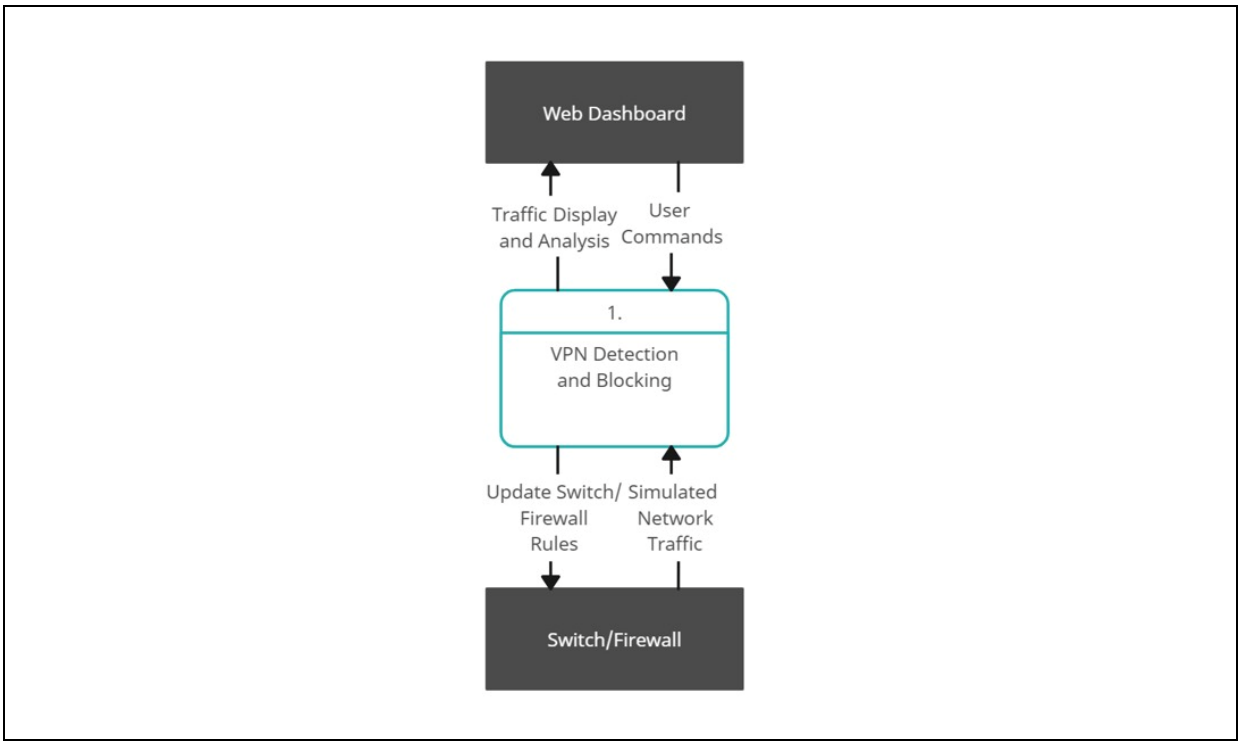


Figure 9: Level 0 (Context Diagram)

Level 1 DFD

Description

- **Processes**: User Interface, Traffic Monitoring, VPN Categorization, and VPN Blocking.
- **Data Stores**: Database.
- **Data Flows**: Enter Credentials, Verify Credentials, Access Granted, Block Traffic, Display Traffic, Categorized VPNs, Analysis, Simulated Traffic, and Command to Update Rules.

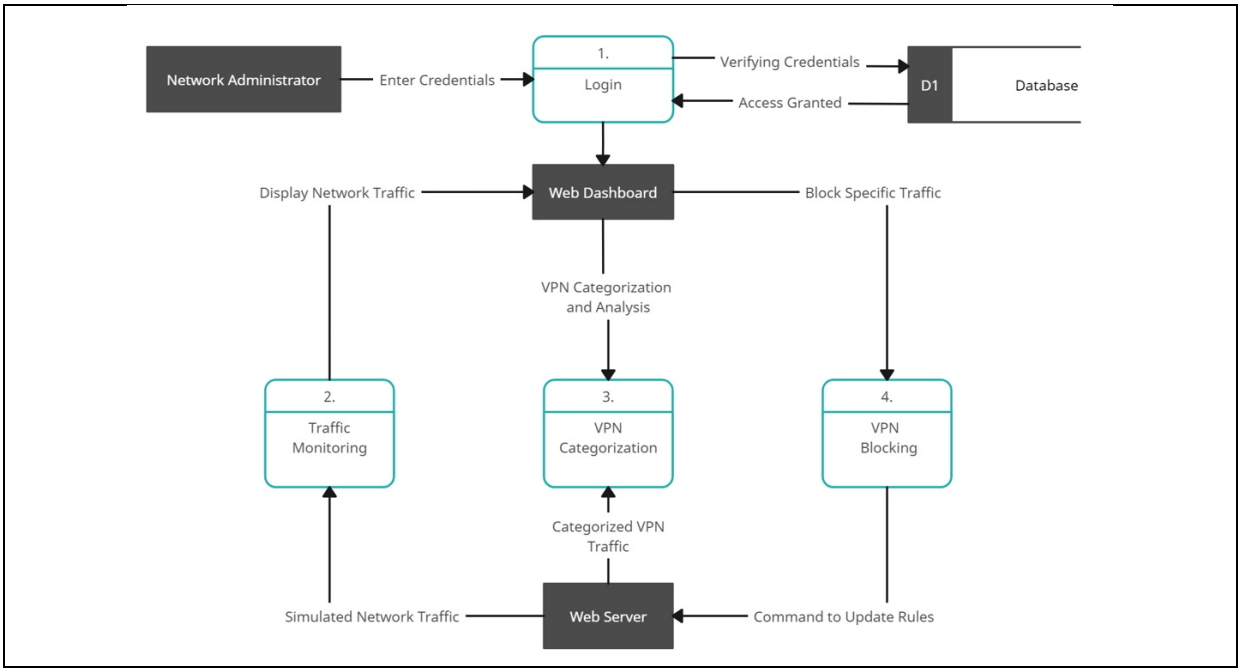


Figure 10: Level 1 DFD

Level 2 DFD

Packet Identification

Description

- **Processes**: Receive Packet Data, Signature Pattern Matching, VPN Protocol Identification, Generate Match Report.
- **Data Stores**: Signature Patterns, Protocol Identification Rules, Match Reports.
- **Data Flows**: Packet data, signature match queries, protocol identification data, and report generation details.

Diagram

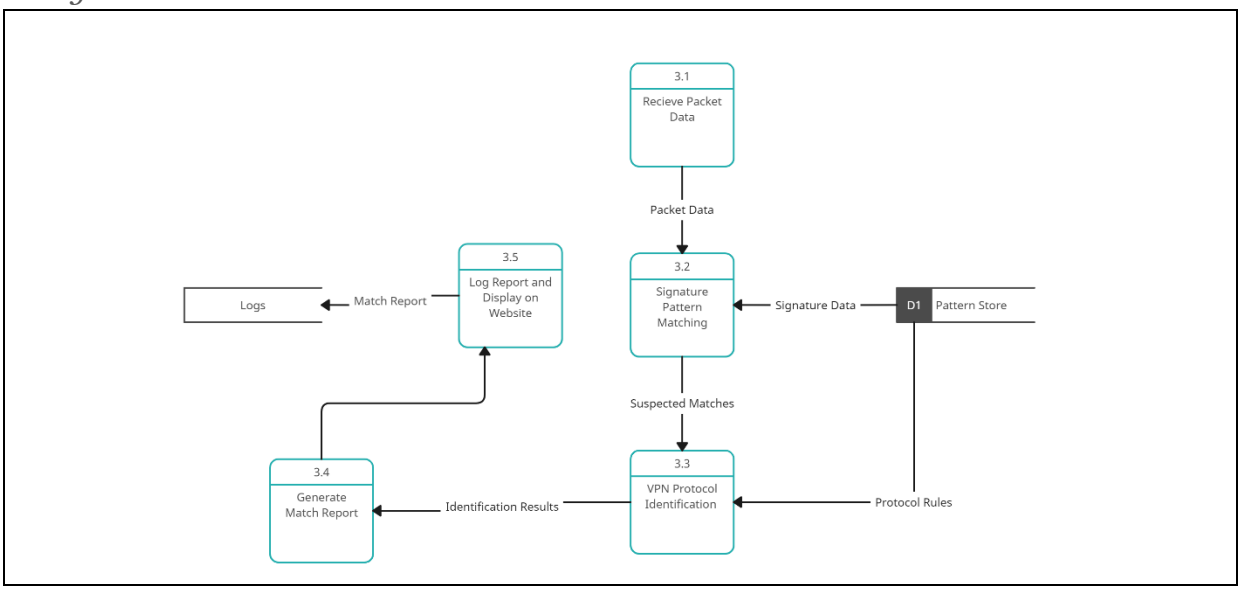


Figure 11: Packet Identification Level 2

Explanation

- **Receive Packet Data**: This process involves receiving the raw packet data from the previous stage.
- **Signature Pattern Matching**: Compares packet data against a database of signature patterns.
- **Patterns Store**: A data store containing various signature patterns used for matching against packet data.
- **VPN Protocol Identification**: Determines the specific VPN protocol used if a VPN signature is matched.
- **Protocol Rules**: A set of rules or criteria used to identify different VPN protocols.
- **Generate Match Report**: Creates a detailed report based on the signature matching and protocol identification results.
- **Match Reports**: A repository where these detailed match reports are stored.
- **Log Analysis**: The process where these reports are analyzed, and actions are determined.

VPN Blocking

Description

- **Processes**: Receive Block Request, Validate VPN, Update Blocking Rules, and Apply Blocking.
- **Data Stores**: VPN Identity Database
- **Data Flows**: Block requests, rule updates, and applied block.

Diagram

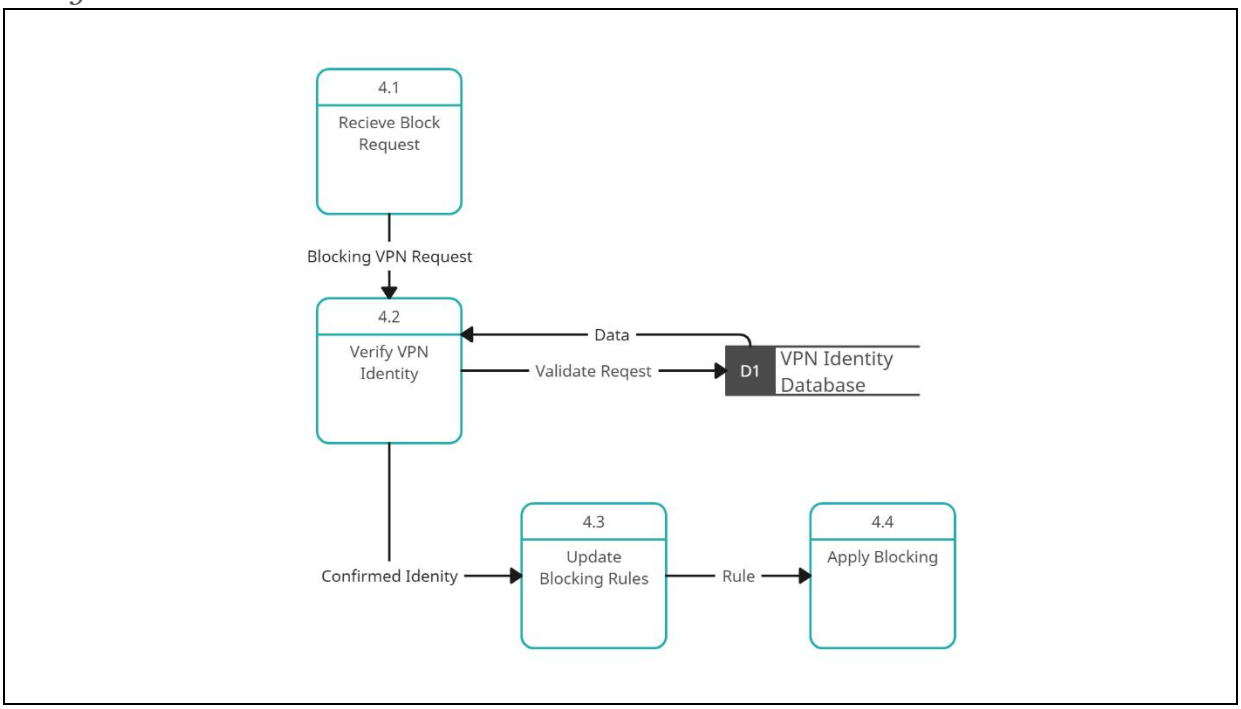


Figure 12: Packet Blocking Level 2

Explanation

- **Receive Block Request**: This process starts with receiving a request to block a specific VPN connection.

- **Validate VPN Identity**: Validates the identity of the VPN connection against a database to ensure accuracy.
- **VPN Identity Database**: A data store containing details about known VPN connections and identifiers.
- **Update Blocking Rules**: Updates the system's blocking rules with the new VPN to be blocked.
- **Blocking Rules**: A database of rules used by the system to block VPN traffic.
- **Apply Blocking**: The process where the updated rules are applied to the network, effectively blocking the VPN traffic.
- **VPN Management**: The higher-level process that oversees and manages the VPN blocking actions.

6.3.3 State Transition Diagram

For the purpose of the VPN SpyGlass project, a Behavioral State Machine Diagram must be created. This entails outlining all the states and transitions that are pertinent to the system's functioning, with special attention paid to how the system responds to user interactions and network activity.

Sate Working

What each state does is as follows:

1. **Idle State**: Waiting for a scheduling trigger or user action to initiate monitoring.
2. **Monitoring State**: obtaining network packets and examining them to look for VPN signatures.
3. **Analysis State**: VPN traffic is identified using deep packet inspection.
4. **VPN Detected State**: VPN communication is recognized and recorded.
5. **Action Pending State**: User-selected handling of observed VPN traffic is still pending.
6. **Block VPN State**: Uses user-defined criteria to prevent certain VPN traffic from passing through.

Transitions

The state transition can be given as:

1. **Idle State**: The system is running but not actively monitoring network traffic.
   - **Transition**: Start monitoring.
   - **Event**: User command or scheduled task.
   - **Action**: Begin capturing network packets.
2. **Monitoring State**: The system is actively monitoring network traffic.
   - **Transition to Analysis**: Packet capture.
   - **Event**: Network packet received.
   - **Action**: Analyze packet for VPN signatures.
3. **Analysis State**: The system is analyzing a network packet.
   - **Transition to VPN Detected**: VPN signature found.
   - **Event**: Matching VPN protocol patterns.
   - **Action**: Log and categorize VPN traffic.
4. **Transition to Monitoring**: Packet analyzed.
   - **Event**: Analysis complete.

- **Action**: Resume monitoring.

5. **VPN Detected State**: A VPN packet has been identified.
   - **Transition to Action Pending**: User or system decision.
   - **Event**: Analysis report review.
   - **Action**: Notify user and await action.
6. **Action Pending State**: Awaiting user decision on detected VPN traffic.
   - **Transition to Block VPN**: User command.
   - **Event**: User selects to block VPN traffic.
   - **Action**: Apply OpenFlow rules to block traffic.
   - **Transition to Monitoring**: User ignores or accepts VPN.
   - **Event**: User decision.
   - **Action**: Log decision and continue monitoring.
7. **Block VPN State**: The system blocks identified VPN traffic.
   - **Transition to Monitoring**: Blocking rules applied.
   - **Event**: VPN traffic blocked.
   - **Action**: Resume normal monitoring.

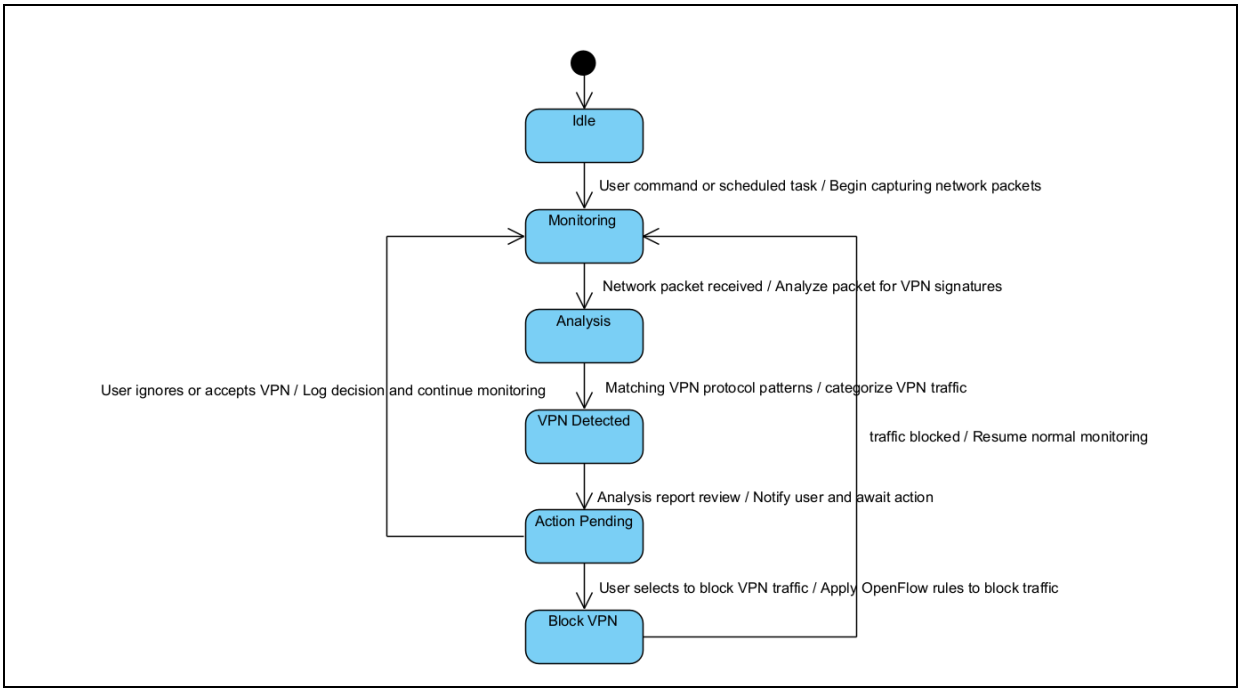


Figure 13: State Transition Diagram

6.3.4 Sequence Diagram

The sequence diagram given here is intended to provide a clear and systematic visual representation of a user's interactions with the VPN SpyGlass system. The graphic depicts the communication flows that occur when a user interacts with the system's online dashboard and associated backend services for authentication, traffic analysis, and VPN blocking.

Diagram

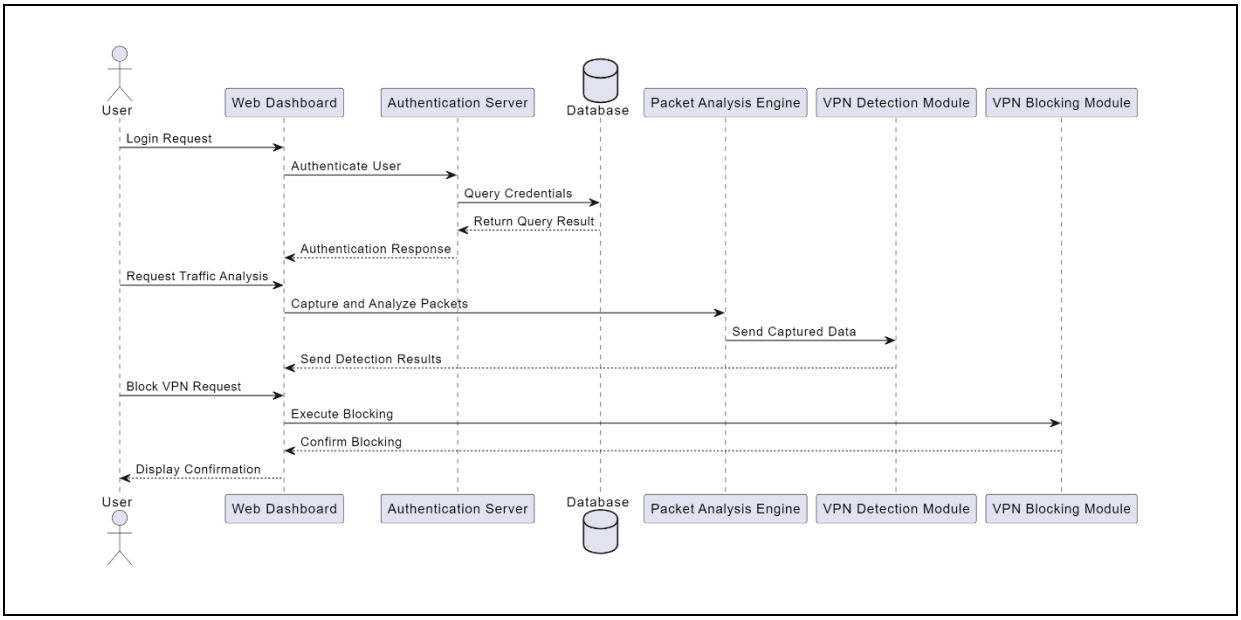


Figure 14: Sequence Diagram

Explanation

1. **Login Procedure for Users:**
   - A Login Request to the online dashboard is initiated by the user.
   - This request is forwarded to the Authentication Server via the web dashboard.
   - With the user's credentials, the Authentication Server searches the Database.
   - The query result is returned to the Authentication Server by the Database.
   - The Authentication Server returns to the web dashboard an Authentication Response, which presumably indicates whether the login attempt was successful.
   - The conclusion is displayed to the user via the online dashboard.
2. **Request for Traffic Analysis**
   - After logging in successfully, the user requests Traffic Analysis using the online dashboard.
   - The online dashboard starts the Capture and Analyze Packets process using the Packet Analysis Engine.
   - The Packet Analysis Engine returns the Detection Results to the online dashboard after the analysis is complete.
   - The online dashboard shows the user the results of the analysis.
3. **VPN Blocking Request**
   - If a user detects unwanted VPN traffic, they may utilize the online dashboard to submit a VPN Blocking Request.
   - The VPN Blocking Module is instructed to execute blocking by the online dashboard.
   - The VPN Blocking Module then validates the activity by returning to the web dashboard with a Confirm Blocking message.
   - Finally, the web dashboard displays a Display Confirmation message to the user, indicating that the VPN has been successfully blocked.

6.4 Data Design

The Data Design element of the VPN SpyGlass project is a crucial component that methodically explains how the system's informational domain is converted into structured and useful data structures. This part examines the subtleties of the system's data collecting, processing, and organization with a focus on the detection and management of VPN traffic to guarantee the effective handling of network traffic data. Extensive descriptions of the relationships and exchanges between the data structures that hold user credentials, VPN signatures, packet information, and system settings are crucial elements. The Entity-Relationship Diagrams (ERDs) and other tables and schemas are used by the VPN SpyGlass system to illustrate the structure of databases and data storage methods, emphasizing their design. By enhancing the system's capacity for network monitoring and security management and enabling efficient data processing and retrieval, this approach ensures a robust and trustworthy tool for network managers.

As part of the VPN SpyGlass project's data architecture, the information domain is arranged into certain data structures to facilitate efficient processing, storing, and retrieval. Network traffic data, user management data, and VPN categorization rules are the three primary data types that the system handles.

6.4.1 Data Structures and Storage

- Network Traffic Data: This includes intercepted network packets. Each packet is represented by a data structure that contains the source and destination IP addresses, port numbers, timestamps, packet lengths, and protocol types. This data is saved for further analysis after being assessed in real time for VPN detection.
- User Management Data: Contains preferences and administrator login information for the web dashboard. includes username, encrypted password, role information, and dashboard settings unique to each user.
- VPN Signature Database: An index of popular VPN signatures and patterns used in deep packet inspection. This database contains information on VPN protocol types, signature patterns, and associated metadata.
- VPN Traffic Logs: Identified and preserved records of VPN traffic that contain details on the kind of VPN, the source and destination IP addresses, the time of detection, and the action performed (blocked, authorized, etc.).
- System Configuration Data: Configuration parameters for the system, including network interfaces to monitor, alert thresholds, and other operational information.

6.4.2 ERD Diagram

The Entity-Relationship Diagram (ERD) provides an extensive visual representation of the system architecture and data relationships. This figure makes it easy to understand how different parts inside a network traffic analysis tool interact and link, such as packets, VPN signatures, user logs, system configurations, and VPN records. The ERD lists the attributes of each object and their relationships with one another, giving a clear and comprehensive overview of the system's data architecture. It is an essential tool for the project's development and maintenance phases, ensuring that all aspects of data management are rationally and successfully integrated to support the VPN SpyGlass system's primary goal of effectively monitoring, identifying, and controlling VPN traffic inside a network environment.

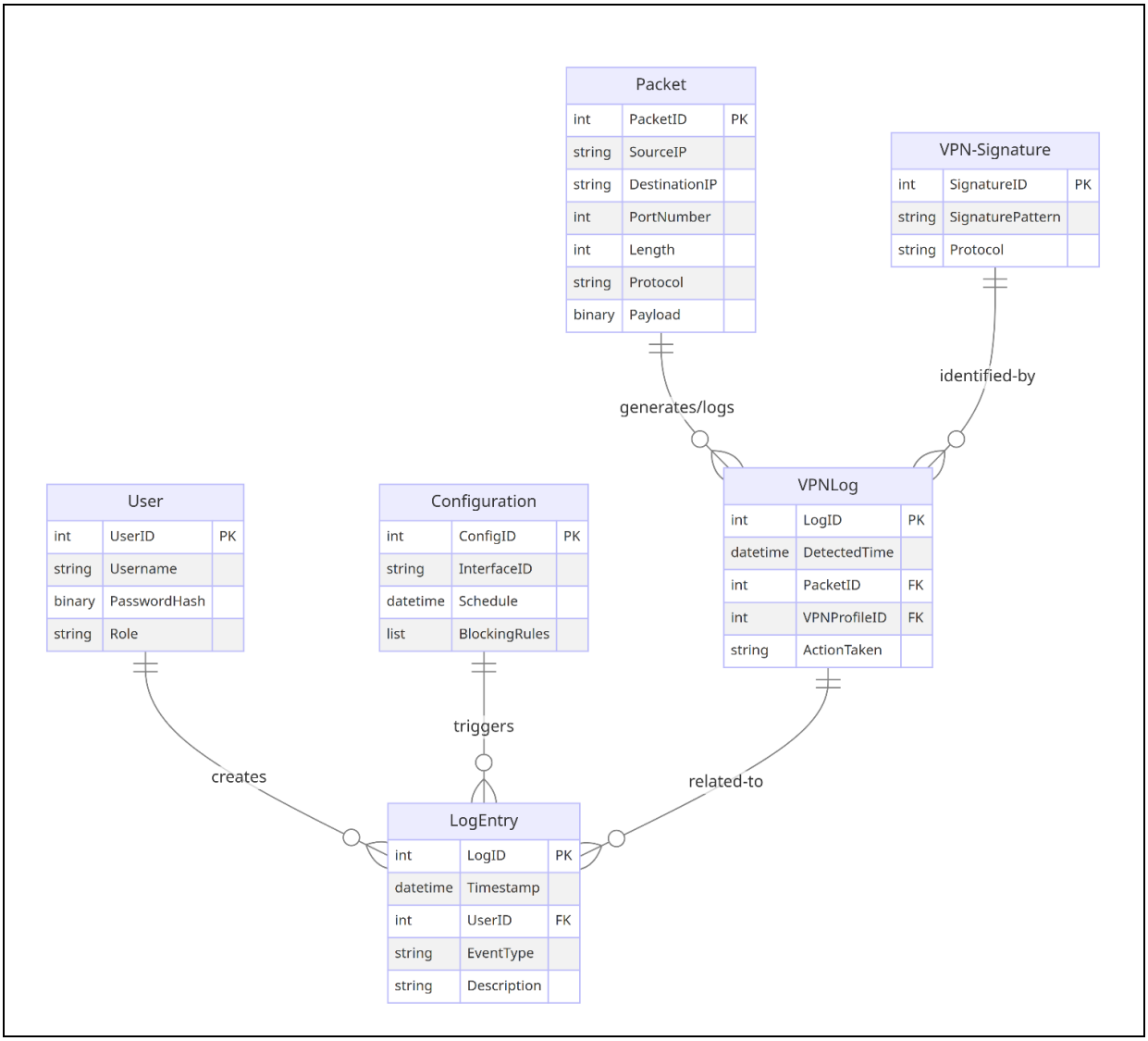


Figure 15: ERD Diagram

6.4.3 XML Schema

In terms of network security and tracking, the VPN SpyGlass project offers a cutting-edge tool for controlling and examining VPN traffic. To manage the extensive and varied data involved in this process, we have built an XML schema to arrange the key data pieces in a hierarchical and structured manner. This schema needs to represent all the components of the VPN SpyGlass system, including user profiles, log entries, and configuration settings. The XML schema provides our data with a clear and well-defined structure, ensuring consistency, integrity, and ease of change and retrieval. It serves as the basis for data processing and archiving, enabling precise and effective VPN traffic analysis.

The XML schema used in the VPN SpyGlass project is described in the section that follows. It describes the format and structure of the main data pieces that power our system's operation.

Description of the Schema

- Configurations: Shows the settings for the system configuration. The network interface ID, the monitoring schedule, and a set of traffic analysis and blocking rules are all included.

- User: Specifies the structure of the user profile. This contains the user's identity, username, password hash (which is saved as a base64 encoded binary for security purposes), and role (administrator, analyst, etc.).

- LogEntry**:** Stores system-generated log entries. Every entry contains an ID, a timestamp, the kind of event, the user ID that started it, and a description.

- VPNLog: Dedicated to recording VPN detection occurrences. It contains the VPN profile ID that matched, the log ID, the time of detection, the related packet ID, and the reaction (block, allow, alert) to the detection.

XML Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

    <!-- Configuration Settings -->
    <xs:element name="Configurations">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="InterfaceID" type="xs:string"/>
                <xs:element name="Schedule" type="xs:dateTime"/>
                <xs:element name="Rules" type="xs:string"
maxOccurs="unbounded"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>

    <!-- User Profile -->
    <xs:element name="User">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="UserID" type="xs:string"/>
                <xs:element name="Username" type="xs:string"/>
                <xs:element name="PasswordHash" type="xs:base64Binary"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>

    <!-- Log Entry -->
    <xs:element name="LogEntry">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="EntryID" type="xs:int"/>
                <xs:element name="Timestamp" type="xs:dateTime"/>
                <xs:element name="UserID" type="xs:string"/>
                <xs:element name="EventType" type="xs:string"/>
                <xs:element name="Description" type="xs:string"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
```

```
    <!-- VPN Log -->
    <xs:element name="VPNLog">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="LogID" type="xs:int"/>
                <xs:element name="DetectedTime" type="xs:dateTime"/>
                <xs:element name="PacketID" type="xs:int"/>
                <xs:element name="VPNProfileID" type="xs:int"/>
                <xs:element name="ActionTaken" type="xs:string"/>
            </xs:sequence>
        </xs:complexType>
    </xs:element>

</xs:schema>
```

6.4.4 Data Dictionary

1. Network Packet
   - **Attributes**: Source IP (String), Destination IP (String), Port Number (Integer), Timestamp (DateTime), Packet Length (Integer), Protocol Type (String).
   - **Description**: Represents a network packet captured for analysis.
2. User
   - **Attributes**: Username (String), Password (Encrypted String), Role (String), Settings (JSON/Object).
   - **Methods**: Authenticate(), UpdateSettings().
   - **Description**: User entity for access control to the dashboard.
3. VPN Signature
   - **Attributes**: Signature ID (Integer), Protocol Type (String), Pattern (String), Metadata (JSON/Object).
   - **Description**: Represents a VPN traffic pattern for detection purposes.
4. VPN Log
   - **Attributes**: Log ID (Integer), Detection Time (DateTime), VPN Type (String), Source IP (String), Destination IP (String), Action (String).
   - **Description**: Log entry for each VPN traffic detection.

5. System Configuration
   - **Attributes**: Config ID (Integer), Network Interfaces (Array of Strings), Alert Thresholds (JSON/Object), Operational Parameters (JSON/Object).
   - **Description**: Stores system operational settings.

6.5 User Interface Design

Now let's design the VPN SpyGlass user interface, emphasizing usability, social media engagement, and system input. The design would aim to provide a smooth and user-friendly interface for network traffic management and VPN traffic monitoring.

6.5.1 Functionality from User's Perspective

Network administrators may easily monitor, analyze, and control VPN traffic on their network thanks to the user-friendly interface of VPN SpyGlass. Important features from the viewpoint of the user consist of:

6.5.1.1 Login & Authentication Screen

Functionality: To gain secure access, users must first log in using their credentials.
Feedback: Users receive authentication success or failure messages.

6.5.1.2 Dashboard/Home Screen

Functionality: The main interface displays real-time network traffic as well as VPN operation indicators. Users can start and stop monitoring as well as view active connections.
Feedback: Live network traffic updates, alarms on detected VPN traffic, and system status

6.5.1.3 VPN detection and Analysis Section

Functionality: Provide comprehensive details, such as IP addresses, protocols, and actions done, regarding VPN traffic that has been discovered.
Feedback: Historical statistics, classification information, and a list of VPN connections that have been found.

6.5.1.4 VPN Management Section

Functionality: Choices to permit or prohibit particular VPN traffic. Rules and exceptions can be managed by users.
Feedback: Verification of the rules used, connections successfully blocked, and any mistakes.

6.5.1.5 Settings & Configuration

Functionality: Manage user accounts, set up notification preferences, and alter the tool's settings.
Feedback: System warnings, configuration status, and confirmation of changes.

6.5.1.6 Reports & Logs

Functionality: Access and produce reports on system performance, user activity, and VPN traffic.
Feedback: Analytic charts, historical logs, and generated reports.

6.6 Screen Objects and Actions

6.6.1 Navigation Bar

Objects: Account Settings, Historical Data, Alerts, Traffic Analysis, and Dashboard menu items.
Actions: The user can access the corresponding sections by clicking on these items.

6.6.2 Real-time Traffic Graph (Dashboard)

Objects: VPN traffic highlight, line chart or heat map.

Actions: Specific traffic information is displayed when you hover over an element. The VPN Analysis screen is displayed to the user when clicking on a VPN highlight.

6.6.3 VPN Traffic List (Analysis Screen)

Objects: List of VPN connections, 'Details' and 'Block' buttons.
Actions: Selecting 'Details' displays a modal window containing further details. Verifying VPN blocking is prompted when you click "Block."

6.6.4 Alerts List

Objects: A list of alerts with a synopsis for each.
Actions: Tapping an alert enlarges it to provide further information or directs the user to the relevant traffic analysis.

6.6.5 Historical Data Charts

Objects: A range of filters and charts.
Actions: By using filters, the charts are updated to show the necessary data.

6.6.6 Account Management Forms

Objects: Notification settings, password, and user information input boxes.
Actions: The user's account settings are updated upon completing and submitting the form.

6.6.7 Breakdown into smaller parts

- **Buttons**: Save settings, create reports, block/unblock VPN, start/stop monitoring. These can be clicked to carry out the corresponding activities.
- **Fields**: Input fields for settings parameters, list filters, and login. Users have the option to choose or enter the required data.
- **Tabs**: Navigate through the various areas, such as Reports, Dashboard, VPN Management, and Settings.
- **Alerts and Notifications**: Real-time feedback on system status, VPN detections, or faults through pop-ups or dedicated alert sections.
- **Graphs and Charts**: Data is visually represented to make traffic patterns and VPN activity easier to comprehend.
- **Tables/List Views**: Provide choices for sorting and other actions, as well as full information on identified VPNs, logs, and user activity.

Chapter 7

SYSTEM TESTING

7.1 Unit Testing

Unit testing was performed using PyTest, a popular testing library for Python, and module for performing network traffic classification. All the unit tests passed with the latest version of the code.

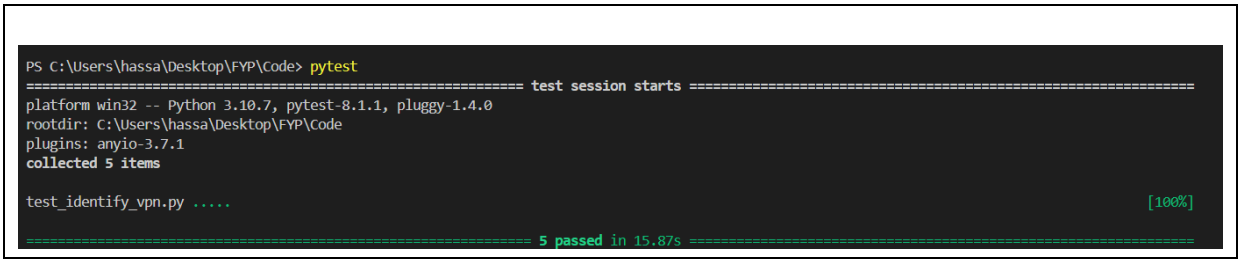


Figure 16: Unit Testing

7.2 Integration Testing

Integration testing was performed by making all connections between the live traffic capturing, and displaying classified traffic on the endpoints, which help with monitoring the traffic.

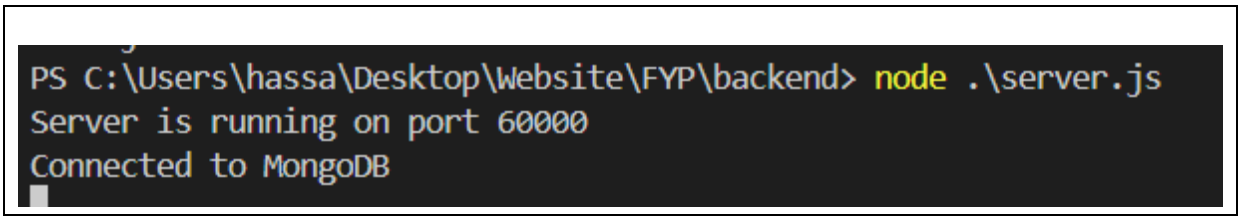


Figure 17: Integration Testing

7.3 User Acceptance Testing

A set of user acceptance test cases was prepared to validate the basic requirement of the product. They are mentioned below:

Test Case ID	Description	Precondition	Test Steps	Expected Result	Status
UAT_AUTH_001	Verify user access by logging in with valid credentials	Registered user credentials	Login with valid username and password	Successful login redirects to the dashboard	Passed
UAT_DATA_001	Ensure accurate data retrieval and display on the dashboard	Backend connected to database	View displayed data on the dashboard	Real-time data accurately presented	Passed
UAT_VISUAL_001	Confirm proper data analysis and visualization	Implemented data analysis algorithms	Review graphical representations on dashboard	Clear and accurate visualization of traffic patterns	Passed
UAT_VPN_001	Validate correct categorization of VPN traffic	VPN traffic simulation or available	Analyze categorized VPN traffic	Accurate VPN traffic labeling	Passed

UAT_RULES_001	Verify the functionality of rules for categorizing VPN traffic	Defined rules for VPN traffic categorization	Send packets representing different VPN protocols, port numbers, and IP addresses. Analyze the categorized traffic based on predefined rules.	Each packet is accurately categorized according to the defined rules, with proper identification of VPN usage	Passed
UAT_BLOCKING_001	Ensure seamless transition from VPN traffic blocking to normal flow	Implemented VPN traffic blocking	Apply and remove VPN traffic block	Smooth transition to normal traffic upon block removal	Passed

Table 8: User Acceptance Testing

Chapter 8

DEPLOYMENT AND SYSTEM INTEGRATION

All the services are deployed on the Vercel and MongoDB Atlas which are clouds providing computing and storage capabilities respectively.

The items system specifications are as:

- **Dashboard Application:** Build files of React application on Vercel.
- **Classification Model:** A Python based Script running on admin machine.
- **Backend:** Nodejs to Fetch Data on Web service
- **Database:** MongoDB Atlas cloud database

The Backend running on the server end utilizes the following:

- Linux Ubuntu 22.04
- Nodejs v20.12.2
- Wireshark
- PfSense
- Express
- Python

The system diagram is shown in section 5.1.

Chapter 9

USER INTERFACE DESIGN

9.1 Web Application

The tool is available as a web application for the end user. User can access the application from their browser by going to www.vpnspyglass.vercel.app site. It will be shown as below:

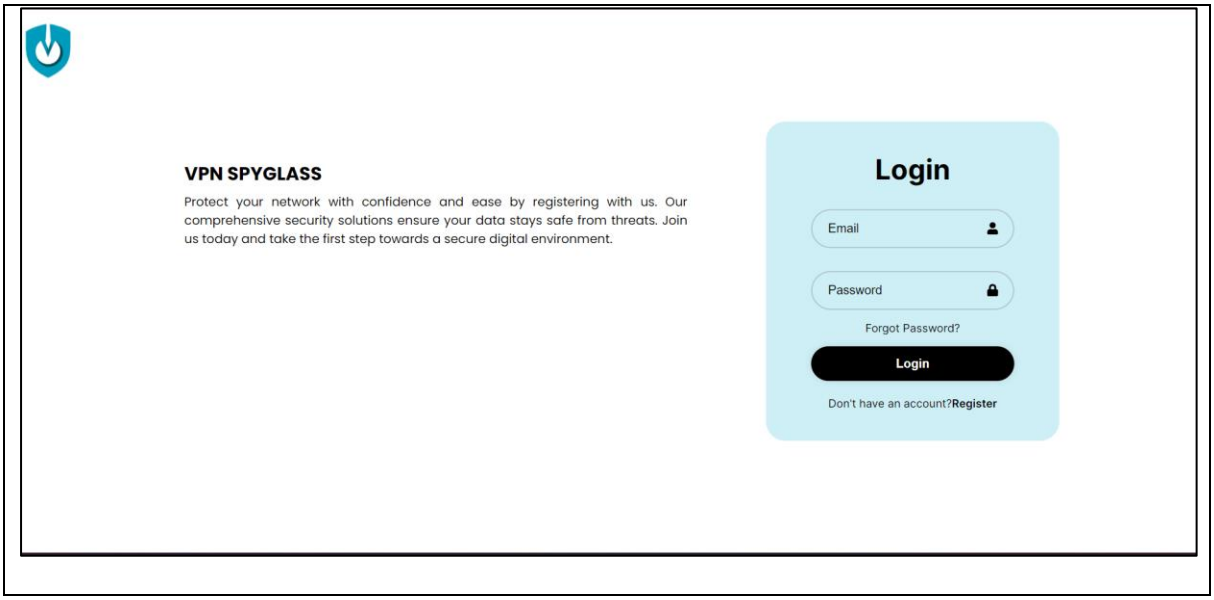


Figure 18: Login & Authentication Screen

One accessed, the user will have to log in and authenticate his credentials.

9.2 Admin Dashboard

Dashboard is available for admin only. It shows analytics about detections, blacklisted sites, whitelisted sites, managing blacklisted and whitelisted sites. It is available in light dark theme.

There are 3 views available in dashboard. Users can analyze and control the traffic by using these 3 views.

- Analytics – Stats for pervious detection by system
- Top VPNs – List of most used VPNs Depending on history.
- Live Traffic Section – Packets of live traffic classified.

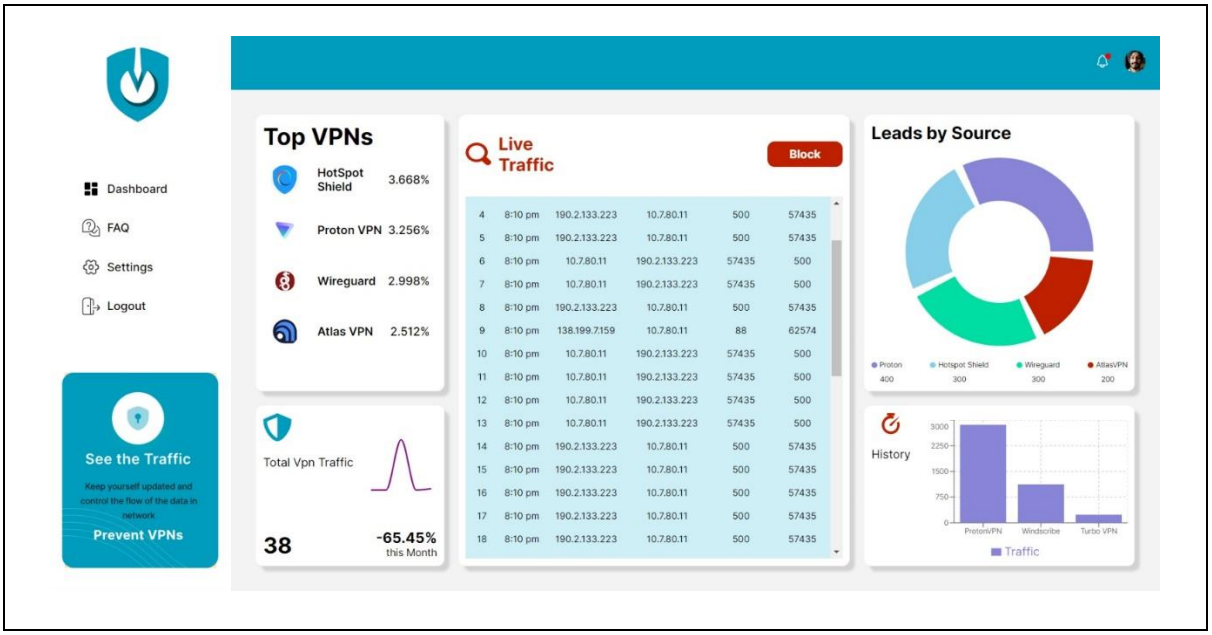


Figure 19: Dashboard/Home Screen

9.2.1 Live VPN detection Section

The live traffic analysis and detection section is the primary part of the web dashboard. The Network administrator can detect the use of VPN in real-time. It provides information like:

- Source IP and Port
- Destination IP and Port
- Packet Length
- VPN Type

23	8:11 pm	178.175.133.132	10.7.225.219	51820	51820
24	8:11 pm	178.175.133.132	10.7.225.219	51820	51820
25	8:11 pm	178.175.133.132	10.7.225.219	51820	51820
26	8:11 pm	178.175.133.132	10.7.225.219	51820	51820
27	8:11 pm	178.175.133.132	10.7.225.219	51820	51820
28	8:11 pm	10.7.225.219	178.175.133.132	51820	51820
29	8:11 pm	10.7.225.219	178.175.133.132	51820	51820
30	8:11 pm	10.7.225.219	178.175.133.132	51820	51820
31	8:11 pm	10.7.225.219	178.175.133.132	51820	51820
32	8:11 pm	178.175.133.132	10.7.225.219	51820	51820
33	8:11 pm	178.175.133.132	10.7.225.219	51820	51820
34	8:11 pm	178.175.133.132	10.7.225.219	51820	51820
35	8:11 pm	10.7.225.219	178.175.133.132	51820	51820
36	8:11 pm	178.175.133.132	10.7.225.219	51820	51820
37	8:11 pm	178.175.133.132	10.7.225.219	51820	51820
38	8:11 pm	10.7.225.219	178.175.133.132	51820	51820

Figure 20: VPN detection and Analysis Section

9.2.2 VPN Management Section

This section provides information about the most used VPNs depending on the previous history of use of the VPN SpyGlass. It helps the administrator to decide which VPN to block.

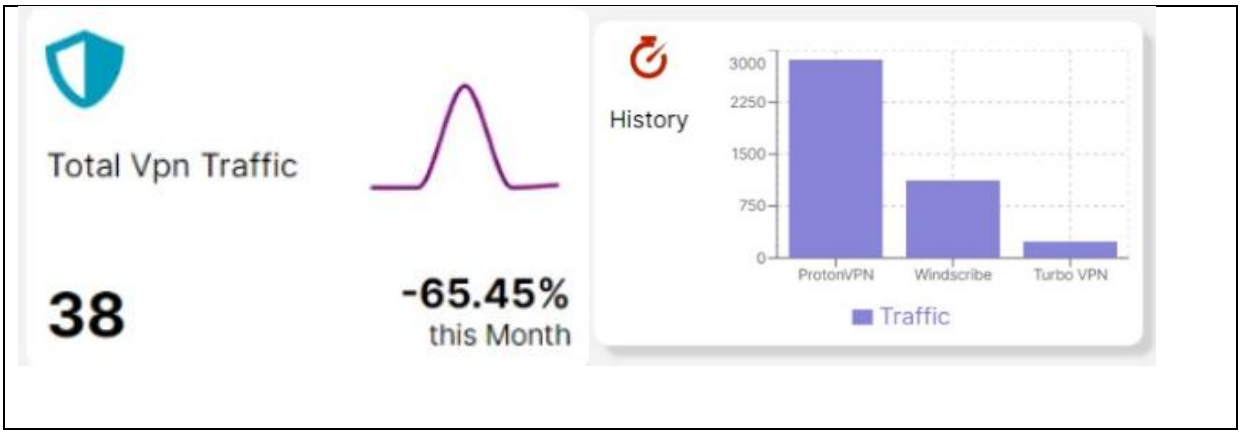


Figure 21: VPN Management Section

9.2.3 Reports & Logs:

Provides information about current logs of the VPNs being used.

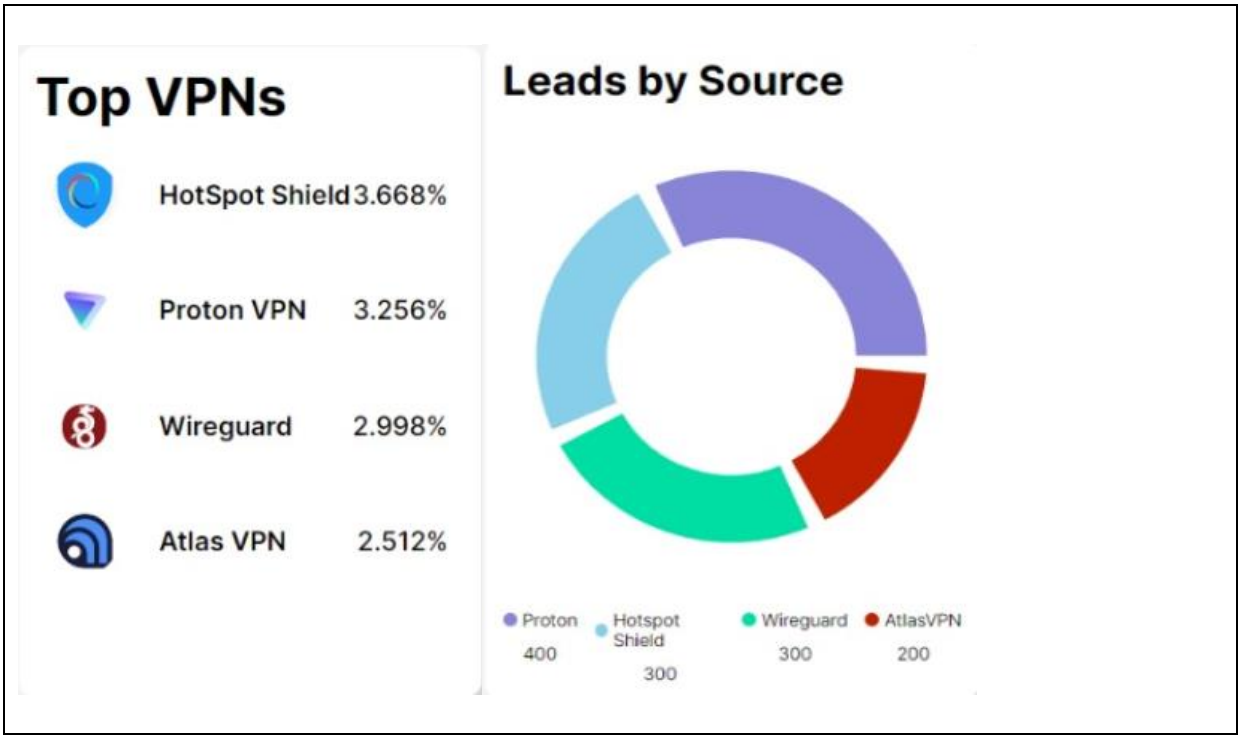


Figure 22: Reports & Logs

9.3 Blocking VPNs

9.3.1 PfSense Login

Once the user clicks the “Block button” on the web dashboard, user will be redirected to PfSense page. On the page, user enters the authentication credentials as “Username: Admin” and “Password: pfsense”. It will open the pfsense dashboard.

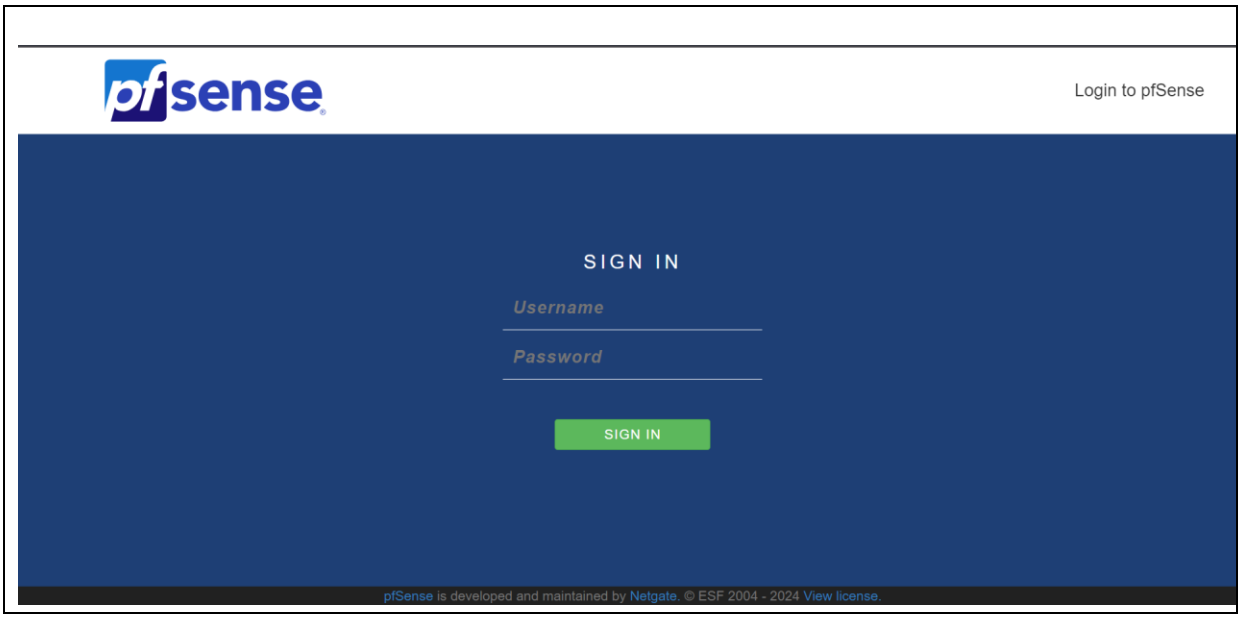


Figure 23: PfSense Login

9.3.2 Firewall Rule Enablement

On the Firewall rules page, all the rules have been set. The network administrator will only enable or disable the rule for a particular VPN.

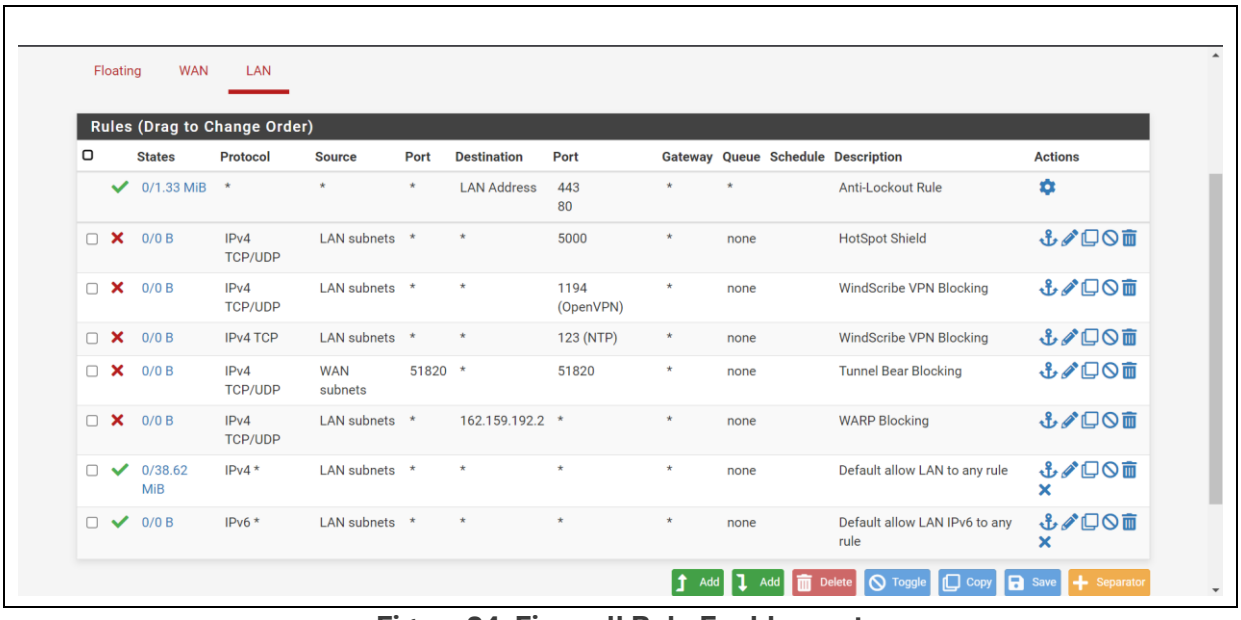


Figure 24: Firewall Rule Enablement

9.3.3 Resetting Network

Once the rules are disabled or enabled, the network administrator will have to reset the whole network on the run after which the VPN will stop working.

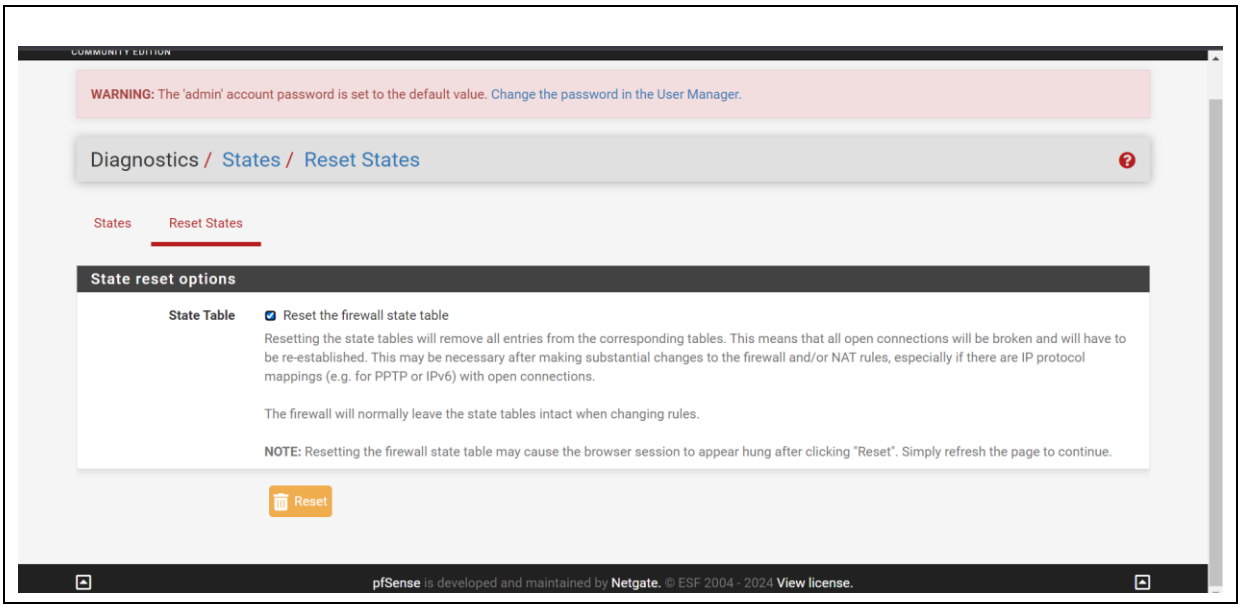


Figure 25: Resetting Network

9.4 User Account

This section will provide the user with the ability to edit his credentials and personal information.

9.4.1 User Profile Settings:

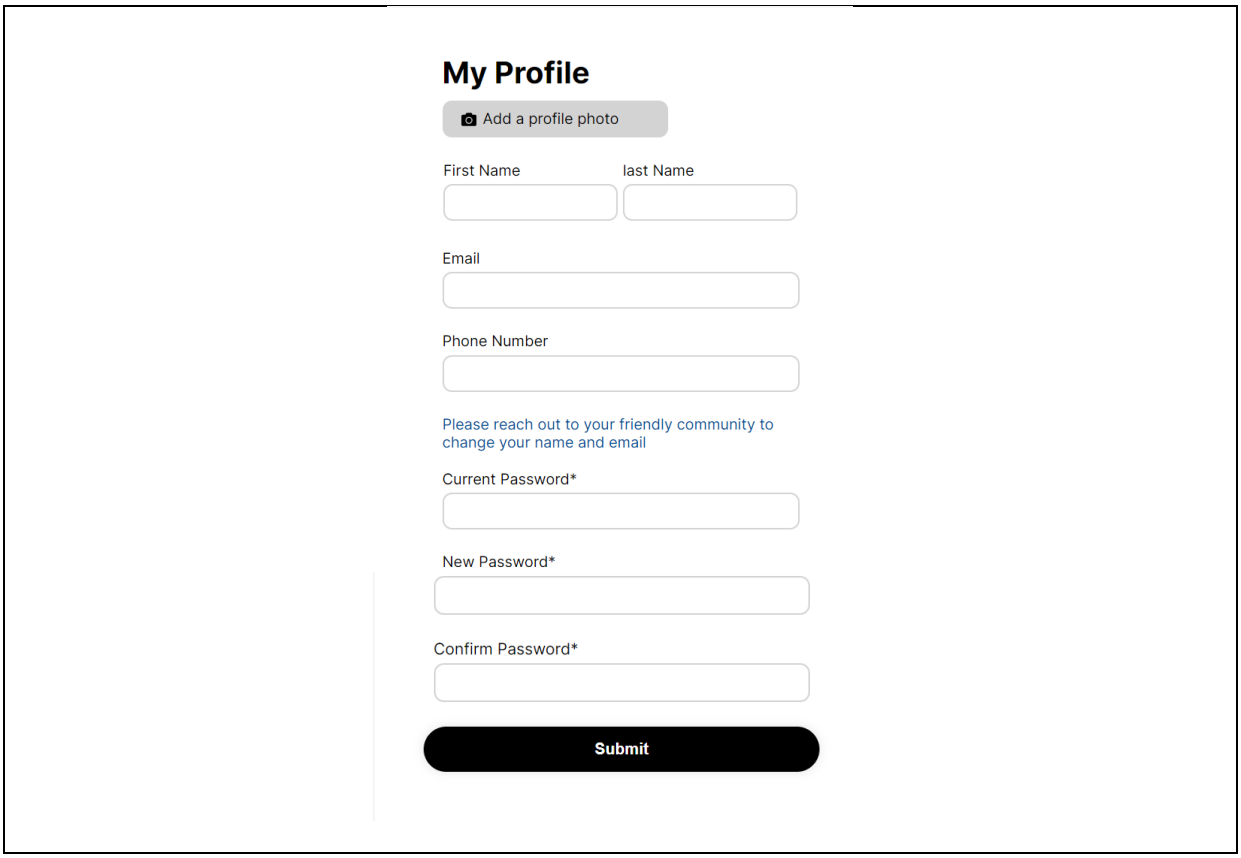


Figure 26: User Profile Settings

Chapter 10

RESULTS AND DISCUSSIONS

Encryption methods are used by VPN applications to protect data while communicating. Only packet sizes, frequencies, and recurring patterns of packets may be used for analysis because of the encrypted payload. Consequently, distinct traffic may be identified with the use of payload lengths, sizes, port numbers, and IP addresses. Extensive analysis of packet sizes (patterns, frequencies) discloses the significant operations of the applications, even while privacy and secrecy remain unaffected.

This study's thorough examination of traffic dumps produced patterns in the bytes and payloads associated with various activities. While patterns did appear, they were predictably ineffective in aiding in the reconstruction of the original data. Regulatory bodies, on the other hand, can ascertain the behaviors of VPN applications by observing constant connectivity patterns, thanks to the experimental setup provided in the previous section. Towards the conclusion of the Results and Analysis section is the comprehensive summary.

The following users were created and given descriptions to facilitate the app operations for better understanding:

- **User A:** target user, whose entire set of activities are monitored.
- **User B:** Network Administrator, who monitors the network traffic.
-

10.1 Identification of VPN Traffic

We dumped the network data from the laptop that was the target to find the activity related to the VPN apps. Without the cryptographic key, it was impossible to decipher the intercepted traffic since it was encrypted. As a result, to identify traffic patterns against the various activities, we thoroughly carried out and recorded the various activities. The target device, User A, is used for a variety of tasks.

To gain a deeper comprehension, we examined and categorized target User A's (device's) app behaviors as follows:

1. Handshake with the VPN server
2. Idle State after connection
3. Accessing Google Site
4. Searching on Google
5. YouTube Video Streaming

For this study we have looked at a variety of the VPNs [Table 2] as listed below:

1. Proton VPN
2. Hide.me VPN
3. Windscribe VPN
4. Tunnel-Bear VPN
5. Turbo VPN
6. Cloudflare WARP
7. Hotspot Shield VPN

Now we move on to individual analysis of each VPN application.

10.1.1 Proton VPN

The firm behind Proton Mail, Proton AG, is based in Switzerland and runs Proton VPN, which is a VPN service. The IPSEC protocol may be used to implement its service, which is compatible with Windows, MacOS, Linux, Android, iOS, and ChromeOS. It also provides a command-line utility for Linux. Proton VPN employs AES-256 encryption with the OpenVPN (UDP/TCP), IKEv2, and WireGuard protocols.

Now we examine and categorize target application behaviors as follows:

10.1.1.1 WireGuard Mode

One VPN protocol, called WireGuard, is a set of Guidelines that controls data encryption and transfer inside virtual private networks (VPNs). Typically, a VPN server and a client—your phone app, for instance—are involved in a WireGuard VPN. Similar to other encryption protocols, WireGuard provides an encrypted tunnel between the client and server through communication with the server. The data that travels between the WireGuard client and server, the two nodes on the network, is encrypted and converted into unintelligible code that requires the right encryption keys to decrypt.

Handshake with the VPN server

The Proton VPN Handshake activity signifies the beginning of the program on User A's device. At first, it was challenging to distinguish VPN-only traffic from packets originating from the common network. Consequently, many trace files were recorded and kept an eye on during the Proton VPN application and VPN server interaction.

The process of the above activity is as follows:

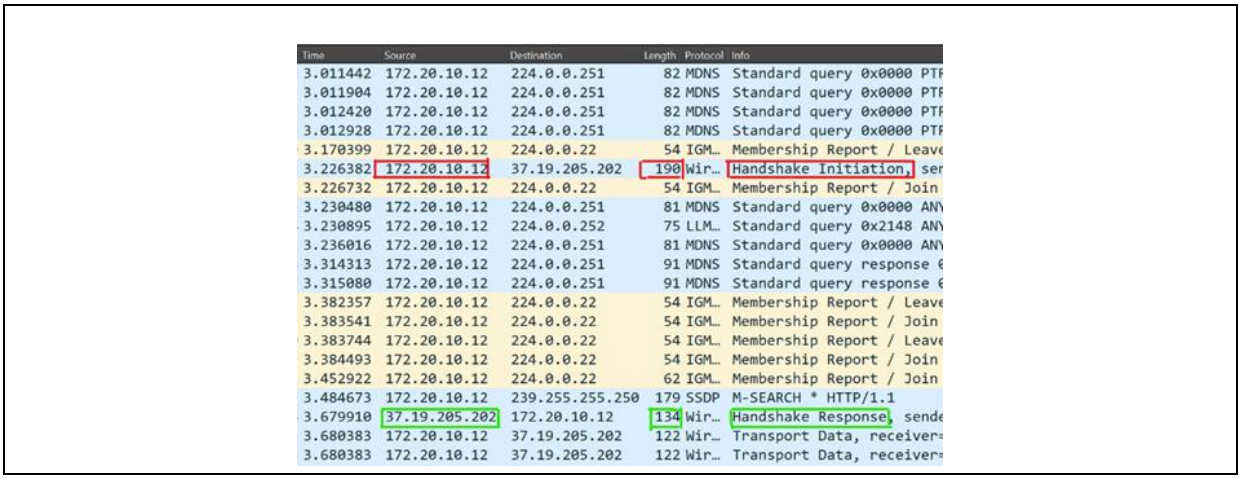
- Open the VPN application on User A device and Press Connect.
- Wait for 3 to 4s without interacting with the app.
- Click on Disconnect and close the application.

We saw that User A sends a Handshake Initiation packet to the VPN server. To establish a connection, a Handshake Response packet containing IP addresses was returned to the client. Furthermore, it was noted that User A was using a random port on their Proton VPN to connect to server 51820.

Following the exchange of session keys between the client and server, the subsequent packet patterns with payloads were observed between the server and client endpoints.

- Proton VPN Application (IP: 172.20.10.12) sent packets of (190) bytes with a payload of size (148), to the Proton servers.
- In response, the server (IP: 37.19.205.202) sent packets of (134) bytes to the VPN application with a payload of (92).

To validate the Proton VPN app's connection patterns, the aforementioned action was repeated several times, as seen in Figure 27. Thus, we deduced that the aforementioned byte transfer patterns signify the launch or establishment of the Proton VPN connection.

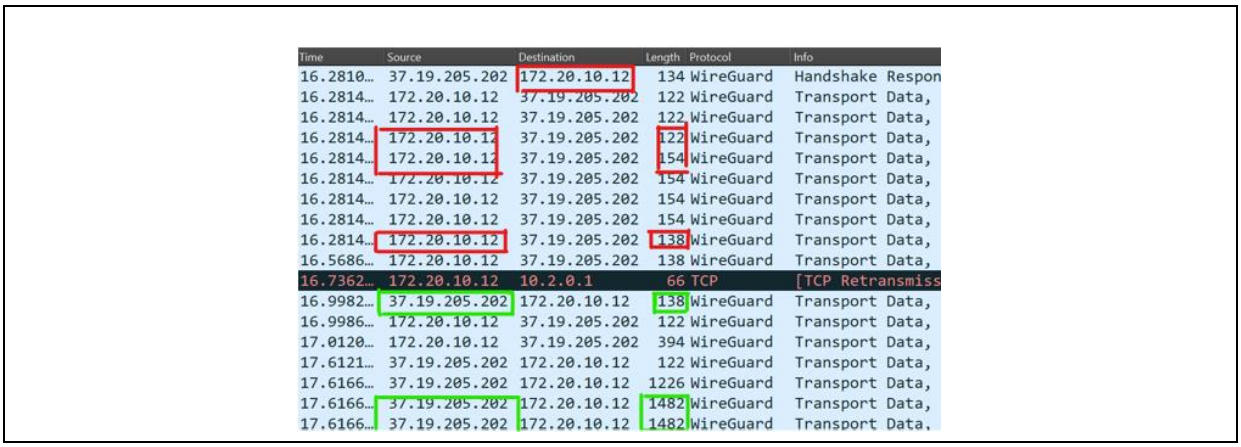


Time	Source	Destination	Length	Protocol	Info
3.011442	172.20.10.12	224.0.0.251	82	MDNS	Standard query 0x0000 PTR
3.011904	172.20.10.12	224.0.0.251	82	MDNS	Standard query 0x0000 PTR
3.012420	172.20.10.12	224.0.0.251	82	MDNS	Standard query 0x0000 PTR
3.012928	172.20.10.12	224.0.0.251	82	MDNS	Standard query 0x0000 PTR
3.170399	172.20.10.12	224.0.0.22	54	IGM…	Membership Report / Leave
3.226382	172.20.10.12	37.19.205.202	190	Wir…	Handshake Initiation, ser
3.226732	172.20.10.12	224.0.0.22	54	IGM…	Membership Report / Join
3.230480	172.20.10.12	224.0.0.251	81	MDNS	Standard query 0x0000 ANY
3.230895	172.20.10.12	224.0.0.252	75	LLM…	Standard query 0x2148 ANY
3.236016	172.20.10.12	224.0.0.251	81	MDNS	Standard query 0x0000 ANY
3.314313	172.20.10.12	224.0.0.251	91	MDNS	Standard query response 0
3.315080	172.20.10.12	224.0.0.251	91	MDNS	Standard query response 0
3.382357	172.20.10.12	224.0.0.22	54	IGM…	Membership Report / Leave
3.383541	172.20.10.12	224.0.0.22	54	IGM…	Membership Report / Join
3.383744	172.20.10.12	224.0.0.22	54	IGM…	Membership Report / Leave
3.384493	172.20.10.12	224.0.0.22	54	IGM…	Membership Report / Join
3.452922	172.20.10.12	224.0.0.22	62	IGM…	Membership Report / Join
3.484673	172.20.10.12	239.255.255.250	179	SSDP	M-SEARCH * HTTP/1.1
3.679910	37.19.205.202	172.20.10.12	134	Wir…	Handshake Response, sende
3.680383	172.20.10.12	37.19.205.202	122	Wir…	Transport Data, receiver=
3.680383	172.20.10.12	37.19.205.202	122	Wir…	Transport Data, receiver=

Figure 27: Proton VPN – Handshake Packets – WireGuard

Idle State after Connection

Certain flow patterns with defined payload sizes were observed to alert the Proton VPN app's patterns when target User A is connected and not actively using it. To be sure, similar patterns were often noticed in trace files, which led to the conclusions depicted in Figure 28. IP 172.20.10.12 is highlighted for Target User A. It was observed that when User A, whose IP address is 172.20.10.12, began transmitting data packets with payload sizes of 80, 112, and 96 bytes to the server, whose IP address is 37.19.205.202, 122, 154, and 138 bytes were sent.



Time	Source	Destination	Length	Protocol	Info
16.2810…	37.19.205.202	172.20.10.12	134	WireGuard	Handshake Respon
16.2814…	172.20.10.12	37.19.205.202	122	WireGuard	Transport Data,
16.2814…	172.20.10.12	37.19.205.202	122	WireGuard	Transport Data,
16.2814…	172.20.10.12	37.19.205.202	122	WireGuard	Transport Data,
16.2814…	172.20.10.12	37.19.205.202	154	WireGuard	Transport Data,
16.2814…	172.20.10.12	37.19.205.202	154	WireGuard	Transport Data,
16.2814…	172.20.10.12	37.19.205.202	154	WireGuard	Transport Data,
16.2814…	172.20.10.12	37.19.205.202	154	WireGuard	Transport Data,
16.2814…	172.20.10.12	37.19.205.202	138	WireGuard	Transport Data,
16.5686…	172.20.10.12	37.19.205.202	138	WireGuard	Transport Data,
16.7362…	172.20.10.12	10.2.0.1	66	TCP	[TCP Retransmiss
16.9982…	37.19.205.202	172.20.10.12	138	WireGuard	Transport Data,
16.9986…	172.20.10.12	37.19.205.202	122	WireGuard	Transport Data,
17.0120…	172.20.10.12	37.19.205.202	394	WireGuard	Transport Data,
17.6121…	37.19.205.202	172.20.10.12	122	WireGuard	Transport Data,
17.6166…	37.19.205.202	172.20.10.12	1226	WireGuard	Transport Data,
17.6166…	37.19.205.202	172.20.10.12	1482	WireGuard	Transport Data,
17.6166…	37.19.205.202	172.20.10.12	1482	WireGuard	Transport Data,

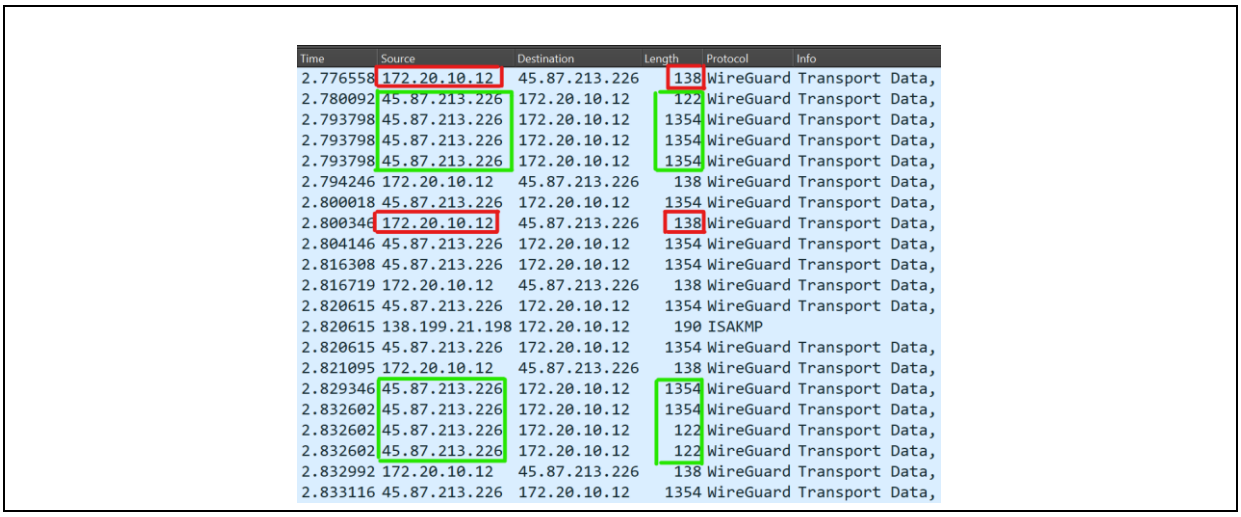
Figure 28: Proton VPN – Idle State – WireGuard

The server replied by sending the Proton VPN client 138 or 1482 bytes of data packets with payload sizes of 96, 1440. In the meantime, it was observed that even though nothing was being done, the Proton VPN app showed the connected status to the corresponding servers.

Accessing Google site

This section will look into the traffic patterns that User A utilizes to get to the Google website. It was consistently observed that, after User A's browser access to the webpage, the client at target User A sent 138 bytes of the packet in request with 96 bytes of payload to the server. The target server (IP: 45.87.213.226) received this request. The server sent 1354 and 122 bytes of packets with 1312 and 80 bytes of payload in response to the client at target User A's device. A sequence of 1354-byte and 122-byte data packets with payload sizes of 1312 and 80 bytes, respectively,

were transmitted by the server to the client, as shown in Figure 29. This pattern was observed and confirmed repeatedly through multiple iterations.

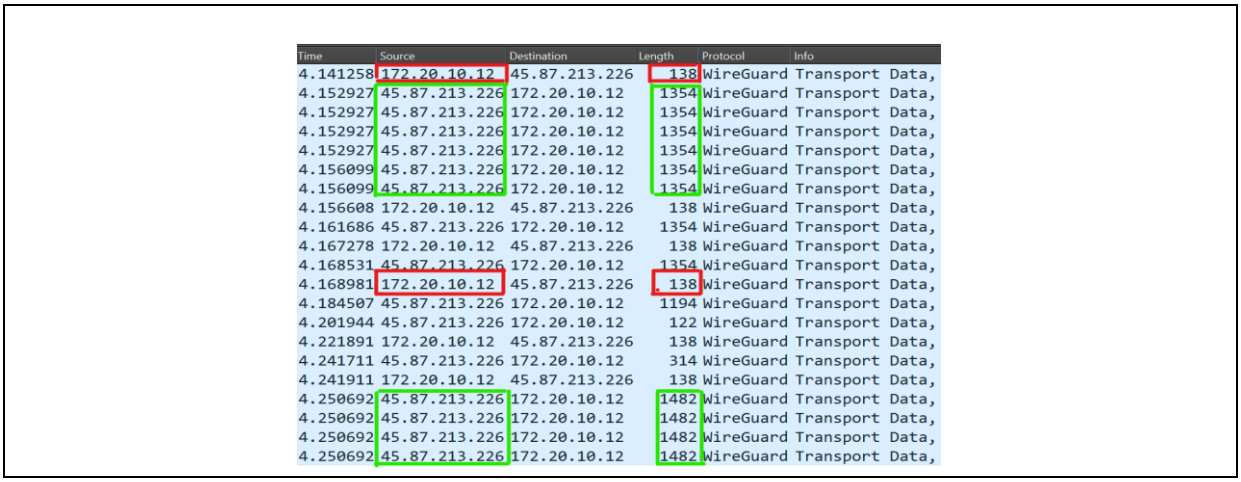


Time	Source	Destination	Length	Protocol	Info
2.776558	172.20.10.12	45.87.213.226	138	WireGuard	Transport Data,
2.780092	45.87.213.226	172.20.10.12	122	WireGuard	Transport Data,
2.793798	45.87.213.226	172.20.10.12	1354	WireGuard	Transport Data,
2.793798	45.87.213.226	172.20.10.12	1354	WireGuard	Transport Data,
2.793798	45.87.213.226	172.20.10.12	1354	WireGuard	Transport Data,
2.794246	172.20.10.12	45.87.213.226	138	WireGuard	Transport Data,
2.800018	45.87.213.226	172.20.10.12	1354	WireGuard	Transport Data,
2.800346	172.20.10.12	45.87.213.226	138	WireGuard	Transport Data,
2.804146	45.87.213.226	172.20.10.12	1354	WireGuard	Transport Data,
2.816308	45.87.213.226	172.20.10.12	1354	WireGuard	Transport Data,
2.816719	172.20.10.12	45.87.213.226	138	WireGuard	Transport Data,
2.820615	45.87.213.226	172.20.10.12	1354	WireGuard	Transport Data,
2.820615	138.199.21.198	172.20.10.12	190	ISAKMP	
2.820615	45.87.213.226	172.20.10.12	1354	WireGuard	Transport Data,
2.821095	172.20.10.12	45.87.213.226	138	WireGuard	Transport Data,
2.829346	45.87.213.226	172.20.10.12	1354	WireGuard	Transport Data,
2.832602	45.87.213.226	172.20.10.12	1354	WireGuard	Transport Data,
2.832602	45.87.213.226	172.20.10.12	122	WireGuard	Transport Data,
2.832602	45.87.213.226	172.20.10.12	122	WireGuard	Transport Data,
2.832992	172.20.10.12	45.87.213.226	138	WireGuard	Transport Data,
2.833116	45.87.213.226	172.20.10.12	1354	WireGuard	Transport Data,

Figure 29: Proton VPN – Accessing Google Website – WireGuard

Searching on Google

In this section, we will analyze the traffic patterns of User A when he/she performs a search on Google Search Engine. Once User A started searching his/her browser, which would be sent to the target server, it was always noticed that 1354 bytes of packets with 1312 bytes of payload were sent from the server to the client in response to the request. It was also observed that maximum packet length is 1482 byte with 1408 byte of payload.



Time	Source	Destination	Length	Protocol	Info
4.141258	172.20.10.12	45.87.213.226	138	WireGuard	Transport Data,
4.152927	45.87.213.226	172.20.10.12	1354	WireGuard	Transport Data,
4.152927	45.87.213.226	172.20.10.12	1354	WireGuard	Transport Data,
4.152927	45.87.213.226	172.20.10.12	1354	WireGuard	Transport Data,
4.152927	45.87.213.226	172.20.10.12	1354	WireGuard	Transport Data,
4.156099	45.87.213.226	172.20.10.12	1354	WireGuard	Transport Data,
4.156099	45.87.213.226	172.20.10.12	1354	WireGuard	Transport Data,
4.156608	172.20.10.12	45.87.213.226	138	WireGuard	Transport Data,
4.161686	45.87.213.226	172.20.10.12	1354	WireGuard	Transport Data,
4.167278	172.20.10.12	45.87.213.226	138	WireGuard	Transport Data,
4.168531	45.87.213.226	172.20.10.12	1354	WireGuard	Transport Data,
4.168981	172.20.10.12	45.87.213.226	138	WireGuard	Transport Data,
4.184507	45.87.213.226	172.20.10.12	1194	WireGuard	Transport Data,
4.201944	45.87.213.226	172.20.10.12	122	WireGuard	Transport Data,
4.221891	172.20.10.12	45.87.213.226	138	WireGuard	Transport Data,
4.241711	45.87.213.226	172.20.10.12	314	WireGuard	Transport Data,
4.241911	172.20.10.12	45.87.213.226	138	WireGuard	Transport Data,
4.250692	45.87.213.226	172.20.10.12	1482	WireGuard	Transport Data,
4.250692	45.87.213.226	172.20.10.12	1482	WireGuard	Transport Data,
4.250692	45.87.213.226	172.20.10.12	1482	WireGuard	Transport Data,
4.250692	45.87.213.226	172.20.10.12	1482	WireGuard	Transport Data,

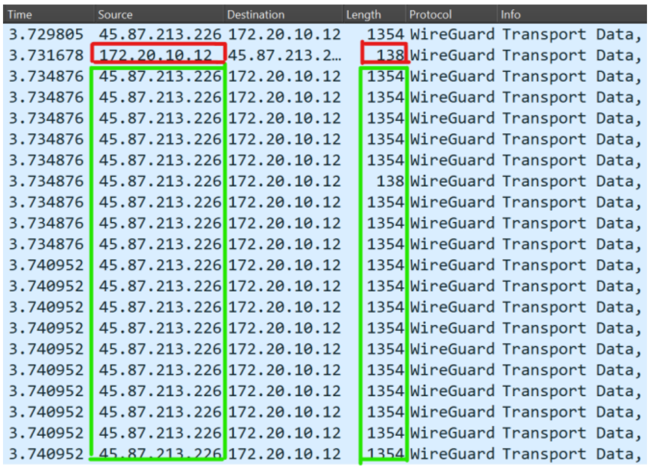
Figure 30: Proton VPN – Searching on Google – WireGuard

As seen in Figure 30, the client at target User A transmitted a packet containing 96 bytes of payload and 138 bytes of request. Through several cycles, this pattern was frequently seen and verified.

YouTube Video Streaming

Certain traffic patterns with defined payload sizes were observed to monitor the Proton VPN pattern while video content was played. A video request for 138 bytes with a 96-byte payload was delivered by the targeted user. The Proton VPN server replied by sending out 1354-byte packets with a payload of 1312 bytes. These packets occasionally grew to a size of 1482 by 138 bytes, with a payload of 1408 by

96 bytes. Most of the packets had a payload of 1312 bytes and a size of 1354 bytes. As seen in Figure 31, this trend was frequently noted and verified across several cycles.



Time	Source	Destination	Length	Protocol	Info
3.729805	45.87.213.226	172.20.10.12	1354	WireGuard	Transport Data,
3.731678	172.20.10.12	45.87.213.2…	138	WireGuard	Transport Data,
3.734876	45.87.213.226	172.20.10.12	1354	WireGuard	Transport Data,
3.734876	45.87.213.226	172.20.10.12	1354	WireGuard	Transport Data,
3.734876	45.87.213.226	172.20.10.12	1354	WireGuard	Transport Data,
3.734876	45.87.213.226	172.20.10.12	1354	WireGuard	Transport Data,
3.734876	45.87.213.226	172.20.10.12	1354	WireGuard	Transport Data,
3.734876	45.87.213.226	172.20.10.12	138	WireGuard	Transport Data,
3.734876	45.87.213.226	172.20.10.12	1354	WireGuard	Transport Data,
3.734876	45.87.213.226	172.20.10.12	1354	WireGuard	Transport Data,
3.734876	45.87.213.226	172.20.10.12	1354	WireGuard	Transport Data,
3.740952	45.87.213.226	172.20.10.12	1354	WireGuard	Transport Data,
3.740952	45.87.213.226	172.20.10.12	1354	WireGuard	Transport Data,
3.740952	45.87.213.226	172.20.10.12	1354	WireGuard	Transport Data,
3.740952	45.87.213.226	172.20.10.12	1354	WireGuard	Transport Data,
3.740952	45.87.213.226	172.20.10.12	1354	WireGuard	Transport Data,
3.740952	45.87.213.226	172.20.10.12	1354	WireGuard	Transport Data,
3.740952	45.87.213.226	172.20.10.12	1354	WireGuard	Transport Data,
3.740952	45.87.213.226	172.20.10.12	1354	WireGuard	Transport Data,
3.740952	45.87.213.226	172.20.10.12	1354	WireGuard	Transport Data,
3.740952	45.87.213.226	172.20.10.12	1354	WireGuard	Transport Data,

Figure 31: Proton VPN – YouTube Video Streaming – WireGuard

10.1.1.2 OpenVPN (UDP) Mode

OpenVPN is a highly customizable protocol that is relatively new. The fact that OpenVPN is open source is its greatest feature. Although the term "open" may not seem like a desirable thing for a privacy tool, it really has a lot of benefits. UDP ports work well for OpenVPN. The receiver cannot request that information be resent or acknowledge receipt of the data via UDP. As a result, UDP can create connections and send data more quickly.

Handshake with the VPN server

The establishment of connection activity of the Proton while using OpenVPN (UDP) is relatively different from any other connection formation. As UDP does not require handshake there are no handshake packets, but there must be an exchange. The process of the above activity is as follows:

- Open the VPN application on User A device and Press Connect.
- Wait for 3 to 4s without interacting with the app.
- Click on Disconnect and close the application.

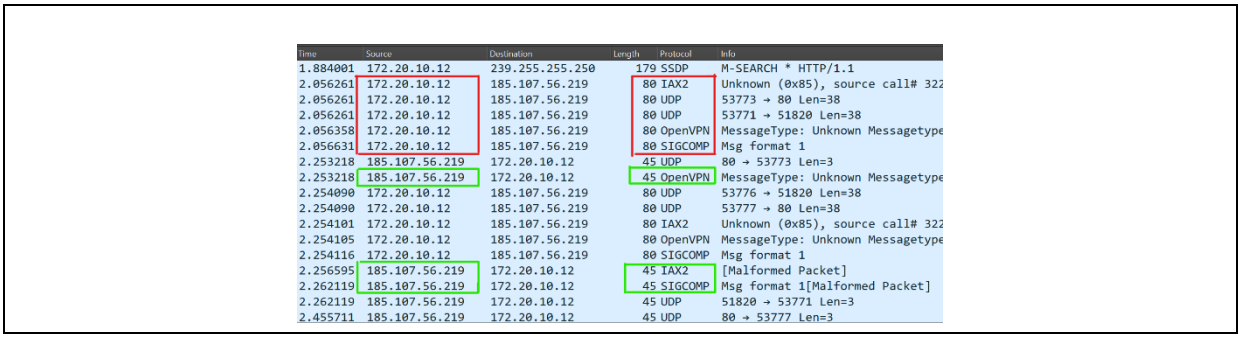
We observed that the first packets sent from User A to the VPN server uses IAX2 protocol followed by OpenVPN and SIGOMP. In response, the server sends a packet of OpenVPN protocol then followed by IAX2 and SIGCOMP packets. In addition, it was observed that User A's Proton VPN was connecting to the server 5060 [Table 9] over a random port on their end.

The following packet patterns with payloads were seen between the server and client endpoints following the negotiation of session keys between the client and server.

- Proton VPN Application (IP: 172.20.10.12) sent packets of (45,80) bytes with a payload of size (3, 38), to the Proton servers.
- In response, the server (IP: 185.107.56.219) sent packets of (45,80) bytes to the VPN application with a payload of (3, 38).

To validate the Proton VPN app's connection patterns, the aforementioned

action was repeated several times, as seen in Figure 32. Thus, we deduced that the aforementioned byte transfer patterns signify the launch or establishment of the Proton VPN connection.

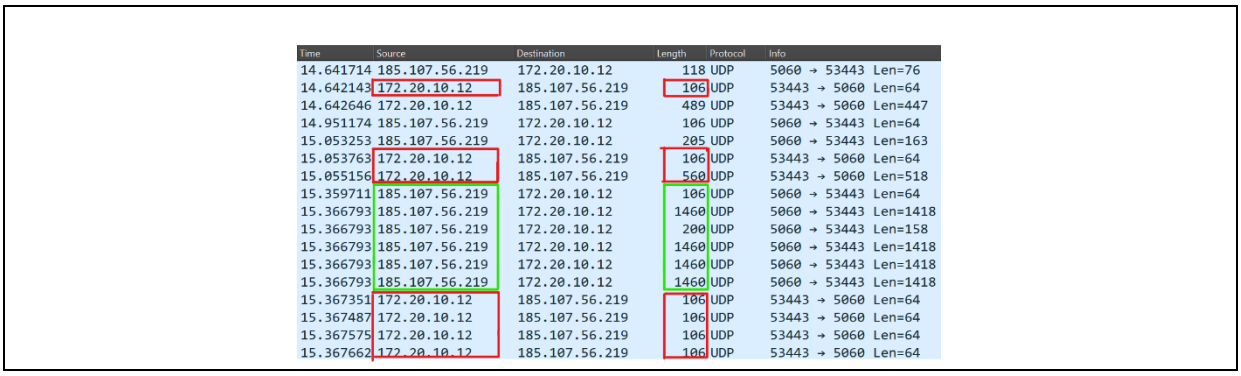


Time	Source	Destination	Length	Protocol	Info
1.884001	172.20.10.12	239.255.255.250	179	SSDP	M-SEARCH * HTTP/1.1
2.056261	172.20.10.12	185.107.56.219	80	IAX2	Unknown (0x85), source call# 322
2.056261	172.20.10.12	185.107.56.219	80	UDP	53773 → 80 Len=38
2.056261	172.20.10.12	185.107.56.219	80	UDP	53771 → 51820 Len=38
2.056358	172.20.10.12	185.107.56.219	80	OpenVPN	MessageType: Unknown Messagetype
2.056631	172.20.10.12	185.107.56.219	80	SIGCOMP	Msg format 1
2.253218	185.107.56.219	172.20.10.12	45	UDP	80 → 53773 Len=3
2.253218	185.107.56.219	172.20.10.12	45	OpenVPN	MessageType: Unknown Messagetype
2.254090	172.20.10.12	185.107.56.219	80	UDP	53776 → 51820 Len=38
2.254090	172.20.10.12	185.107.56.219	80	UDP	53777 → 80 Len=38
2.254101	172.20.10.12	185.107.56.219	80	IAX2	Unknown (0x85), source call# 322
2.254105	172.20.10.12	185.107.56.219	80	OpenVPN	MessageType: Unknown Messagetype
2.254116	172.20.10.12	185.107.56.219	80	SIGCOMP	Msg format 1
2.256595	185.107.56.219	172.20.10.12	45	IAX2	[Malformed Packet]
2.262119	185.107.56.219	172.20.10.12	45	SIGCOMP	Msg format 1[Malformed Packet]
2.262119	185.107.56.219	172.20.10.12	45	UDP	51820 → 53771 Len=3
2.455711	185.107.56.219	172.20.10.12	45	UDP	80 → 53777 Len=3

Figure 32: Proton VPN – Handshake Packets – OpenVPN (UDP)

Idle State after Connection

Certain flow patterns with defined payload sizes were observed to alert the Proton VPN app's patterns when target User A is connected and not actively using it. To be sure, similar patterns were repeatedly noticed in trace files to infer conclusions, as Figure 33 illustrates. IP 172.20.10.12 is highlighted for Target User A. The more noteworthy thing was observed when User A, whose IP address was 172.20.10.12, began sending packets to the server, which had the IP address 185.107.56.219, containing 106 bytes of data and 64 payloads. In response, the server sent data packets of 106 and 1460 bytes, with payload sizes of 64 and 1418, to the Proton VPN client. In the meantime, it was observed that even though nothing was being done, the Proton VPN app showed the connected status to the corresponding servers.



Time	Source	Destination	Length	Protocol	Info
14.641714	185.107.56.219	172.20.10.12	118	UDP	5060 → 53443 Len=76
14.642143	172.20.10.12	185.107.56.219	106	UDP	53443 → 5060 Len=64
14.642646	172.20.10.12	185.107.56.219	489	UDP	53443 → 5060 Len=447
14.951174	185.107.56.219	172.20.10.12	106	UDP	5060 → 53443 Len=64
15.053253	185.107.56.219	172.20.10.12	205	UDP	5060 → 53443 Len=163
15.053763	172.20.10.12	185.107.56.219	106	UDP	53443 → 5060 Len=64
15.055156	172.20.10.12	185.107.56.219	560	UDP	53443 → 5060 Len=518
15.359711	185.107.56.219	172.20.10.12	106	UDP	5060 → 53443 Len=64
15.366793	185.107.56.219	172.20.10.12	1460	UDP	5060 → 53443 Len=1418
15.366793	185.107.56.219	172.20.10.12	200	UDP	5060 → 53443 Len=158
15.366793	185.107.56.219	172.20.10.12	1460	UDP	5060 → 53443 Len=1418
15.366793	185.107.56.219	172.20.10.12	1460	UDP	5060 → 53443 Len=1418
15.366793	185.107.56.219	172.20.10.12	1460	UDP	5060 → 53443 Len=1418
15.367351	172.20.10.12	185.107.56.219	106	UDP	53443 → 5060 Len=64
15.367487	172.20.10.12	185.107.56.219	106	UDP	53443 → 5060 Len=64
15.367575	172.20.10.12	185.107.56.219	106	UDP	53443 → 5060 Len=64
15.367662	172.20.10.12	185.107.56.219	106	UDP	53443 → 5060 Len=64

Figure 33: Proton VPN – Idle State – OpenVPN (UDP)

Accessing Google site

This section will investigate the traffic patterns that User A utilizes to get to the Google website. It was consistently observed that when User A browsed the webpage in his or her browser, the client at target User A sent 127 bytes of the packet in request with 85 bytes of payload to the server. The target server (IP: 185.107.56.219) received this request. Target User A's device received a response from the server consisting of 1344 bytes of packets with 1302 bytes of payload. Figure 34 illustrates how this trend was consistently noticed and verified over the course of several rounds.

Time	Source	Destination	Length	Protocol	Info
4.856602	172.20.10.12	185.107.56.219	127	UDP	53443 → 5060 Len=85
4.857095	185.107.56.219	172.20.10.12	1344	UDP	5060 → 53443 Len=1302
4.857095	185.107.56.219	172.20.10.12	1344	UDP	5060 → 53443 Len=1302
4.857095	185.107.56.219	172.20.10.12	1344	UDP	5060 → 53443 Len=1302
4.857095	185.107.56.219	172.20.10.12	1344	UDP	5060 → 53443 Len=1302
4.857095	185.107.56.219	172.20.10.12	1344	UDP	5060 → 53443 Len=1302
4.857095	185.107.56.219	172.20.10.12	1344	UDP	5060 → 53443 Len=1302
4.857095	185.107.56.219	172.20.10.12	1344	UDP	5060 → 53443 Len=1302
4.857095	185.107.56.219	172.20.10.12	1344	UDP	5060 → 53443 Len=1302
4.857532	172.20.10.12	185.107.56.219	127	UDP	53443 → 5060 Len=85
4.868710	185.107.56.219	172.20.10.12	1344	UDP	5060 → 53443 Len=1302
4.868710	185.107.56.219	172.20.10.12	1344	UDP	5060 → 53443 Len=1302
4.868710	185.107.56.219	172.20.10.12	1344	UDP	5060 → 53443 Len=1302
4.869731	185.107.56.219	172.20.10.12	1344	UDP	5060 → 53443 Len=1302
4.869731	185.107.56.219	172.20.10.12	1344	UDP	5060 → 53443 Len=1302
4.869731	185.107.56.219	172.20.10.12	1344	UDP	5060 → 53443 Len=1302
4.869731	185.107.56.219	172.20.10.12	1344	UDP	5060 → 53443 Len=1302

Figure 34: Proton VPN – Accessing Google Website – OpenVPN (UDP)

Searching on Google

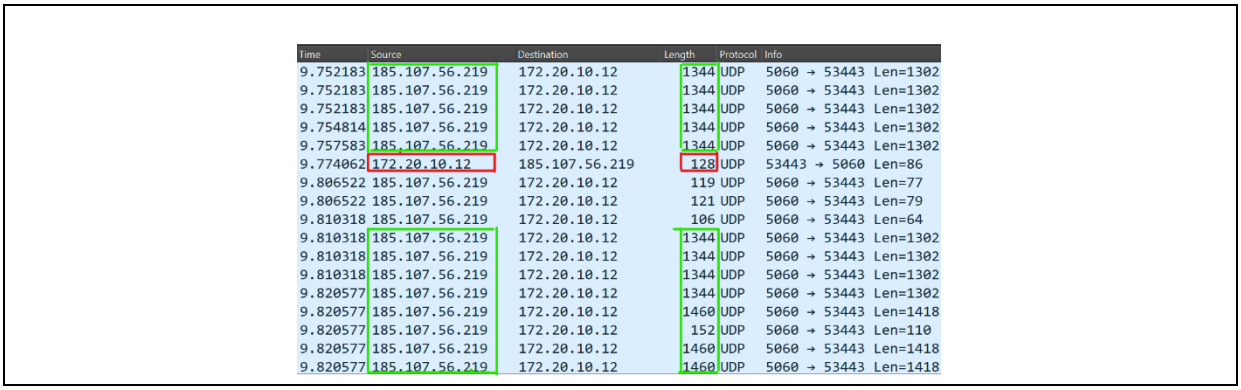
We will examine User A's search traffic patterns in this part when the user conducts a Google search. It was observed that 1344 bytes of packets with 1302 bytes of payload were transferred from the server to the client in response to the request after User A began searching his or her browser, which would be transmitted to the target server. Additionally, a maximum packet length of 1460 bytes with a payload of 1418 bytes was noted. As seen in Figure 35, the client at target User A transmitted 126 bytes of the packet along with 84 bytes of payload as required. Through several cycles, this pattern was frequently seen and verified.

Time	Source	Destination	Length	Protocol	Info
2.957937	172.20.10.12	185.107.56.219	126	UDP	53443 → 5060 Len=84
2.960726	185.107.56.219	172.20.10.12	1344	UDP	5060 → 53443 Len=1302
2.960726	185.107.56.219	172.20.10.12	1344	UDP	5060 → 53443 Len=1302
2.960726	185.107.56.219	172.20.10.12	1344	UDP	5060 → 53443 Len=1302
2.960726	185.107.56.219	172.20.10.12	1344	UDP	5060 → 53443 Len=1302
2.960726	185.107.56.219	172.20.10.12	1344	UDP	5060 → 53443 Len=1302
2.960726	185.107.56.219	172.20.10.12	1344	UDP	5060 → 53443 Len=1302
2.960726	185.107.56.219	172.20.10.12	1344	UDP	5060 → 53443 Len=1302
2.961375	185.107.56.219	172.20.10.12	1344	UDP	5060 → 53443 Len=1302
2.961375	185.107.56.219	172.20.10.12	1344	UDP	5060 → 53443 Len=1302
2.961375	185.107.56.219	172.20.10.12	1344	UDP	5060 → 53443 Len=1302
2.961375	185.107.56.219	172.20.10.12	1344	UDP	5060 → 53443 Len=1302
2.961867	172.20.10.12	185.107.56.219	126	UDP	53443 → 5060 Len=84
2.964683	185.107.56.219	172.20.10.12	1344	UDP	5060 → 53443 Len=1302
2.964683	185.107.56.219	172.20.10.12	1344	UDP	5060 → 53443 Len=1302
2.964683	185.107.56.219	172.20.10.12	1344	UDP	5060 → 53443 Len=1302
2.964683	185.107.56.219	172.20.10.12	1344	UDP	5060 → 53443 Len=1302

Figure 35: Proton VPN – Searching on Google – OpenVPN (UDP)

YouTube Video Streaming

Certain traffic patterns with defined payload sizes were observed to monitor the Proton VPN pattern while video content was played. A video request for 128, 126 bytes with a payload size of 86, 84 bytes was delivered by the targeted user. The Proton VPN server replied by sending out 1344-byte packets containing 1302 bytes of payload. These packets occasionally grew to a size of 1460 bytes, with a payload of 1418 bytes. Most of the packets had a payload of 1302 bytes and a size of 1344 bytes. As seen in Figure 36, this trend was frequently noted and verified via several rounds.



Time	Source	Destination	Length	Protocol	Info
9.752183	185.107.56.219	172.20.10.12	1344	UDP	5060 → 53443 Len=1302
9.752183	185.107.56.219	172.20.10.12	1344	UDP	5060 → 53443 Len=1302
9.752183	185.107.56.219	172.20.10.12	1344	UDP	5060 → 53443 Len=1302
9.754814	185.107.56.219	172.20.10.12	1344	UDP	5060 → 53443 Len=1302
9.757583	185.107.56.219	172.20.10.12	1344	UDP	5060 → 53443 Len=1302
9.774062	172.20.10.12	185.107.56.219	128	UDP	53443 → 5060 Len=86
9.806522	185.107.56.219	172.20.10.12	119	UDP	5060 → 53443 Len=77
9.806522	185.107.56.219	172.20.10.12	121	UDP	5060 → 53443 Len=79
9.810318	185.107.56.219	172.20.10.12	106	UDP	5060 → 53443 Len=64
9.810318	185.107.56.219	172.20.10.12	1344	UDP	5060 → 53443 Len=1302
9.810318	185.107.56.219	172.20.10.12	1344	UDP	5060 → 53443 Len=1302
9.810318	185.107.56.219	172.20.10.12	1344	UDP	5060 → 53443 Len=1302
9.820577	185.107.56.219	172.20.10.12	1344	UDP	5060 → 53443 Len=1302
9.820577	185.107.56.219	172.20.10.12	1460	UDP	5060 → 53443 Len=1418
9.820577	185.107.56.219	172.20.10.12	152	UDP	5060 → 53443 Len=110
9.820577	185.107.56.219	172.20.10.12	1460	UDP	5060 → 53443 Len=1418
9.820577	185.107.56.219	172.20.10.12	1460	UDP	5060 → 53443 Len=1418

Figure 36: Proton VPN – YouTube Video Streaming – OpenVPN (UDP)

10.1.1.3 OpenVPN (TCP) Mode

OpenVPN is a highly customizable protocol that is relatively new. The fact that OpenVPN is open source is its greatest feature. Although the term "open" may not seem like a desirable thing for a privacy tool, it really has a lot of benefits. In contrast to UDP, TCP makes sure that the receiver receives the data in the right format and sequence before allowing the request to be made again. As a result, latency increases at the price of higher dependability.

Handshake with the VPN server

The Handshake activity of the Proton VPN represents the start of application on the User A's device. Initially, it was difficult to classify the VPN only traffic from the shared network traffic packets. Therefore, an extensive number of trace files were captured and monitored during the handshake of the Proton VPN application with VPN server.

The process of the above activity is as follows:

- Open the VPN application on User A device and Press Connect.
- Wait for 3 to 4s without interacting with the app.
- Click on Disconnect and close the application.

We observed that a standard DNS query was sent to access the Proton VPN server from the client end, i.e., vpn-api.proton.me OPT. Between the client and server, a TCP connection was formed in three stages: SYN, SYN ACK, and ACK. Following that, TLSv1.3 was used for the TLS handshake. The client and server exchanged certificates and session keys. The client then sends the server an encrypted handshake message after exchanging the key change cypher specifications. The server responded by sending the client device a new session ticket, modifying the cypher specs, and sending an encrypted handshake message over TLS. In response, a server's IP address was sent back to the client to establish a connection. We observed that a Handshake Initiation packet was sent to the VPN Server from User A. In response, a Handshake Response packet with IP addresses was sent back to the client to establish a connection. In addition, it was observed that User A's Proton VPN was connecting to the server 443 [Table 9] over a random port on their end.

The following packet patterns with payloads were seen between the server and client endpoints following the negotiation of session keys between the client and server.

- Proton VPN Application (IP: 172.20.10.12) sent packets of (66, 54) bytes with a payload of size (34, 34), to the Proton servers.
- In response, the server (IP: 185.159.159.148) sent packets of (66, 54) bytes to the VPN application with a payload of (34, 34).

To validate the Proton VPN app's connection patterns, the aforementioned action was repeated several times, as seen in Figure 37. Thus, we deduced that the aforementioned byte transfer patterns signify the launch or establishment of the Proton VPN connection.

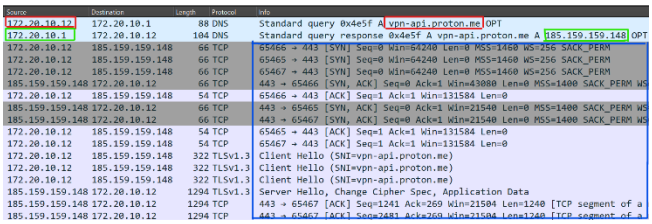


Figure 37: Proton VPN – Handshake Packets – OpenVPN (TCP)

Idle State after Connection

Certain flow patterns with defined payload sizes were observed to alert the Proton VPN app's patterns when target User A is connected and not actively using it. To be sure, similar patterns were repeatedly noticed in trace files to infer conclusions, as Figure 38 illustrates. IP 172.20.10.12 is highlighted for Target User A. More notably, it was observed that when User A with IP 172.20.10.12 began sending packets to the server with IP 185.159.159.148, the packets had 34 payload sizes and 54 bytes of data. In response, the server sent 54 bytes of data packets with 34 bytes of payload size to the Proton VPN client. In the meantime, it was observed that even though nothing was being done, the Proton VPN app showed the connected status to the corresponding servers using SSL packets.

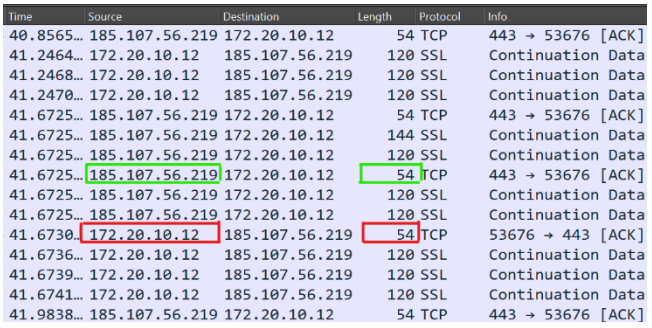
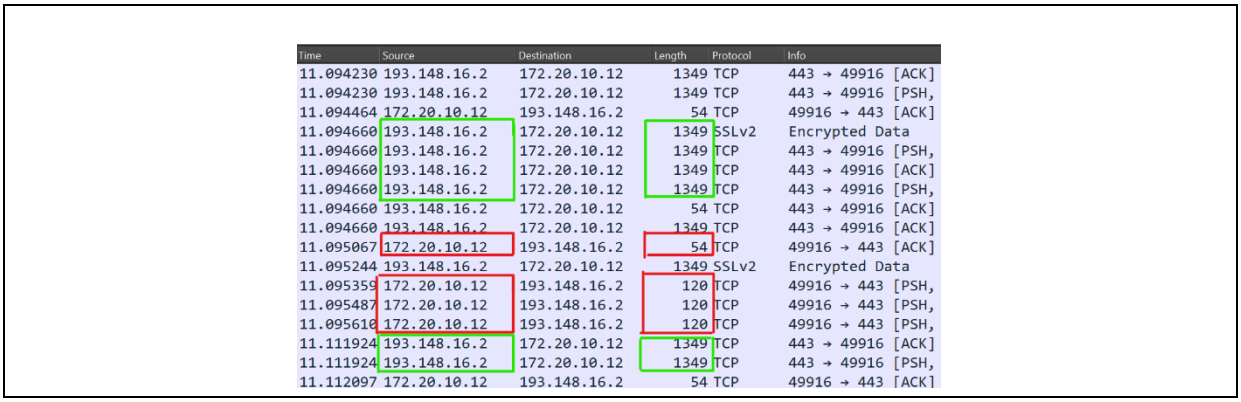


Figure 38: Proton VPN – Idle State – OpenVPN (TCP)

Accessing Google site

The traffic patterns that User A uses to reach the Google website will be examined in this section. It was always seen that the client at target User A transmitted 54 and 120 bytes of the packet in request with 34 and 66 bytes of payload to the server after User A accessed the webpage in his or her browser. This request was made to the target server (IP: 193.148.16.2). In answer to the client at target User A's device, the server transmitted 1349 bytes of packets containing 1295

bytes of payload. As seen in Figure 39 this pattern was observed and confirmed repeatedly through multiple iterations.

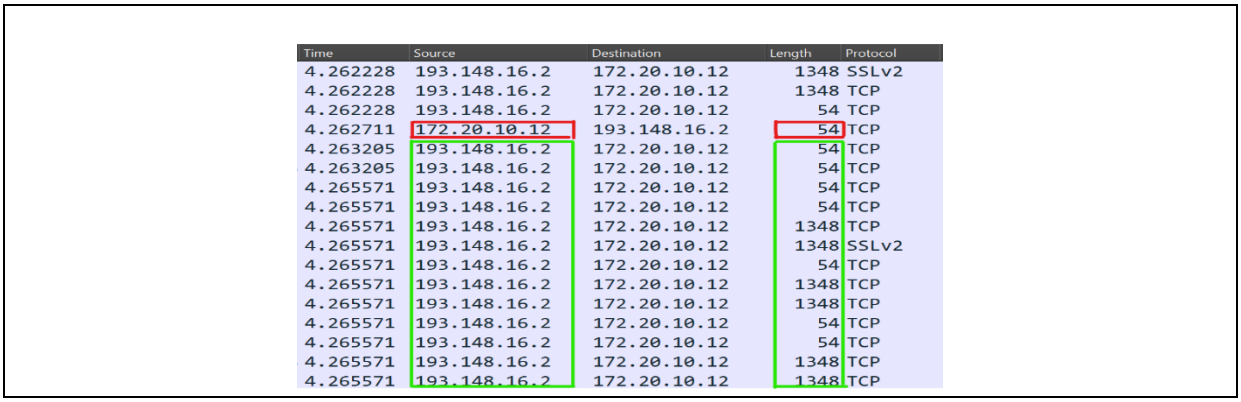


Time	Source	Destination	Length	Protocol	Info
11.094230	193.148.16.2	172.20.10.12	1349	TCP	443 → 49916 [ACK]
11.094230	193.148.16.2	172.20.10.12	1349	TCP	443 → 49916 [PSH,
11.094464	172.20.10.12	193.148.16.2	54	TCP	49916 → 443 [ACK]
11.094660	193.148.16.2	172.20.10.12	1349	SSLv2	Encrypted Data
11.094660	193.148.16.2	172.20.10.12	1349	TCP	443 → 49916 [PSH,
11.094660	193.148.16.2	172.20.10.12	1349	TCP	443 → 49916 [ACK]
11.094660	193.148.16.2	172.20.10.12	1349	TCP	443 → 49916 [PSH,
11.094660	193.148.16.2	172.20.10.12	54	TCP	443 → 49916 [ACK]
11.094660	193.148.16.2	172.20.10.12	1349	TCP	443 → 49916 [ACK]
11.095067	172.20.10.12	193.148.16.2	54	TCP	49916 → 443 [ACK]
11.095244	193.148.16.2	172.20.10.12	1349	SSLv2	Encrypted Data
11.095359	172.20.10.12	193.148.16.2	120	TCP	49916 → 443 [PSH,
11.095487	172.20.10.12	193.148.16.2	120	TCP	49916 → 443 [PSH,
11.095618	172.20.10.12	193.148.16.2	120	TCP	49916 → 443 [PSH,
11.111924	193.148.16.2	172.20.10.12	1349	TCP	443 → 49916 [ACK]
11.111924	193.148.16.2	172.20.10.12	1349	TCP	443 → 49916 [PSH,
11.112097	172.20.10.12	193.148.16.2	54	TCP	49916 → 443 [ACK]

Figure 39: Proton VPN – Accessing Google Website – OpenVPN (TCP)

Searching on Google

We will examine User A's search traffic patterns in this part when the user conducts a Google search. It was observed that 1348 and 54 bytes of packets with 1294 and 34 bytes of payload were transferred from the server to the client in response to the request after User A began searching his or her browser, which would be transmitted to the target server. Additionally, a maximum packet length of 1358 bytes with a payload of 1304 bytes was noted. As seen in Figure 40, the client at target User A transmitted 54 bytes of the packet as a request with 34 bytes of payload. Through several cycles, this pattern was frequently seen and verified.

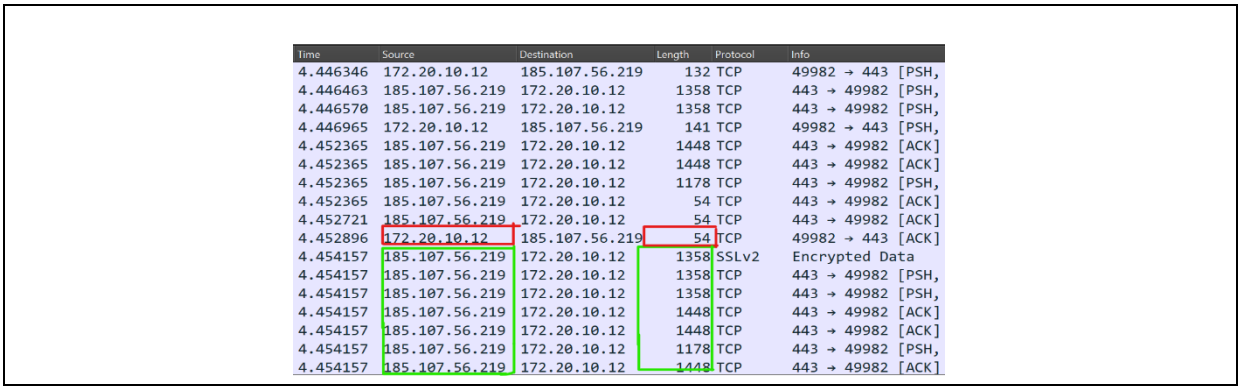


Time	Source	Destination	Length	Protocol
4.262228	193.148.16.2	172.20.10.12	1348	SSLv2
4.262228	193.148.16.2	172.20.10.12	1348	TCP
4.262228	193.148.16.2	172.20.10.12	54	TCP
4.262711	172.20.10.12	193.148.16.2	54	TCP
4.263205	193.148.16.2	172.20.10.12	54	TCP
4.263205	193.148.16.2	172.20.10.12	54	TCP
4.265571	193.148.16.2	172.20.10.12	54	TCP
4.265571	193.148.16.2	172.20.10.12	54	TCP
4.265571	193.148.16.2	172.20.10.12	1348	TCP
4.265571	193.148.16.2	172.20.10.12	1348	SSLv2
4.265571	193.148.16.2	172.20.10.12	54	TCP
4.265571	193.148.16.2	172.20.10.12	1348	TCP
4.265571	193.148.16.2	172.20.10.12	1348	TCP
4.265571	193.148.16.2	172.20.10.12	54	TCP
4.265571	193.148.16.2	172.20.10.12	54	TCP
4.265571	193.148.16.2	172.20.10.12	1348	TCP
4.265571	193.148.16.2	172.20.10.12	1348	TCP

Figure 40: Proton VPN – Searching on Google – OpenVPN (TCP)

YouTube Video Streaming

To observe the pattern of Proton VPN when a video content was played, as a result certain flow patterns with fixed payload sizes were noticed. Targeted User A sent a request for 54 bytes (which is the most prominent packet size) with payload size of 34 bytes for a video. In response, the Proton VPN server responded with 1358- and 1448-bytes packets with 1304 and 1394 bytes of payload. Occasionally, these packets reached the size of 1448 bytes with payload size of 1439 bytes.



Time	Source	Destination	Length	Protocol	Info
4.446346	172.20.10.12	185.107.56.219	132	TCP	49982 → 443 [PSH,
4.446463	185.107.56.219	172.20.10.12	1358	TCP	443 → 49982 [PSH,
4.446570	185.107.56.219	172.20.10.12	1358	TCP	443 → 49982 [PSH,
4.446965	172.20.10.12	185.107.56.219	141	TCP	49982 → 443 [PSH,
4.452365	185.107.56.219	172.20.10.12	1448	TCP	443 → 49982 [ACK]
4.452365	185.107.56.219	172.20.10.12	1448	TCP	443 → 49982 [ACK]
4.452365	185.107.56.219	172.20.10.12	1178	TCP	443 → 49982 [PSH,
4.452365	185.107.56.219	172.20.10.12	54	TCP	443 → 49982 [ACK]
4.452721	185.107.56.219	172.20.10.12	54	TCP	443 → 49982 [ACK]
4.452896	172.20.10.12	185.107.56.219	54	TCP	49982 → 443 [ACK]
4.454157	185.107.56.219	172.20.10.12	1358	SSLv2	Encrypted Data
4.454157	185.107.56.219	172.20.10.12	1358	TCP	443 → 49982 [PSH,
4.454157	185.107.56.219	172.20.10.12	1358	TCP	443 → 49982 [PSH,
4.454157	185.107.56.219	172.20.10.12	1448	TCP	443 → 49982 [ACK]
4.454157	185.107.56.219	172.20.10.12	1448	TCP	443 → 49982 [ACK]
4.454157	185.107.56.219	172.20.10.12	1178	TCP	443 → 49982 [PSH,
4.454157	185.107.56.219	172.20.10.12	1448	TCP	443 → 49982 [ACK]

Figure 41: Proton VPN – YouTube Video Streaming – OpenVPN (TCP)

. The majority of packets were at size 1358 byte with 1304 byte of payload. This pattern was observed and confirmed repeatedly through multiple iterations as shown in Figure 41.

VPN	WireGuard	OpenVPN (UDP)	OpenVPN (TCP)	IKEv2	Stealth	WStunnel	SoftEther	SSTP	Hydra	TCP	UDP
Proton VPN	✓	✓	✓	X	X	X	X	X	X	X	X
Hide.me VPN	✓	✓	✓	✓	X	X	✓	✓	X	X	X
Windscribe VPN	✓	X	X	✓	✓	✓	X	X	X	✓	✓
Tunnel Bear VPN	✓	X	X	✓	X	X	X	X	X	X	X
Turbo VPN	✓	X	X	X	X	X	X	X	X	X	X
Cloudflare WARP	X	X	X	X	X	X	X	X	X	X	X
Hotspot Shield VPN	✓	X	X	✓	X	X	X	X	✓	X	X

Table 9: VPN and their used Protocols

10.1.2 Hide.me VPN

Hide.me VPN provides encryption, wi-fi security, and privacy protection for a genuinely private online browsing experience, no matter where you are. It makes use of SSTP, WireGuard, OpenVPN, SoftEther VPN, and IKEv2 [Table 2].

In this study, we thoroughly examined Hide.me VPN across a range of criteria, adhering to the same methods used to assess Proton VPN. We carefully watched and categorized our target user, User A,'s application behaviors on their device to have a thorough insight. These included the handshake procedure with the VPN server, the idle state following the formation of a connection, browsing Google websites, using Google to search, and watching YouTube videos. We have condensed the in-depth observations into Table 9 for convenience of reference and interpretation to simplify the presenting of our findings.

10.1.3 Windscribe VPN

Based in Canada, Windscribe is a cross-platform, commercial virtual private network (VPN) service provider with operations throughout the globe. In its manual setups and applications, Windscribe makes use of IKEv2, WireGuard, and OpenVPN protocols [Table 9]. P2P file sharing is supported on Windscribe servers, who advertise themselves as a no-log VPN service in their privacy statement.

Following up on our research into VPN services, similar to the methodology used for Proton VPN and Hide.me VPN, we carefully evaluated Windscribe VPN based on a number of performance indicators. We examined User A's device app behaviors across several important factors to gain further insight into the operation and effectiveness of Windscribe VPN. To improve reader accessibility and understanding and to enable a concise presentation of our findings, we have summarized the in-depth observations in Table 9.

10.1.4 Tunnel-Bear VPN

The simplest VPN in the world to use is Tunnel-Bear. The way Tunnel-Bear operates is by enabled to connect to any place worldwide over an encrypted tunnel. Users may browse the internet as if they were physically in the nation they are connected to, and their true IP address stays disguised while you are connected. It uses WireGuard and IKEv2.

We thoroughly investigated Tunnel-Bear VPN using the same approach we used to evaluate ProtonVPN, Hide.me VPN and Windscribe VPN. Our research included a detailed examination of the VPN service's functionality according to several different criteria. We have condensed the in-depth observations into Table 9 to simplify the presentation of our findings and offer a clear and concise summary for further examination and comparison.

10.1.5 Turbo VPN

Turbo VPN, created in 2018 by Innovative Connecting, a Singaporean firm, has received about 180,000 ratings on the Apple App Store and over 100 million downloads on the Google Play Store. It uses WireGuard Protocol.

Following the established process that we used to assess Previous VPN, we looked more closely at Turbo VPN. We meticulously examined the VPN service's performance in relation to a number of different metrics as part of our study. We did this by carefully observing and classifying our assigned user, User A,'s application behaviors on their device. To improve readability and enable comparison, we have condensed our in-depth observations into Table 9, which provides a concise synopsis for thorough comprehension and analysis.

10.1.6 Cloudflare WARP

Much attention was aroused when Cloudflare defined WARP as a fast, thin, mobile-only VPN that doesn't drain your phone's battery. People may enjoy the internet quicker, more securely, and privately with the help of the Cloudflare WARP client. The WARP client, which stands in between your device and the Internet, offers a variety of connection types to accommodate various requirements. It does not use the traditional protocols like other VPNs.

Following the established process that we used to assess Previous VPN, we looked more closely at Cloudflare WARP. To improve readability and enable comparison, we have condensed our in-depth observations into Table 9, which gives

a clear summary for further study and assessment.

10.1.7 Hotspot Shield VPN

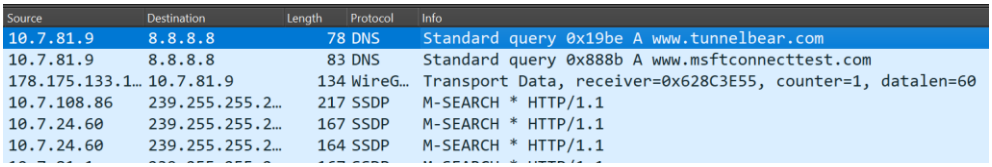
Anchor Free, Inc. is the company behind Hotspot Shield, a free VPN service. Through one of its supported public VPN servers, the Hotspot Shield client creates an encrypted VPN connection that allows the user to access the Internet. It makes use of IKEv2, WireGuard, and Hydra protocols.

Following the same protocol that was used to assess Proton VPN, Windscribe VPN, Tunnel-Bear VPN, and Turbo VPN, we also looked at Hotspot Shield VPN. We performed a comprehensive analysis of the VPN service's performance across several factors by using a methodical methodology. These metrics included the handshake with the VPN server, the idle state following the formation of the connection, the use of Google sites, Google searches, and YouTube video streaming. We condensed our in-depth findings into Table 9 to ensure clarity and make comparison easier. This gives a clear summary for further study and assessment.

10.2 Blocking of VPNs

10.2.1 Blocking Tunnel Bear VPN

While performing all the above-mentioned activities, certain results are deduced. For example, during multiple activities, we observed that tunnel bear operates on port 4500 on both server and client side. While monitoring the network packets during the establishment of connection between the User A and Tunnel Bear VPN app, it was noted that the app always connects to one of the servers mentioned in the DNS query sent, i.e., www.tunnelbear.com. It was also observed that an application establishes a connection with one of these servers using 4500 ports. The details of all those servers can be found in packets details in Wireshark as shown in Figure 42 & 43.



Source	Destination	Length	Protocol	Info
10.7.81.9	8.8.8.8	78	DNS	Standard query 0x19be A www.tunnelbear.com
10.7.81.9	8.8.8.8	83	DNS	Standard query 0x888b A www.msftconnecttest.com
178.175.133.1…	10.7.81.9	134	WireG…	Transport Data, receiver=0x628C3E55, counter=1, datalen=60
10.7.108.86	239.255.255.2…	217	SSDP	M-SEARCH * HTTP/1.1
10.7.24.60	239.255.255.2…	167	SSDP	M-SEARCH * HTTP/1.1
10.7.24.60	239.255.255.2…	164	SSDP	M-SEARCH * HTTP/1.1

Figure 42: Wireshark Packets

Figure 43: DNS Packet

Now, we update the firewall rules such that it blocks any UDP from LAN

Subnets from 4500 to any destination at 4500 port, Figure 44. After which, when the Tunnel Bear VPN application attempts to connect with its servers but PfSense Firewall blocks the traffic because the firewall always provides an edge to control and monitor the ongoing activities within the network.

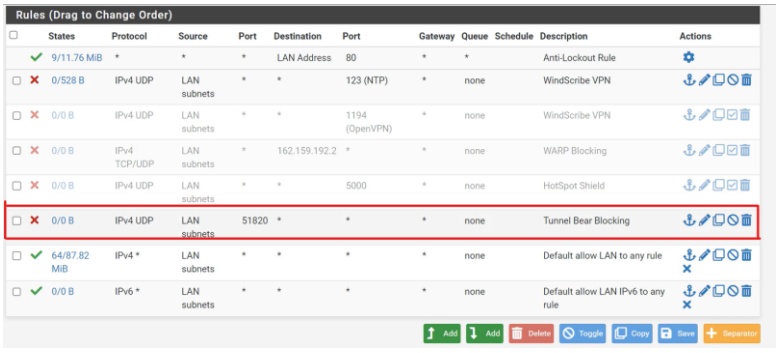


Figure 44: PfSense - Tunnel Bear Block

The rules are as follows:

Step	Protocol	Source Port	Destination Port	Action	Observation
1	TCP	Any	Any	Default allow LAN to any rule.	All services work smoothly
2	UDP	4500	4500	Block the entire traffic which work on 4500 ports on both ends.	Tunnel Bear Connection Failed

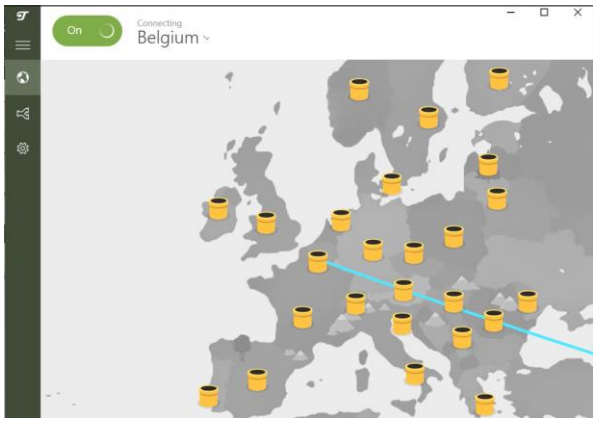


Figure 45: Blocked Tunnel Bear

10.2.2 Blocking Cloudflare WARP

While performing all the above-mentioned activities, certain results are deduced. However, during the monitoring of network packets for the establishment of connections involving Cloudflare WARP, it was noticed that the application connects to IP address 162.159.192.2. Unlike Tunnel Bear, which consistently utilizes port 4500, Cloudflare WARP demonstrates flexibility in port usage, connecting to different ports during its operation.

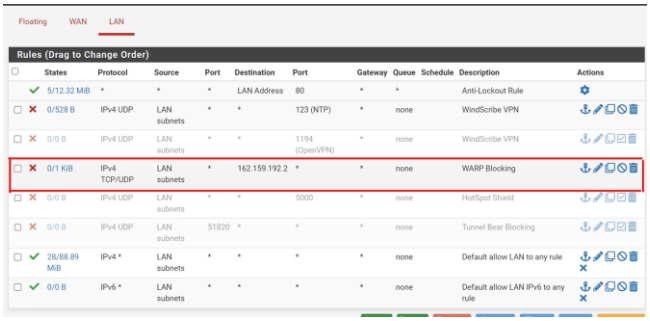


Figure 46: PfSense - WARP Blocking

Figure 47: Blocked Cloudflare WARP

Therefore, we utilize the PfSense Firewall to block any traffic connection Figure 46 from Lan subnet to 162.159.192.2 destination on any ports. The firewall rules are as follows:

Step	Protocol	Source Port	Destination Port	Action	Observation
1	TCP	Any	Any	Default allow LAN to any rule.	All services work smoothly
2	UDP /TCP	Any	Any	Block the entire traffic moving to 162.159.192.2 port.	Cloudflare Warp connection Failed

10.2.3 Other VPN Blocking

While conducting network analysis for Other VPNs services, it was observed that these VPNs exhibit distinct patterns in their network traffic behavior. Although they may not have the same port or IP address as Tunnel Bear or Cloudflare WARP, they still possess identifiable characteristics that can be leveraged for blocking as shown in Table 9.

To block all Other VPN services, we can adopt a strategy similar to the approach used for Tunnel Bear and Cloudflare WARP. Once these patterns are identified, firewall rules can be configured on PfSense to block traffic associated with services. This involves blocking specific ports, IP addresses, or even applying more advanced filtering techniques based on packet inspection and analysis.

VPN	Port No.	Packets Length	IP Ranges
Proton VPN	51820, 1224, 88, 4500, 4569, 500, 1194, 5060, 7770, 443, 8443	122, 138, 154, 1306, 1354, 1482, 127, 128, 1344, 1428, 1433, 1460, 1354, 1358, 1448, 1507	Annex A
Hide.me VPN	432, 443, 30xx, 4000xx (with 444 on User A)	122, 138, 170, 1050, 1354, 1466, 174, 1358, 1486, 126, 1344, 1346, 1458, 1499, 1514, 54, 66, 1296, 1362, 1406	Annex A
Windscribe VPN	80, 53, 123, 1194, 65142, 54784, 587, 21,22, 3306, 54786, 1194, 8443, 443, 4500	122, 138, 154, 1354, 1358, 1494, 118, 1478, 1514, 4250, 60, 1384, 1344, 1392	Annex A
Tunnel-Bear VPN	51820 (with 51820 on User A) and 4500 (with 4500 on User A)	114, 122, 138, 1352, 1354, 1494, 126, 1358, 1486	Annex A
Turbo VPN	443, 8080	56, 66, 1514	Annex A
Cloudflare Warp	2408, 1029, many	74, 114, 126, 651, 1352, 1354	162.159.192.2
Hotspot Shield VPN	5000, 4500 (with 4500 on User A), 443	114, 122, 1318, 1354, 1450, 1494, 126, 142, 222, 398, 1486, 1494, 1044, 1279, 1328, 1389, 1514	Annex A

Table 10: Identification of VPN

10.3 Benefits of Proposed Strategy

The recommended technique presented in this paper has numerous substantial benefits.

- The method captures and analyses network data to understand VPN application behavior, including handshakes, idle states, and user activities like website access and multimedia streaming.
- The technique analyses packet sizes, frequencies, and patterns to differentiate VPN traffic from regular network traffic, even with encryption.
- Continuous network traffic monitoring improves security by proactively detecting possible attacks and abnormalities.
- The study informs network management decisions, including optimizing setups, adopting security measures, and addressing performance concerns with VPN usage.
- The method prioritizes network traffic analysis above decrypting encrypted payloads to preserve user privacy and comply with rules and ethical standards.

Chapter 11

FUTURE WORK AND CONCLUSION

11.1 Conclusion:

Finally, our study offers a complete technique to decoding encrypted VPN data as well as relevant information on how various VPN companies operate. We demonstrated the effectiveness of our strategy in understanding VPN application behavior by meticulously analyzing idle states, handshake protocols, and user behaviors across many VPN platforms. We increase network security and management by implementing firewall rules that prevent specific ports and IP addresses connected to VPN services. This enables the proactive detection of potential dangers and informed decision-making.

Despite its limitations, our method provides a solid foundation for future study in network security and digital forensics. Future research efforts may employ machine learning techniques to adapt to changing encryption standards and analyze messages more effectively. By constantly improving and modifying our technique, we can stay ahead of emerging challenges in the realm of encrypted VPN traffic analysis.

11.2 Future Work:

The analysis of encrypted payloads in VPN traffic is the main source of constraint for this study. The capacity to completely comprehend the content and purpose of connections is restricted by the encrypted nature of payload data, even though packet sizes, ports, and patterns provide insights about VPN behavior. The firewall rules and network settings that are put in place will have a significant impact on how well the suggested approach works. The effectiveness of the proposed strategy heavily relies on the specific network configuration and firewall rules implemented. Variations in network setups may result in different traffic patterns and behaviors, potentially impacting the generalizability of findings. Scalability and performance concerns may also surface as network traffic volumes increase, scalability and performance issues may arise in capturing, storing, and analyzing large datasets.

The version dependence is the main drawback. Forensics analysis of VPN applications is dependent on the specific version of these VPN applications. As a result, the suggested work also depends on the version. If a new version of the VPN is published that fundamentally changes the behavior or structure of the network, the network traffic patterns of the VPN may change and deviate from the conclusions presented in this study. We will investigate methods in the future to leverage machine learning algorithms for traffic analysis to retain efficient traffic analysis capabilities while keeping up with changing encryption standards. This might improve our capacity to precisely recognize and categorize VPN behaviors. To generalize the suggested approach and perhaps make it portable to future application versions, we will seek to add P4 language to give more comprehensive DPI based on the P4 Switch with adjustable properties, such as but not limited to ports and related IPs, etc.

Chapter 12

REFERENCES

[1] Arafune, M., Goswami, B., Kulkarni, M., Venkatachalam, N., & Asadollahi, S. (2023). Lightweight anti ddos security tool: Edge level filtering in SDN using P4. 2023 Fifth International Conference on Electrical, Computer and Communication Technologies (ICECCT). https://doi.org/10.1109/icecct56650.2023.10179747

[2] Goel, A., Kashyap, A., Reddy, B. D., Kaushik, R., Nagasundari, S., & Honnavali, P. B. (2022). Detection of VPN network traffic. *2022 IEEE Delhi Section Conference (DELCON)*. https://doi.org/10.1109/delcon54057.2022.9753621

[3] Afzal, A., Hussain, M., Saleem, S., Shahzad, M. K., Ho, A. T., & Jung, K.-H. (2021). Encrypted network traffic analysis of secure instant messaging application: A case study of signal messenger app. *Applied Sciences*, *11*(17), 7789. https://doi.org/10.3390/app11177789

[4] Papadogiannaki, E., & Ioannidis, S. (2021). A survey on encrypted network traffic analysis applications, techniques, and countermeasures. *ACM Computing Surveys*, *54*(6), 1–35. https://doi.org/10.1145/3457904

[5] Draper-Gil, G., Lashkari, A. H., Mamun, M. S., & A. Ghorbani, A. (2016). Characterization of encrypted and VPN traffic using time-related features. *Proceedings of the 2nd International Conference on Information Systems Security and Privacy*. https://doi.org/10.5220/0005740704070414

Annex

Note: In IPs “x” is any number from 0-9.

A. Proton VPN

103.125.235.**xx**	185.107.80.**xxx**	212.38.97.**xxx**	89.187.177.**50-100**	93.190.138.**150-200**
109.236.81.**xxx**	185.159.156.**xxx**	212.8.243.**xxx**	89.187.179.**xxx**	93.190.140.**xxx**
138.199.21.**xxx**	185.159.158.**xxx**	212.8.253.**xxx**	89.187.180.**xxx**	
138.199.22.**xxx**	185.177.124.**xxx**	217.138.206.**xxx**	89.187.185.**xxx**	
138.199.50.**xxx**	185.177.125.**xxx**	217.23.3.**xxx**	89.38.97.**xxx**	
138.199.52.**xxx**	185.177.126.**xxx**	37.120.217.**xxx**	89.38.99.**xxx**	
138.199.7.**xxx**	185.182.193.**xxx**	37.120.244.**xxx**	89.39.104.**xxx**	
143.244.44.**xxx**	185.183.33.**xxx**	37.19.200.**xxx**	89.39.106.xxx	
146.70.147.**xxx**	185.183.34.**xxx**	37.19.201.**xxx**	89.39.107.**xxx**	
146.70.174.**xxx**	185.230.126.**xx**	37.19.205.**xxx**	89.45.4.**xxx**	
146.70.202.**xxx**	185.236.200.**xxx**	37.19.221.**xxx**	93.190.138.**150-200**	
146.70.45.**xxx**	190.2.130.**xxx**	38.132.103.**xxx**	93.190.140.**xxx**	
149.34.244.**xxx**	190.2.133.**xxx**	45.14.71.**xxx**	93.190.138.**xxx**	
156.146.51.**xxx**	193.148.18.**xxx**	45.87.214.**xxx**	93.190.140.**xxx**	
156.146.54.**xxx**	192.2.132.**xxx**	45.89.173.**xxx**	93.190.138.**xxx**	
165.150.169.**xxx**	195.181.162.**xxx**	46.166.182.**xxx**	93.190.140.**xxx**	

169.150.196.**xxx**	195.181.163.**xxx**	77.247.178.**xxx**	93.190.138.**xxx**	
169.150.218.**xxx**	198.148.18.**xxx**	79.110.55.**xxx**	93.190.140.**xxx**	
185.107.56.**xxx**	212.102.35.**xxx**	87.249.134.**xxx**	93.190.138.**xxx**	
185.107.57.xxx	212.102.51.**xxx**	89.187.170.**xxx**	93.190.140.**xxx**	

B. Windscribe VPN

103.10.197.xxx	143.244.44.xxx	162.222.198.xxx	193.27.14.xxx	212.102.xxx.xxx
104.129.xxx.xxx	146.70.xxx.xxx	169.150.19x.xxx	194.59.249.xxx	212.103.xxx.xxx
104.223.xxx.xxx	149.102.229.xxx	172.98.68.xxx	198.55.126.xxx	217.138.25x.xxx
104.233.xxx.xxx	149.36.xx.xxx	173.44.36.xxx	198.8.85.xxx	223.123.88.xxx
104.245.146.xxx	149.50.208.xxx	185.120.147.xxx	198.96.95.xxx	23.105.1xx.xxx
107.150.xx.xxx	149.57.xx.xxx	185.156.173.xxx	2.58.44.xxx	27.122.1x.xxx
107.161.86.xxx	154.47.26.xxx	185.189.113.xxx	204.44.122.xxx	37.120.2xx.xxx
107.7.60.xxx	155.94.xxx.xxx	185.217.68.xxx	207.244.91.xxx	45.87.21x.xxx
138.199.xx.xxx	155.97.217.xxx	185.236.200.xxx	208.77.22.xxx	68.235.3x.xxx
139.199.47.xxx	161.129.70.xxx	185.253.97.xxx	208.78.41.xxx	68.235.4x.xxx
68.174.103.xxx	71.19.25x.xxx	77.81.136.xxx	84.17.43.xxx	84.17.50.xxx
86.106.87.xxx	89.41.26.xxx	89.47.62.xxx	91.2xx.xxx.xxx	92.119.117.xxx

C. Tunnel-Bear VPN

103.50.33.xxx	134.209.xx.xxx	143.110.1xx.xxx	159.2xx.xxx.xxx	165.2xx.xxx.xxx
104.131.xx.xxx	137.184.xx.xxx	143.198.26.xxx	159.65.xxx.xxx	167.172.xxx.xxx
104.248.162.xxx	138.197.xxx.xxx	146.190.xxx.xxx	159.89.xxx.xxx	167.71.xxx.xxx
107.170.xx.xxx	139.180.xxx.xxx	149.28.164.xxx	161.35.16x.xxx	174.138.xxx.xxx
134.122.xx.xxx	139.59.xxx.xxx	157.2xx.xxx.xxx	162.243.xxx.xxx	188.166.xxx.xxx
192.241.xxx.xxx	209.97.xxx.xxx	45.55.xxx.xxx	65.225.56.xxx	68.183.58.xxx
200.25.50.xxx	37.120.234.xxx	46.101.xxx.xxx	67.205.185.xxx	106.189.xx.xxx

D. Hide.me VPN

146.70.106.xx	31.13.189.xxx
146.70.118.xx	37.120.192.xxx
146.70.128.xx	45.141.152.xxx
185.216.33.xxx	45.152.18x.xxx
217.138.194.xxx	72.10.160.xxx
217.138.195.xxx	72.10.162.xxx
217.138.208.xxx	95.174.67.xxx
217.138.215.xxx	

E. Turbo VPN

134.209.212.xxx
139.99.90.xxx
157.245.218.xxx
159.89.180.xxx
162.243.x.xxx
165.227.xxx.xxx
51.79.xxx.xxx

F. Hotspot Shield VPN

103.105.164.xxx	198.145.2xx.xxx
103.216.198.xxx	204.14.xx.xxx
104.232.xxx.xxx	217.151.xxx.xxx
107.182.231.xxx	23.249.xxx.xxx
146.70.1xx.xxx	45.56.1xx.xxx

173.244.217.xxx	64.141.xx.xxx
185.208.152.xxx	89.11x.xxx.xxx
192.119.160.xxx	92.119.xxx
20.190.147.xxx	13.38.12x.xxx
63.141.48.xx	51.44.41.xxx

G. Cloudflare WARP

162.159.192.2