CONTENTS

LIST OF USED ABBREVIATIONS

AI - Artificial Intelligence

DPI - Deep Packet Inspection

GFW - Great Firewall of China

HTTPS - Hypertext Transfer Protocol Secure

IP - Internet Protocol

ISP - Internet Service Provider

ML - Machine Learning

NAT - Network Address Translation

SSH - Secure Shell

SSL - Secure Sockets Layer

TCB - TCP Control Block

TCP - Transmission Control Protocol

TCP - Transmission Control Protocol

TLS - Transport Security Layer

UDP - User Datagram Protocol

VM - Virtual Machine

VPN - Virtual Private Network

INTRODUCTION

In an time when digital communication plays a fairly important role in modern socienty, ensuring the confidentiality and integrity of network communications has become crucial. The arrival of virtual private networks (VPNs) as a way of securing and privatizing digital communications on the publicly used Internet has been a significant development in this direction. Because of the ever growing popularity of virtual private networks, government agencies and corporate network administrators have ever so increased their efforts to identify and block these encrypted connections. The result has been, what one may call, an arms race between organizations seeking to monitor, control, or censor Internet traffic and VPN service providers competing for the privacy of their users. The implementation of VPN obfuscation, an advanced collection of technologies capable of disguising VPN traffic as routine Internet operations, is central to this dispute. This obfuscation allows free access to the global digital landscape.

Initially, VPNs were implemented as just a way to establish secure network connections across the insecure infrastructure of the public Internet. However, the use of VPN technology eventually grown ouside of the scope of corporate applications. Privacy oriented users are increasingly adopting virtual private networks for their own benefit, due to the potential for enhanced privacy, circumvention of surveillance, circumvention of content location based restrictions, and protection against cyber threats. In response, efforts to undermine the effectiveness of virtual private networks have increased. These initiatives include corporate firewall policies, government censorship initiatives such as China's Great Firewall, and anti-VPN measures implemented by content providers (El-Maghraby et al. 2017).

The gradual and yet noticable proliferation of anti-VPN initiatives has sparked the emergence of a complex area of cybersecurity that uses data packet signatures to distinguish VPN connections. In response, the VPN obfuscation movement has developed a number of countermeasures. As a recent example, we have the Geneva algorithm which employs genetic algorithms to generate dynamic, packet- manipulation-based evasion strategies, providing a new take on Deep Packet Inspection (DPI) evasion (Bock et al. 2019).

Steganography can also be utilized in VPN obfuscation. This method provides a way around DPI systems by embedding VPN traffic inside regular data streams. Steganography enables regular web traffic, like audio or video streams, to conceal VPN packets, making it difficult for DPI tools to discern between encrypted and regular traffic (Kundur & Ahsan 2003). The effectiveness and subtlety of this method are assessed, offering insights into its limitations and useful applications in different network environments.

Another way of obfuscating VPN traffic is through the application of port switching. To avoid detection, a VPN server is configured to use non-standard ports, which can assist in getting

around simple DPI systems that keep an eye on popular VPN ports.

Another way of providing VPN obfuscation is the integration of encrypted proxy tunneling software like Shadowsocks and Stunnel. It can be used to make VPN traffic look like HTTPS traffic (Clowwindy et al. n.d.).

All of these methods, excluding port switching, bring with them a trade-off of in connection speed and throughput, though of varying degrees.

The thesis begins with an overview of VPN technology and then describes the various use cases for VPNs as well as the fundamental principles underlying their operation. It then describes the techniques used to detect VPN traffic. The following is a comprehensive examination of various obfuscation methods and their adaptation to evolving VPN traffic detection strategies. The goal of this analysis is to provide insight into VPN detection methodologies and obfuscation techniques through experimentation, literature analysis, qualitative assessment, and technical proficiency in the process developing a methodology by which one can consider selecting an obfuscation method in real-life scenarios. The bulk of the experimentation was conducted utilizing a virtual networking topology with the prime VPN being OpenVPN and ndpi being used as the DPI system.

1. LITERATURE

For the purposes of gaining insight into the topic of VPN obfuscation an extensive dive and analysis was conducted into the topic of VPNs, DPI and protocol obfuscation. Many related sources and research publications were found, though it may be noted that none of the research results found were of a similar nature to my own topic. Much of the research, was of a more focused nature, typically centering around a more narrow topic. None of them took a comparative approach to the VPN obfuscation methods. In addition, they approached the topic from a different perspective. The perspective of general data hiding, not taking VPNs into consideration, some examples being Bock et al. (2019), Kundur & Ahsan (2003) and Wang et al. (2020) where there is no explicit mention of VPNs. In addition, many research publications were aimed towards the opposite, with goal of developing progressively new ways of detecting VPN traffic instead of hiding it, examples of thing being Chen & Lin (2021), Ghatikar & Sai (2022) and Naidu & Jha (2023).

1.1. Search Strategy

The process of finding source of literature was incremental and partially iterative in its nature. At first, a list of topic specific terms and keywords was compiled: Deep packet inspection, network traffic analysis, censorship circumvention. With these keywords, a number of initial sources were found, those most prominently included: Bock et al. (2019), Wang et al. (2014) and El-Maghraby et al. (2017).

Based on the found literature sources, a more clear understanding of the topic and its terminology was garnered. Accordingly, an extended list terms and keyword was compiled which now included: steganography, packet fragmentation, traffic manipulation, network security, Obfsproxy. This process was then again repeated until an adequate understanding of the topic was achieved.

1.2. Literature Repository Selection

In order to find the literature a number of digital libraries were utilized based on the degree of relevance in relation to the topics of computer science, networking and software engineering. The most prominent of those repositoiries being:

- Research Gate
- IEEE Xplore
- ArXiv

- ACM Digital Library
- Springer Link
- Dergi Park

In some exceptional cases, more unoffical sources were used, for example, jouralist articles or website documentation. This applied for cases like Clowwindy et al. (n.d.) and Network Solutions LLC (2021) where the most reliable source of information for how Shadowsocks and OpenVPN work came from website documentation.

1.3. Exclusion criteria

For the systematic review of literature relating to VPN obfuscation, Deep Packet Inspection, and protocol obfuscation, certain exclusion criterias were employed to ensure the relevance and validity of the chosen sources. These criteria were used in filtering out literature that did not directly contribute to the understanding of the topic or meet the requirements. The used exclusion criteria are as so:

- **Non-Academic Sources:** Sources lacking peer-review or not published in recognized academic journals or conferences were excluded to the utmost degree. This includes blog posts, non-reviewed whitepapers, and informal publications. Though still, some non-academic sources were used when absolutely needed.
- **Irrelevant Topics:** Literature that did not directly address VPNs, VPN obfuscation, DPI, or protocol obfuscation was excluded. This encompasses studies focused solely on broader topics of corporate network security or general VPN usage. Though, to be clear, papers that provide in some way a detailed technical explanation on how protocols, VPNs or DPI were, in certain cases, were included.
- **Language Constraints:** Sources not available in English were excluded.
- **Duplicate Studies:** In cases where multiple publications reporting the same information or research, only the most comprehensive or recent version was included.
- **Geographical Irrelevance:** Studies focusing on region specific VPN usage or censorship practices, which are not applicable or relevant to the broader context of global VPN obfuscation, were excluded.
- **Technical Inapplicability:** Research that focuses on obsolete or rarely used VPN protocols or technologies was excluded, to concentrate on methods and findings relevant to current practices.

These exclusion criteria were applied to make sure the collection of literature was focused and relevant. The criteria were developed to strike a balance between inclusiveness and specificity, therefore providing a comprehensive overview of the current state of research in VPN obfuscation, DPI, and protocol obfuscation.

1.4. Found literature

As mentioned before, a number of sources relating to VPNs, protocol obfuscation and DPI were successfully found. Though all of these sources differ in their goal from this thesis. None of them perform a wide comparative review and analysis for VPN obfuscation methods. The found literature is used as a source of information, from which a comprehensive overview of VPN obfuscation is synthesized

1.4.1. VPN related literature

The found literature addressing VPNs is quite diverse. Covering various topics from technical implementation to traffic identification and obfuscation. Here is a general overview of the key literature in this area:

- **OpenVPN Traffic Identification:** Pang et al. (2013) in their work titled ”OpenVPN Traffic Identification Using Traffic Fingerprints and Statistical Characteristics” provides a good general overview on how OpenVPN traffic could be identified. This information is critical towards understanding the characteristics of VPN traffic and how DPI can go about detecting it.

- **SSL VPN Advanteges:** The benefits of SSL based VPNs is discussed in-depth by Sun (2011) and this work highlights certain features and benefits of SSL VPNs, which are critical for secure and remote access.

- **General VPN overview:** a basic definition and understanding of VPNs is provided by Ferguson & Huston (1998) helping to establish a baseline border between data obfuscation and VPN obfuscation.

- **Tor and VPNs:** The intersection of Tor and VPNs is discussed in the chapter ”Tor Relays, Bridges, and Obfsproxy” by Loshin (2013). The chapter provides a fairly well-rounded understanding of how Tor and VPNs can be used together for enhanced anonymity and privacy.

1.4.2. DPI related literature

DPI is an important part of corporate networking and traffic analysis and several sources have been found that provide an insightful overview of this technology. Here are some of the key highlight in terms of found literature:

- **Deep Packet Inspection Survey:** In the paper by El-Maghraby et al. (2017) ”A survey on deep packet inspection”, a comprehensive look is provided for DPI technologies, expositing fundamental facts about DPI such as on what OSI layer they operate and what mechanism are used to inspect packet contents, such as regular expression.

- **DPI and Traffic Manipulation:** Yoo & Ahmed (2019) in ”Control Logic Injection Attacks on Industrial Control Systems” and Wang et al. (2017) in ”Your State is Not Mine: A Closer Look at Evading Stateful Internet Censorship”, study DPI in terms of traffic manipulation and evasion methods. These papers provide insights into how DPI can be circumvented or exploited, which is important for understanding its limitations and possible vulnerabilities.

- **DPI and Machine Learning:** Trivedi & Patel (2016) postulate the integration of machine learning into DPI systems and present an new approach for DPI systems enhanced by machine learning which potentially describes the future of DPI and how they will further evolve.

- **DPI and SSL Inspection:** Chakraborty et al. (2022) assess the DPI system of network traffic, including anomaly detection. This study is important for understanding how DPI is adapting to modern encrypted traffic.

These chosen studies give a broad look at DPI, looking at its scientific roots, its uses in different areas, and new problems that are coming up. To fully understand DPI in the bigger picture of VPN obfuscation and protocol obfuscation, which are the main ideas of this thesis, you need to understand these aspects.

1.4.3. Protocol obfuscation related literature

Protocol obfuscation refers to the process of using various methods and techniques to hide the true identity of the protocol being used for communication between network devices. This is most important area of study pertaining to my thesis and as such many different sources and papers were found for it:

- **Geneva: Evolving Censorship Evasion Strategies:** Bock et al. (2019) provide an modern and innovative new method for automated DPI evasion by utilizing genetic algorithms for developing a unique strategy that could be used to bypass any DPI system in place.

- **Practical Internet Steganography:** Kundur & Ahsan (2003) presents steganography as a means of obfuscating communication protocols, an important aspect of circumventing DPI. The paper itself talks about utilizing redundant IP header fields for carrying the conent data.
- **SymTCP: Automated Discrepancy Discovery:** Wang et al. (2020) discuss a method for evading DPI through the discovery and use of discrepancies/inconsistencies in TCP implementations. Mainly, this study helps in understanding how protocol behaviors can be manipulated to evade detection.
- **Hiding Data in the OSI Network Model:** Handel & Sandford (1996) discuss the idea of data hiding at various layers of the OSI model and their corresponding protocols which in some cases have field headers that are unnecessary in certain situations. This research provides foundational knowledge on the versatility of obfuscation techniques across different network layers.

All together, these studies give a full picture of protocol deception methods and how they are used. They show how protocol obfuscation is changing because DPI and filtering technologies are getting better. This thesis needs to put these methods in the bigger picture of VPN obfuscation, so understanding how they work is very important.

2. BACKGROUND

2.1. Virtual Private Networks

2.1.1. Definition and Purpose of VPNs

A VPN is a fairly complicated communication system that enables safe and private connections within a public network infrastructure, such as the internet by using secure protocols and encryption. Its purpose is to facilitate the secure and private transmission of data over a shared network, restricting access to only authorized users within a certain community of interest.(Ferguson & Huston 1998) VPNs ensure anonymity by utilizing techniques such as tunneling and encryption. This allows for the creation of a virtual network that makes use of the existing infrastructure of the public network, while maintaining high levels of security and privacy. A VPN serves the purpose of facilitating safe and confidential communication, which is particularly crucial for the transmission of sensitive data and distant connectivity to private networks.

2.1.2. Types of VPNs

For lone users who must connect to a private network from a distance, **remote access VPNs** are made. It is frequently used by people who work from home and who need safe/secure remote access to company or personal network resources. The user's device's VPN client software creates a secure connection to a VPN server, which then provides access to the private network. With this configuration, data transported over the internet is supposed to be encrypted, protecting its integrity and privacy.

Site-to-Site VPNs, also known as Router-to-Router VPNs, are fairly often used by large enteprises with offices that are dispersed in different places. The goal of these VPNs is to link entire networks together. For example, a company connecting its office network in New York to its office network in London may be able to do so by using a Site-to-Site VPN. This kind of VPN can be further divided into two categories: Extranet-Based (which connects to sites outside the organization, such as partners or vendors) and Intranet-Based (which connects to numerous sites within the same organization). (Rathore et al. 2009)

Personal VPN services are sold to people by outside companies and service providers. These services help people protect their privacy and safety by encrypting their internet data and hiding their IP address from other parties, like ISPs. People often use personal VPNs to get around location based content limits, and to stay anonymous online, and, of course, to keep their data safe when they use public Wi-Fi networks.

A **Mobile VPN** is a VPN that maintains a consistent connection as the user moves and changes network connections, making it very comfortable for professionals who frequently switch

between Wi-Fi and cellular data networks. And unlike traditional VPNs, which can drop connections during such transitions, Mobile VPNs ensure uninterrupted connectivity and application session persistence so that, for example, a used logged into a website is not suddenly disconnected. (Alshalan et al. 2016)

Hardware VPNs offer a separate standalone device with dedicated processors to handle VPN operations. This type offers also robust security, naturally high performance, and ease of maintenance, making it fairly suitable for large businesses. Although, they can be more expensive than software-based solutions and could require more technical knowledge and expertise for setup and management. A valid example of this may be a pure hardware implementation of WireGuard on FPGA. (Liu et al. 2023)

Secure Sockets Layer (SSL) VPNs are VPNs that provide safe remote access to an organization's internal network and applications, in a similar fashion to the before mentioned ones, but unlike traditional VPNs that require installing specific client software, SSL VPNs can be used from a standard web browser, providing a more flexible/robust solution for users. They are especially useful for providing access to web applications and services. (Sun 2011)

2.1.3. VPN Protocols and Standards

Different protocols and standards, each with their specific functionality and security mechanisms, could be represented as the core of VPN technology. Understanding VPN protocol obfuscation and Deep Packet Inspection evasion methods requires a solid understanding of these protocols:

- **OpenVPN**: An very configurable and robust protocol. One of the things its known fore are its open-source implementations of security and performance technologies, including the OpenSSL encryption library. OpenVPN is a highly adaptable software that supports a large range of encryption algorithms and can circumvent firewalls. It is an ideal candidate for research into obfuscation and DPI evasion techniques due to its adaptability.

- **IPSec/IKEv2**: with its acclaimed speed and reliability, Internet Protocol Security (IPSec) and Internet Key Exchange version 2 (IKEv2) provide strong security. However, it is less suitable for in-depth obfuscation research because to its intricacy and the need for a third-party client on some platforms. (Matalgah et al. 2002)

- **L2TP/IPSec**: Due to how IPSec and Layer 2 Tunneling Protocol (L2TP) work complementary to each other, many devices use them together. But because L2TP/IPSec depend on fixed ports, it is less suitable for obfuscation research because it is more prone to DPI and simpler to stop.

- **PPTP**: One of the first VPN methods was Point-to-Point Tunneling Protocol (PPTP). PPTP is not recommended for secure communications because it has known security gaps and encryption that is not very strong. This makes it less relevant for current study into obfuscation and DPI evasion.

- **SSTP**: Secure Socket Tunneling Protocol is well-known for its capacity to circumvent the majority of firewalls. However, because it is Microsoft's proprietary software, it does not have the openness and public inspection that are required for conducting in-depth study on obfuscation. (Kim et al. 2011)

- **WireGuard**: WireGuard, a more recent addition, provides both simplicity and high-speed performance. Although it shows potential, its relative newness has limited its testing in various circumstances, especially in situations involving obfuscation. (Lipp et al. 2019)

Taking these things into account, OpenVPN becomes the main topic of experimentation into hiding VPN protocols and getting around DPI. Because it is open-source, it can be studied and configured in a lot of detail, which is necessary for studying effective obfuscation methods, in addition OpenVPN is widely used and supports many different types of encryption, it is also a good choice for this study.

2.1.4. Security Mechanisms in VPNs and DPI challenges

VPN's build safe, encrypted tunnels that shield users from all kinds of risks, like as censorship, data theft, and spying. However, more sophisticated Deep Packet Inspection (DPI) methods are posing a threat to VPN efficacy. We examine the main VPN security features below, along with how DPI may compromise their efficacy.

Encryption

- **Function**: Encryption is the most important part of VPN security. It changes data into a coded form that can only be read by someone with the right decoding key. AES (Advanced Encryption Standard) and TLS (Transport Layer Security) are two common types of encryption.

- **DPI challenge**: DPI can look at patterns of encrypted data to find VPN use. DPI can't decrypt the data, but it might be able to slow down or stop encrypted traffic, especially if the encryption protocol is well known and easy to spot.

Tunneling Protocols

- **Function**: OpenVPN, L2TP/IPsec, and IKEv2 are standard cases of tunneling protocols that are used for the goal of encapsulating and securely transmitting data over the internet networks. In addition to ensuring that data is kept private and unaltered during the transmission process, they define the manner in which packets are transmitted.
- **DPI challenge**: DPI solutions have the capability to detect and classify network traffic originating from widely used tunneling protocols. DPI can identify and impede VPN traffic by analyzing packet headers and sizes, particularly when conventional ports and protocols are employed.

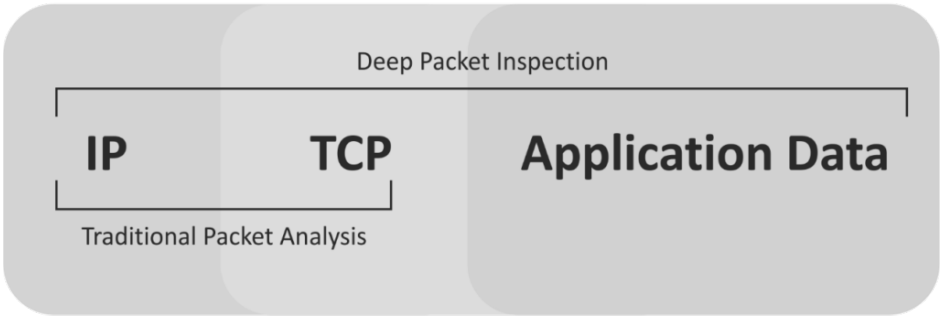
Authentication

- **Function**: VPNs utilize authentication methods to authenticate the identities of users and devices. This procedure frequently includes the use of certificates or credentials to guarantee that only authorized users have access to the VPN.
- **DPI challenge**: DPI doesn't directly affect authentication, but it can prevent users from connecting to VPN servers, which negates the purpose of authentication.

IP Masking

- **Function**: By substituting the IP address of the VPN server for the user's original IP address, VPNs disguise their identity. This aids in preserving anonymity and eschewing limitations based on location.
- **DPI challenge**: It is possible to distinguish between VPN traffic and ordinary traffic using advanced DPI techniques, even when IP masking is in use. This opens the door to the prospect of excluding or restricting VPN traffic depending on its unique properties.

2.2. Deep packet inspection



2.1. Figure. **Inspection area of DPI**

Deep Packet Inspection is a system for network packet filtering. It examines the data part (and often the header) of a packet as it passes an inspection point. As can be seen on Figure 2.1, DPI examines the packet's data payload in depth as opposed to conventional packet filtering, which simply looks at the header section, allowing for more complex packet decision-making. DPI can detect, identify, classify, reroute, or prevent packets containing particular data or code payloads thanks to this sophisticated technique.

Due to its great versatility, DPI technology is employed in many different contexts, such as traffic management, censorship, eavesdropping, and network security. It is an essential tool for internet service providers and regulatory bodies since it can identify and manage apps using the network even in the face of encryption and port-hopping schemes.

2.2.1. DPI in Relation to VPNs

DPI is important when it comes to VPNs. In order to protect privacy and get around censorship or geographical limitations, VPNs usually encrypt data. Nonetheless, DPI can be used to examine encrypted packet metadata, including timing, size, destination, and protocol. This makes it possible to identify VPN activity even when the content is encrypted.

Certain DPI techniques can detect the signatures of the encryption protocols used by VPNs, allowing them to differentiate between conventional and VPN traffic. This capability is essential in situations when the use of VPNs is prohibited or closely watched, like in nations with severe internet censorship.

2.2.2. DPI's Challenges for VPNs

VPN technologies have substantial hurdles due to the efficacy of DPI:

- **Detection and Blocking**: If Advanced DPI detects that a VPN is being used, it has the ability to either block or limit data, making the VPN useless for getting around censorship or geographical restrictions.
- **Avoiding Encryption**: Although DPI is unable to decrypt data, it can identify and potentially block VPN services based on educated estimates about the encrypted traffic's nature.
- **Adaptive Countermeasures**: VPN service providers are compelled to constantly modify and introduce fresh methods to avoid being detected by DPI, resulting in a never-ending game of cat and mouse between VPN providers and DPI-using companies.

2.3. OpenVPN identification

A first impression, may lead one to wonder how DPI systems are capable of differentiating between legitimate encrypted data such as secure HTTPS access to a website and VPN traffic. There are several techniques that DPI can use to identify OpenVPN traffic (Pang et al. 2013):

- **Protocol Signature Identification**: OpenVPN traffic can be recognized by DPI systems by looking for distinctive signatures in the packet headers. This is a conventional method that might work less well for OpenVPN because of its encryption and inconsistent port usage.
- **Heuristic analysis**: In this method, traffic patterns are examined to find traits common to VPN traffic, such as regular timing and packet sizes.
- **IP address and port inspection**: OpenVPN typically uses TCP or UDP port 1194, but it also has the option to use variable ports, which reduces the efficacy of this approach.
- **TLS Handshake Analysis**: VPN traffic, including OpenVPN, can be identified by examining the TLS handshake procedure, paying particular attention to the usage of particular cipher suites or TLS versions.
- **Timing and Packet Size**: DPI can identify consistent timing intervals and packet sizes in OpenVPN traffic.
- **Analysis of Encryption Patterns**: Even though OpenVPN encrypts its traffic, some encryption patterns might still be recognizable.
- **Behavioral Analysis**: VPN usage can be inferred by observing general network behavior, such as persistent, long-lasting connections.

- **Sequence Number and Acknowledgment Number Analysis**: Analyzing these numbers in TCP-based OpenVPN traffic can reveal VPN usage patterns.

- **Opcode Analysis in Packets**: the first 10 packets of a communication can be used as statistical characteristics for early detection of OpenVPN tunnels. By using the five bits of opcode in each packet to classify different types of packets.

These sophisticated methods, which focuses on statistical traits and traffic fingerprints, especially the opcode analysis, are very successful in identifying OpenVPN traffic, with low false positive rates and high accuracy (Pang et al. 2013). In this context, the goal of VPN obfuscation can be described as attepting to bypass detection by trying to render as many as possible of the above mentioned techniques, as ineffectual.

2.4. Packet manipulation

One of the most important tools for the design of hardware implementation for the DPI is the finite state machine. This system design, in certain cases, may be exploited by utilizing packet manipulation techniques like packet fragmentation. (AbuHmed et al. 2008) These attacks are even more relevant in the context of secure network environments, where traditional intrusion detection methods are employed.

In order to avoid detection by DPI systems, the following techniques may be employed:

- **Fragmentation and noise padding**: Harmful data is divided into smaller pieces and mixed in with ”noise” data, which is random or irrelevant data that doesn’t interfere with the payload’s ability to function. The idea is to trick DPI systems into thinking the fragmented packets are harmless, since these systems usually look for patterns in data to identify malicious activity. The attackers can effectively mask the malicious content, preventing it from being identified and prevented, by fragmenting the data and adding noise. DPI systems may find it especially difficult to defeat this strategy since it requires them to precisely reassemble and examine all of the packets in order to find the harmful content that is hidden—a laborious and resource-intensive procedure. (Yoo & Ahmed 2019)

- **TCB Creation Evasion**: The first SYN packet that opens a TCP connection is manipulated in this tactic. A false state is produced in the TCP Control Block (TCB) of the censoring system by changing specific fields in this packet. Actual data packets are able to evade filtering without being noticed because of this manipulation, which makes the censorship monitor think that the link is either nonexistent or has already ended. (Aceto & Pescapé 2015)

- **Data Reassembly Evasion**: This method takes advantage of flaws in the way censorship systems put back together broken TCP segments. When pieces of data are sent purposely out of order or overlapping, it makes it harder for the filtering system to correctly put these pieces back together and analyze them. This means that private information can get through the system that blocks them without being found or stopped. (Khattak et al. 2013)

- **TCB Teardown Evasion**: This method employs deliberately constructed packets, such as RST (reset), RST/ACK (reset acknowledgment), or FIN (finish), to prematurely end the Transmission Control Block in the censoring system. This premature termination results in the censorship system ignoring any subsequent packets from that connection. Nevertheless, the client-server link remains operational, enabling uninterrupted data flow without any intervention from censorship. (Papadogiannaki & Ioannidis 2021)

- **Resync and Desync**: In this approach, particular packets are transmitted in order to initiate a re-synchronization of the censoring system's transmission control block with the specific TCP connection. Following this, an out-of-window sequence packet is transmitted, which ultimately results in the desynchronization of the censorship system from the link that is actually being employed. It is possible for data packets to avoid detection and censorship as a result of this difference. (Wang et al. 2017)

These serve as an example of how packet manipulation methods are capable of bypassing DPI. However, it should be noted that these tactics are, by far, not consistently effective and require a fair amount of manual labor to discover the specific packet sendings order, degree of fragmentation or noise required to bypass the DPI, if at all possible.

2.5. Protocol obfuscation

Protocol obfuscation refers to the strategies used to change or conceal the properties of internet protocols in order to escape detection, censorship, or intervention from third parties. This is especially crucial in situations when internet access is tightly controlled or limited. Obfuscation prevents data packets from being detected by filters or detection systems that depend on established protocol signatures or patterns.

2.5.1. General Methods of Protocol Obfuscation

- **Randomization**: In order to hinder automated systems from identifying patterns, random elements are incorporated into the communication protocol. For example, it may be challenging for Deep Packet Inspection (DPI) to recognize the traffic as being part of a particular protocol when packet sizes or timing change. (Kailanya et al. 2022)

- **Protocol Mimicry**: By transforming the traffic of one protocol into that of another, more widely used protocol (such as HTTPS), users are able to evade filters that target particular types of data. A VPN might make its traffic look like normal HTTPS traffic, for instance. (He & Chen 2016)

- **Header Manipulation**: Systems that use header analysis can't figure out what the data bits are if the header information is changed or encrypted. Changing port numbers or encrypting the whole header are two examples of this. (Munshi 2023)

- **Traffic Pattern Masking**: This technique entails modifying the attributes of traffic, such as its quantity, frequency, or trajectory, in order to conceal its true nature. It can be especially efficient against systems that examine traffic patterns over a period of time. (Iacovazzi & Baiocchi 2014)

2.5.2. Focused methods for practical use

Cosidering the wide range of techniques available, the selection was limited to four specific methods for experimentation:

- **Port Switching**: To get around port-based filtering systems, it's easy but effective to change the port of the VPN server. Numerous DPI systems are set up to either watch or restrict traffic on particular ports that are known to be utilized by VPNs. The VPN traffic can get around these filters by periodically switching the server port. This approach is a sensible option for experimentation because it is simple to use and evaluate. (Crawford 2019)

- **Steganography**: Data concealment is the approach of encoding sensitive information within seemingly innocuous data. It entails hiding the communication within seemingly innocuous data streams, such video or picture streams, within the framework of internet protocols. Systems that aren't built to identify such advanced concealment strategies can be severely harmed by this. To learn how well obfuscation holds up against sophisticated DPI systems, experiments will be conducted with steganography. (Handel & Sandford 1996)

- **Geneva Automated Censorship Evasion**: Geneva is an innovative method that makes use of genetic algorithms to automatically look for new ways to circumvent censorship. Geneva has the capacity to keep one step ahead of censoring systems if it continues to evolve and adapt to the environment with ongoing action. In the subject of protocol obfuscation and censorship evasion, testing Geneva provides a chance to investigate the ways in which automated systems might make a contribution to the field. (Bock et al. 2019)

- **Encryption Tunneling (utilizing stunnel or shadowsocks)**: This solution involves encrypting the communication, making it impossible for DPI devices to examine the data packets' contents. Tools such as stunnel and shadowsocks are designed to conceal and obscure traffic, offering an extra degree of security. This strategy is especially useful in circumstances where DPI systems are powerful and capable of performing in-depth analysis. The use of these tools for experimentation can provide useful information about the effectiveness of encryption-based obfuscation approaches. (Zhao et al. 2018)

To summarize, the discipline of protocol obfuscation provides a variety of methods to avoid censorship and surveillance. The four selected approaches for experimentation - port switching, steganography, Geneva automatic censorship evasion, and encrypted tunneling - possess distinct advantages and cater to various aspects of obfuscation. Their choice to conduct experiments with these subjects is warranted due to their significance in modern situations when internet censorship and surveillance are becoming more advanced.

2.6. Case Studies

There exist numerous real-life examples of DPI being deployed and used on a significant scale. Some of the most notorious being:

- **The Great Firewall of China (GFW)**: China's Great Firewall is a very well-known real-life applications of DPI technology. Among other censoring tools, it uses advanced DPI techniques to identify and restrict VPN connections. VPN service providers frequently need to improve their techniques on a regular basis to avoid being detected. (Wu et al. 2023)

- **Iran's Internet Censorship**: Especially during political turmoil, Iran's government use DPI to monitor and regulate internet traffic, including the detection and blocking of VPN usage. (Bock et al. 2020)

- **Corporate Network Management**: By keeping an eye out for unauthorised VPN use that might go around corporate security measures, DPI is utilised in a business context to guarantee security and compliance.

- **Russia's Telegram Ban**: Using DPI, Russia attempted to ban the messaging service Telegram. Telegram, however, circumvented this restriction using a variety of strategies, including VPNs, illustrating the continuous conflict between DPI implementation and VPN evasion approaches. (Ermoshina & Musiani 2021)

DPI not only poses a serious threat to VPN technologies but also propels the ongoing development of VPN obfuscation strategies including port switching, network steganography, and

sophisticated tunnelling methods like Geneva, Shadowsocks, and Stunnel.

2.7. Early DPI Evasion Methods

Online security and privacy were issues that only a small minority of people worried about in the early days of the internet. But as digital technology developed, so did the techniques for keeping an eye on and filtering online behavior. Deep Packet Inspection started to spread widely and revolutionized the way that institutions and governments could monitor and regulate internet traffic. At first, it was possible to avoid DPI by utilizing networks like Tor or non-standard protocols. However, as DPI technologies advanced, these strategies encountered more difficulties, prompting creative solutions like Tor bridges. (Knapp & Langill 2015)

2.7.1. Rise of DPI and Intial Evasion Tactics

Since Deep packet inspection goes farther than conventional packet filtering by checking the data part (content) of packets as they pass, rather than only the header. Governments and ISPs were able to regulate and control internet traffic with an unparalleled level of precision.

Many people, concerned about their right to privacy and freedom of expression, sought out ways to circumvent DPI as it gained traction. The use of non-standard protocols was one strategy. The original intention of DPI filters was to identify and regulate conventional internet traffic; however, users were able to circumvent these filters by employing less popular or custom-built protocols.

The Tor network was another effective solution against DPI. The Onion Router, or Tor for short, is a system that aims to hide one's online identity. It conceals the user's location and activity from network monitors by rerouting their web traffic through a network of relays. At first, it seemed like a good way to avoid DPI—just connect to Tor and use it. The traffic looked like typical encrypted web traffic.

2.7.2. Advancement of DPI technologies

But things started to change as DPI tools got better. DPI systems got smarter and could look at more protocols and even decrypted traffic trends. They started to notice the signs of Tor activity and the use of non-standard protocols, which made these ways of hiding from surveillance less effective.

Long lists of known IP addresses for Tor relays and entry nodes started to be used. They could stop the Tor network or keep an eye on people who tried to use it with these lists. Advanced DPI systems were also able to recognize Tor's unique traffic patterns, such as its uniform packet

sizes and timing. So, using Tor or other obscure methods was no longer a surefire way to get around DPI.(Dingledine & Mathewson 2006)

In response to these developments in DPI, the Tor Project created additional ways to assist users in maintaining their anonymity while still gaining access to the network. The introduction of Tor bridges was one of the those ways.

Tor bridges are alternate Tor network entry points that are not included in the public Tor directory. Because of this obscurity, they are less likely to be blocked or monitored than conventional Tor relays. Users in countries where the internet is strictly censored could use these bridges to connect to the Tor network without drawing attention to themselves.

Furthermore, the Tor Project launched pluggable transports, a technique meant to change the look of Tor traffic, making it difficult for DPI systems to recognize. Obfsproxy (obfuscated proxy), for example, disguises Tor traffic as conventional, innocuous-looking internet traffic. This makes DPI detection much more difficult. (Loshin 2013)

Meek, another pluggable transport, employs a method known as domain fronting to make Tor traffic appear to be connecting with a prominent website such as Google or Amazon. This strategy takes use of the fact that restricting traffic to these important sites would cause enormous collateral harm, rendering censorship an unfeasible choice. (Fifield et al. 2015)

2.7.3. The Ongoing Cat-and-Mouse Game

The progression of DPI and the accompanying advancements in Tor technology symbolize an ongoing struggle between surveillance parties and proponents of internet privacy. As the technologies for DPI become more sophisticated, the techniques for circumventing them also become increasingly more complex. The implementation of Tor bridges and pluggable transports serving as a valid example of this continuing conflict.

Nevertheless, the usability of these approaches may differ depending on the geographical area and the particular DPI technology being employed. Tor bridges and other obfuscation techniques remain effective for evading detection in certain regions. In some cases, particularly in countries with sufficient government funding for advanced Deep Packet Inspection technologies, even these advanced methods encounter difficulties. (Winter & Lindskog 2012)

2.8. Modern DPI techniques

Recent studies have taken an in-depth examination of the latest techniques employed by the Great Firewall of China to identify and prevent fully encrypted internet traffic. The main techniques are as so (Wu et al. 2023):

- **Heuristic Rules for Exemption**: Based on the assumption that some types of traffic are unlikely to be fully encrypted, the GFW uses a set of heuristic rules to exempt them from blocking. This includes the fraction, position, and maximum contiguous count of ASCII characters; a rough entropy test based on the fraction of bits set; and rules based on common protocol fingerprints.
- **Entropy-Based Blocking**: The GFW denies access to connections based on the entropy of the client's initial TCP payload. As a sign of encrypted data, connections with a specific range of bits set per byte are blocked.
- **Exemption for ASCII Characters**: If the first six bytes are printable, more than half of the bytes are printable, or more than 20 consecutive printable bytes are present, the GFW will not block connections related to ASCII characters in the payload.
- **Protocol Exemptions**: Certain protocols—most notably TLS and HTTP—are specifically spared from blocking because their initial bytes match patterns that these protocols are known to use.
- **Blocking Mechanism**: The client's packets are dropped and never reach the server when the GFW detects encrypted traffic. This prevents further traffic from occurring.
- **Limited Scope and Probabilistic Blocking**: The GFW uses probabilistic blocking, which means that not all connections that satisfy the blocking requirements are obstructed. Blocking is purposefully restricted to particular IP ranges of well-known data centers.
- **UDP Traffic**: Sending UDP datagrams with a random payload does not result in blocking under the new censorship system, which is restricted to TCP.
- **Active Probing System**: Using a payload length-based rule in addition to similar rules, the GFW's active probing system operates in tandem with the traffic analysis system.

These methods highlight the GFW's sophisticated approach to censoring encrypted traffic while allowing standard internet communications. This aids in understanding the censorship mechanism and informs strategies to circumvent these blocks.

2.9. Use of modern DPI techniques

Since modern Deep Packet Inspection techniques have improved over time, effectively combating older methods used to evade such monitoring in the ever-changing world of internet restriction and spying, the capacity of authorities, such as those in charge of the Great Firewall of China,

to identify and stop the many strategies that have historically been used to avoid censorship and surveillance has been greatly improved by recent developments in DPI technology.

- **Advanced Signature Detection**: Older ways of getting around security measures often relied on hiding traffic signatures to look like allowed protocols or hide what kind of data was being sent. These days, DPI systems use more complicated algorithms that can better find trends and oddities in data packets. This includes being able to find small changes in how a protocol works, strange headers, and even patterns that could mean steganography or the use of hidden channels.

- **Behavioral Analysis and Anomaly Detection**: Newer DPI methods use behavioral analysis instead of just looking at the static features of traffic. This lets the systems figure out what normal network behavior looks like and spot changes that might be signs of attempts to get around the system. Anomaly detection algorithms can spot strange traffic patterns, like packet sizes that aren't normal, data amounts that don't match up, or unexpected use of protocols.

- **Machine Learning and AI Integration**: Adding Machine Learning (ML) and Artificial Intelligence (AI) to DPI systems is a big step forward in terms of what they can do. These technologies let the systems learn from the information they handle, which makes them better at finding things over time. In real time, AI algorithms can look at huge amounts of data and find complicated patterns and correlations that humans would never be able to see. (Trivedi & Patel 2016)

- **Encrypted Traffic Analysis**: Many old DPI methods stopped working as much as encryption became more popular as a way to protect user privacy. Modern DPI methods, on the other hand, can look at encrypted data without having to decrypt it first. This is done by looking at statistics, time, and other side-channel data, which lets the authorities guess what kind of data is being encrypted.

- **Active Probing and Response Analysis**: These days' DPI systems don't just watch data; they also probe it and respond to it. These systems can figure out what kind of traffic or application is going on by sending specific packets or requests and looking at the replies. This works especially well against protocols that are made to look like other types of data.(Chakraborty et al. 2022)

All of these DPI advacements have resulted in the slow yet noticable decline of non-modern DPI evasion methods. The most prevalent methods of evading DPI have been proxies and VPNs. The

use of known VPN servers or IP addresses, as well as specific traffic patterns and handshake features, can be detected and blocked by modern DPI techniques.

Even anonymity networks like Tor, which aim to let users browse the internet anonymously and securely, are becoming more and more susceptible to DPI attacks. The unique features of Tor traffic, like packet timing and data transfer sizes, can now be detected by DPI systems, even when Tor employs obfuscation methods like pluggable transports.

Traditional methods of evading DPI also include obfuscation and Secure Shell (SSH) tunneling. Even when communication is disguised or contained within other protocols, contemporary DPI systems can still detect SSH based on its unique handshake and session features. (Dusi et al. 2008)

Methods that masquerade as valid protocols (like HTTPS) in order to conceal harmful or restricted traffic are less successful when faced with current DPI. In order to distinguish between real protocol traffic and imposters, sophisticated algorithms examine time, packet size, and other subtle features. (Pandey et al. 2023)

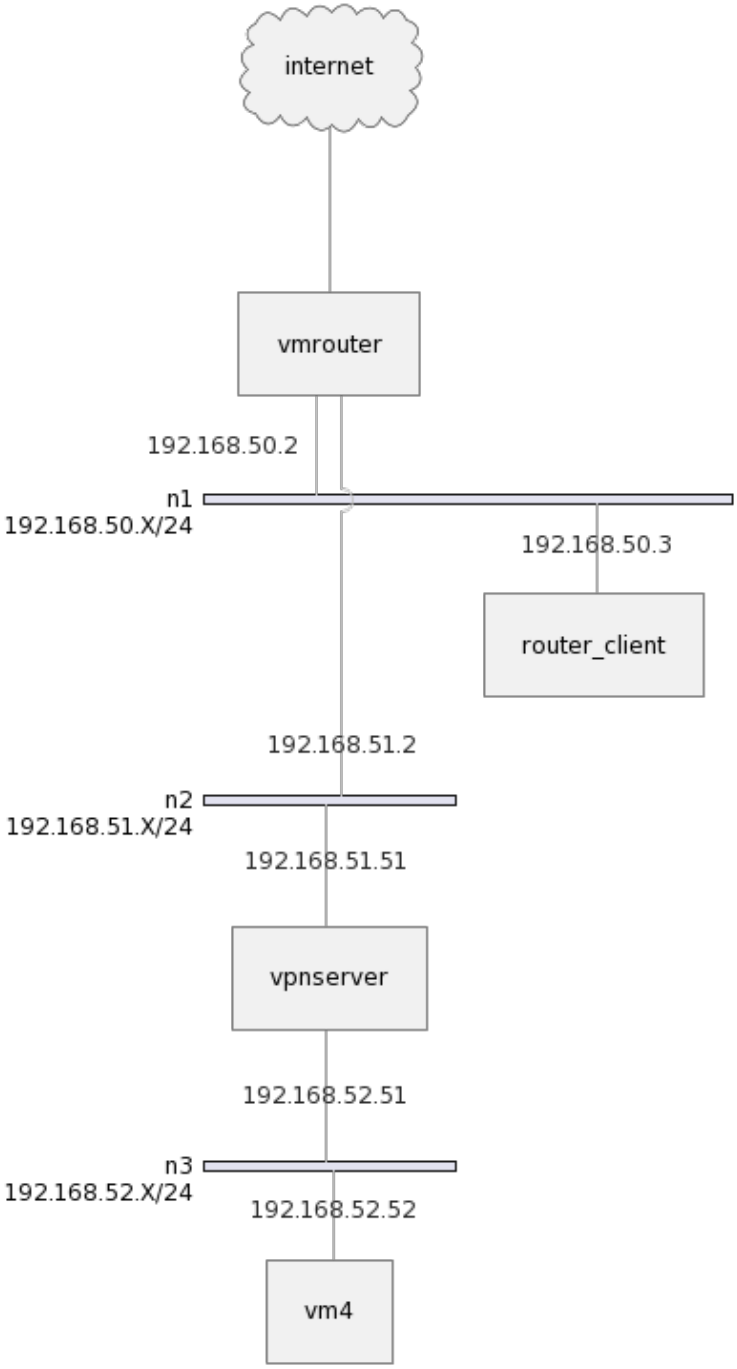
Data packets' genuine nature might be concealed by using traffic fragmentation and padding, which were common techniques in legacy evasion strategies. In order to determine the true type or content of the traffic, modern DPI can reconstruct broken packets by analyzing the padding.

3. EXPERIMENTATION TESTBED

When it comes to VPN encryption methods, building a virtual testbed is important for validating experimention results. This section is about the established testbed, including how it is set up, why certain tools were chosen, and how the network is structured overall. Vagrant and Libvirt were used to set up the testbed. These tools are essential for building and managing virtual machines (VMs) that mimic a real-world network environment so that VPN obfuscation methods can be tested.

An open-source software application, Vagrant allows to create and manage portable virtual environments for software development. It comes with a straightforward command-line client for controlling these settings, and a standard for configuration files to specify the virtual machines needed for a certain project. Because of its versatility, ease of use, and extensive support for several virtualization providers, including Libvirt, Vagrant was chosen for this testbed.

On the other hand, Libvirt presente withing itself a set of tools that can communicate with the virtualization features of newer Linux distributions (and other operating systems). As a management tool for platform virtualization, it is a C toolkit that interacts with Linux's virtualization features. The chosen choice for robust and scalable virtual networking experiments is due to its vast feature set and particularly well-known stability.



3.1. Figure. **Testbed network topology**

The testbed consists of many virtual machines that are configured to imitate the major components of a VPN system as can be seen on Figure 3.1, such as routers, VPN servers, and clients. The Vagrant configuration file defines four different VMs: vmrouter, routerslave, vpnserver, and vm4. Each VM is given unique network interfaces and IP addresses to ensure that it mimics the

common network elements seen in a VPN situation.

The network topology was created to resemble a standard VPN environment with multiple network segments. The vmrouter acts as the primary router, connecting two different subnets. This VM is of course important as it simulates the gateway through which traffic is routed. The second VM, routerslave, is configured to act as a client to the router within the network, providing more realism to the client-server architecture commonly seen in VPN networks.

The VPN server is set up to be the vpnserver VM. This computer is very important to the testbed because it acts like the server side of a VPN. This is where different methods of encrypting data can be used and tried. The last VM, vm4, is a normal client in the network that is used to check how accessible and useful the VPN server is from the point of view of a client.

In some VMs, the network's dual-interface design lets for a complicated routing situation that works like a real VPN, where data traffic often goes through multiple network segments. This set-up is great for trying obfuscation methods that might be affected by different network paths and settings.

3.1. Router Configuration

The vmrouter virtual machine is an important part of the virtual network topology because it is meant to be a router in the Vagrant-based environment. This VM, which was set up using Vagrant with the ”generic/ubuntu1804” box and provisioned for the libvirt provider, serves as a crucial node for network traffic management and routing in your VPN obfuscation method testing setup.

3.1.1. Networking Interface Configuration

1. Table.

Network interfaces configuration on vmrouter

	Interface	IP Address	Subnet Mask	State	MAC Address
0	lo	127.0.0.1	255.0.0.0	UP	00:00:00:00:00:00
1	eth0	192.168.121.216	255.255.255.0	UP	52:54:00:02:6d:7a
2	eth1	192.168.50.2	255.255.255.0	UP	52:54:00:a6:69:a2
3	eth2	192.168.51.2	255.255.255.0	UP	52:54:00:db:30:5d

The IP address table 1 shows that the vmrouter virtual machine has three network interfaces set up. Interfaces like as these are:

- **eth0**: The subnet mask is 255.255.255.0 and the IP address is 192.168.121.216. This port serves as the main interface for communicating with the outside world, connecting the virtual machine to the network ouside the virutal topology.
- **eth1**: Set up with 192.168.50.2 as the IP address and 255.255.255.0 as the subnet mask. This interface is used for internal routing, and it's devoted to a private network.
- **eth2**: This interface, which shares the same subnet mask as eth1 and an IP address of 192.168.51.2, is to link to a separate part of the virtual network and performs internal routing in a similar fashion.

In order to manage and guide traffic within the virtual network, these interfaces are paramount. The setup points to a planned-out topology in which vmrouter is the central hub connecting various parts of the virtual environment.

3.1.2. Routing Table Configuration

2. Table.

Routing table on vmrouter

	Interface	Destination	Gateway	Flags	Metric	Mask
0	eth0	0.0.0.0	192.168.121.1	3	100	0.0.0.0
1	eth1	192.168.50.0	0.0.0.0	1	0	255.255.255.0
2	eth2	192.168.51.0	0.0.0.0	1	0	255.255.255.0
3	eth0	192.168.121.0	0.0.0.0	1	0	255.255.255.0
4	eth0	192.168.121.1	0.0.0.0	5	100	255.255.255.255

The routing table for vmrouter is seen depicted in table 2.

- Through the eth0 interface, 192.168.121.1 is established as the default gateway IP address for outgoing traffic. This indicates that this interface is the primary means by which data is sent to destinations outside the virtual network, to the internet.
- The fact that 192.168.50.0/24 and 192.168.51.0/24 are reserved for private networks shows that vmrouter uses the correct interfaces (eth1 and eth2) to route traffic intended for those networks.

By creating a barrier between internal and external network traffic, this configuration guarantees valid destination-based routing.

3.1.3. Iptables Configuration

3. Table.

Iptables rules on vmrouter

	Chain	Target	Source	Destination	Additional Options
0	POSTROUTING	MASQUERADE	0.0.0.0/0	0.0.0.0/0	-o eth0 -j
1	POSTROUTING	MASQUERADE	0.0.0.0/0	192.168.51.0/24	-o eth2 -j
2	POSTROUTING	MASQUERADE	0.0.0.0/0	192.168.50.0/24	-o eth1 -j
3	FORWARD	ACCEPT	0.0.0.0/0	0.0.0.0/0	-i eth0 -o eth1 -j
4	FORWARD	ACCEPT	0.0.0.0/0	0.0.0.0/0	-i eth1 -o eth0 -j

Based on how iptables is configured, vmrouter is also in charge of firewall rules and network address translation (NAT). The main points are as follows:

- **MASQUERADE Rules:** These rules are set for all three interfaces (eth0, eth1, and eth2). The MASQUERADE target is used in to configure NAT. It allows outgoing traffic from the private network to appear as if it's coming from the vmrouter itself.
- **FORWARD Chain Rules:** The ACCEPT rules in the FORWARD chain for traffic going from eth0 to eth1 and back again show that vmrouter lets traffic go both ways between these interfaces. This is very important for a router that connects different network segments.

3.2. Router Client Configuration

The router_client virtual machine is also an essential element in the network topology configuration. This virtual machine, labeled as "router_client" , is setup to emulate a client within a network that interacts with a router and then later a VPN server. The router_client VM's configuration demonstrates a setup that encompasses network interface settings, routing configurations, and script provisions to enhance setup and functionality.

3.2.1. Networking Interface Configuration

4. Table.

Network interfaces configuration on router_client

	Interface	IP Address	Subnet Mask	State	MAC Address
0	lo	127.0.0.1	255.0.0.0	UP	00:00:00:00:00:00
1	eth0	192.168.121.199	255.255.255.0	UP	52:54:00:71:5c:29
2	eth1	192.168.50.3	255.255.255.0	UP	52:54:00:7b:49:f2

As can be seen on table 4 both eth0 and eth1 are set up as network interfaces in the router_client virtual machine. Separate subnets are used to assign IP addresses to each interface. Both the eth0 and eth1 interfaces have the same subnet mask and IP addresses: 192.168.121.199 and 255.255.255.0, respectively. With its two interfaces, the virtual machine can link to two different networks, each of which represents a different part of the virtual network architecture.

The fact that eth1 has the private network address 192.168.50.3 suggests that it is part of a private network. The 192.168.50.0/24 subnet is for VM-to-VM communication, such as between the router_client VM and vmrouter and other similar virtual machines. On the other hand, eth0's 192.168.121.199 IP address points to a separate portion of the network, which may be utilized for external communications or for connecting to a separate group of virtual devices or services however this eth1 interface has been rendered useless by the routing table configuration, in order to ensure all communication with other network devices happens through the vmrouter.

3.2.2. Routing Table Configuration

5. Table.

Routing table on router_client

	Interface	Destination	Gateway	Flags	Metric	Mask
0	eth1	0.0.0.0	192.168.50.2	3	0	0.0.0.0
1	eth0	0.0.0.0	192.168.121.1	3	100	0.0.0.0
2	eth1	192.168.50.0	0.0.0.0	1	0	255.255.255.0
3	eth0	192.168.121.0	0.0.0.0	1	0	255.255.255.0
4	eth0	192.168.121.1	0.0.0.0	5	100	255.255.255.255

Important information on the management and routing of traffic can be found in the router-_client VM's routing table as shown on table 5. On the eth1 interface, the default gateway for the virtual machine is set to 192.168.50.2, which is the internal IP address of the vmrouter. With this configuration, all outgoing traffic from the router_client VM will be sent to the vmrouter for additional routing decisions if no specified route has been configured.

The two network interfaces are also represented in the routing table. The 192.168.50.0/24 network is shown as a direct route for eth1, meaning that any traffic going to this subnet will be handled internally and won't require an external gateway. Similarly, eth0 handles traffic within this specific subnet as there is a direct link to the 192.168.121.0/24 network.

Moreover, a higher metric entry for the gateway 192.168.121.1 exists on eth0. Metric field indicates the cost of a route. The high metric is meant to make eth1 gateway the preferred route since it has the lower metric value.

3.3. VPN Server Configuration

Creating a flexible and functioning network architecture for testing VPN obfuscation methods is the primary focus of the configuration of the vpnserver virtual machine in the virtual environment. This virtual machine, as stated before, is meant to act as a VPN server, which is a crucial component of a VPN network. The vpnserver VM's configuration includes network interface settings, routing configurations, and script provisions to enhance setup and functionality.

3.3.1. Networking Interface Configuration

6. Table.

Network interfaces configuration on vpnserver

	Interface	IP Address	Subnet Mask	State	MAC Address
0	lo	127.0.0.1	255.0.0.0	UP	00:00:00:00:00:00
1	eth0	192.168.121.248	255.255.255.0	UP	52:54:00:09:e7:27
2	eth1	192.168.51.51	255.255.255.0	UP	52:54:00:12:b1:b6
3	eth2	192.168.52.51	255.255.255.0	UP	52:54:00:8a:d6:81

There are three network interfaces set up on the vpnserver VM as seen in table 6: eth0, eth1, and eth2. Each of these ports is used for a different thing in the network setup:

- **eth0:** The main interface that the VPN server uses to link to the outside world, created by default for the VM. Its subnet mask is 255.255.255.0 and its IP address is 192.168.121.248.

This link is necessary to connect the VM to the outside world, which makes it easier to connect to the Internet and talk to people outside of the VM.

- **eth1:** This interface has an IP address of 192.168.51.51 and a subnet mask of 255.255.255.0. It is only used for interactions within the subnet connecting vpnserver and vmrouter.
- **eth2:** This interface is also used for internal network transmission, but on a different subnet to vm4, making it so only vpnserver has direct access to vm4. Which will be necessary for the testing of the VPN server.

3.3.2. Routing Table Configuration

7. Table.

Routing table on vpnserver

	Interface	Destination	Gateway	Flags	Metric	Mask
0	eth0	0.0.0.0	192.168.121.1	3	100	0.0.0.0
1	eth1	192.168.50.0	192.168.51.2	3	0	255.255.255.0
2	eth1	192.168.51.0	0.0.0.0	1	0	255.255.255.0
3	eth2	192.168.52.0	0.0.0.0	1	0	255.255.255.0
4	eth0	192.168.121.0	0.0.0.0	1	0	255.255.255.0
5	eth0	192.168.121.1	0.0.0.0	5	100	255.255.255.255

The IP routing table 7 of the vpnserver VM provides critical insights into how network traffic is managed and routed through the VM. The table consists of several entries, each defining a specific route:

- The default route is the one that uses eth0 and has the following parameters: destination 0.0.0.0, gateway 192.168.121.1. It sends data packets to the public internet if they don't fit any of the other routing criteria.
- Both 192.168.51.0/24 and 192.168.52.0/24 routes, with eth1's gateway set to 0.0.0.0 and eth2's set to 0.0.0.0, manage traffic within their respective subnets for internal network traffic.
- Another entry in the routing table is the one for the 192.168.50.0/24 network, which goes via 192.168.51.2 on eth1. This route is reserved for sending data packets to a different subnet so that virtual machines can communicate with one another.

3.3.3. Iptables Configuration

8. Table.

Iptables rules on vpnserver

	Chain	Target	Protocol	Source	Destination	Additional Options
0	POSTROUTING	MASQUERADE	N/A	192.168.0.0/24	0.0.0.0/0	-o eth0 -j
1	POSTROUTING	MASQUERADE	N/A	0.0.0.0/0	192.168.52.0/24	-o eth2 -j
2	FORWARD	TCPMSS	tcp	192.168.0.0/24	0.0.0.0/0	-p tcp -m tcp

In the case of the vpnserver, an important part of controlling and protecting network traffic is the iptables setup on the vpnserver virtual machine as depicted on table 8. Network Address Translation rules, packet forwarding rules, and TCP maximum segment size settings are all part of it:

- MASQUERADE (POSTROUTING): The VM can transform the IP addresses of packets coming from the internal networks (192.168.0.0/24 and 192.168.52.0/24) since the MASQUERADE rules are configured for NAT. In order to manage traffic from several sources and make it appear as though it is originating from a single IP, this is necessary for the VPN server.

- TCPMSS (FORWARD Chain): For packets coming from 192.168.0.0/24, the TCP maximum segment size is adjusted by the rule in the FORWARD chain. For VPN setups in particular, this improvement is essential to preventing fragmentation and enhancing speed. The default MSS for an Ethernet network with an MTU of 1500 bytes is 1460 bytes (*A Standard for the Transmission of IP Datagrams over Ethernet Networks* 1984); if this rule is not followed, TCP connections may use this value. However, the effective MTU is decreased when a VPN is utilized since the VPN protocol introduces extra headers. Inefficiencies and possible connectivity problems could result from packet fragmentation or dropping if the MSS is not adjusted appropriately.

3.4. Remote Client Configuration

3.4.1. Networking Interface Configuration

9. Table.

Network interfaces configuration on vm4

	Interface	IP Address	Subnet Mask	State	MAC Address
0	lo	127.0.0.1	255.0.0.0	UP	00:00:00:00:00:00
1	eth0	192.168.121.97	255.255.255.0	UP	52:54:00:19:0e:09
2	eth1	192.168.52.52	255.255.255.0	UP	52:54:00:bf:1f:e6

3.4.2. Routing Table Configuration

10. Table.

Routing table on vm4

	Interface	Destination	Gateway	Flags	Metric	Mask
0	eth0	0.0.0.0	192.168.121.1	3	100	0.0.0.0
1	eth1	192.168.52.0	0.0.0.0	1	0	255.255.255.0
2	eth0	192.168.121.0	0.0.0.0	1	0	255.255.255.0
3	eth0	192.168.121.1	0.0.0.0	5	100	255.255.255.255

4. EXPERIMENTATION RESULTS

4.1. Method Evaluation Criteria

4.1.1. Detection Resistance

4.1.2. Client-Server Accessibility

4.1.3. Latency

4.1.4. Throughput

4.2. Port switching

4.3. Steganography

4.4. Geneva

4.5. Encryption tunneling

CONCLUSION

REFERENCES

1. AbuHmed, T., Mohaisen, A. & Nyang, D. (2008), 'A survey on deep packet inspection for intrusion detection systems'.

2. Aceto, G. & Pescapé, A. (2015), 'Internet censorship detection: A survey', *Computer Networks* **83**, 381–421.
   URL: *https://www.sciencedirect.com/science/article/pii/S1389128615000948*

3. Alshalan, A., Pisharody, S. & Huang, D. (2016), 'A survey of mobile vpn technologies', *IEEE Communications Surveys & Tutorials* **18**(2), 1177–1196.

4. *A Standard for the Transmission of IP Datagrams over Ethernet Networks* (1984), RFC 894.
   URL: *https://www.rfc-editor.org/info/rfc894*

5. Bock, K., Fax, Y., Reese, K., Singh, J. & Levin, D. (2020), Detecting and evading Censorship-in-Depth: A case study of Iran 's protocol whitelister, *in* '10th USENIX Workshop on Free and Open Communications on the Internet (FOCI 20)', USENIX Association.
   URL: *https://www.usenix.org/conference/foci20/presentation/bock*

6. Bock, K., Hughey, G., Qiang, X. & Levin, D. (2019), Geneva: Evolving censorship evasion strategies, pp. 2199–2214.

7. Chakraborty, R., Jain, H. & Seo, G.-S. (2022), 'A review of active probing-based system identification techniques with applications in power systems', *International Journal of Electrical Power & Energy Systems* **140**, 108008.
   URL: *https://www.sciencedirect.com/science/article/pii/S0142061522000539*

8. Chen, H.-Y. & Lin, T.-N. (2021), 'The challenge of only one flow problem for traffic classification in identity obfuscation environments', *IEEE Access* **PP**, 1–1.

9. Clowwindy, Madeye & Max, L. (n.d.), 'Shadowsocks'.
   URL: *http://www.shadowsocks.org/*

10. Crawford, D. (2019), 'How to hide openvpn traffic – a beginner's guide'.
    URL: *https://proprivacy.com/vpn/guides/how-to-hide-openvpn-traffic-an-introduction*

11. Dingledine, R. & Mathewson, N. (2006), 'Design of a blocking-resistant anonymity system draft'.

12. Dusi, M., Gringoli, F. & Salgarelli, L. (2008), A preliminary look at the privacy of ssh tunnels., pp. 626–632.

13. El-Maghraby, R. T., Abd Elazim, N. M. & Bahaa-Eldin, A. M. (2017), A survey on deep packet inspection, *in* '2017 12th International Conference on Computer Engineering and Systems (ICCES)', pp. 188–197.

14. Ermoshina, K. & Musiani, F. (2021), 'The telegram ban: How censorship "made in russia" faces a global internet', *First Monday* .

15. Ferguson, P. & Huston, G. (1998), What is a vpn ? — part i.
URL: *https://api.semanticscholar.org/CorpusID:140112970*

16. Fifield, D., Lan, C., Hynes, R., Wegmann, P. & Paxson, V. (2015), 'Blocking-resistant communication through domain fronting', *Proceedings on Privacy Enhancing Technologies* **2015**.

17. Ghatikar, T. & Sai, V. (2022), 'Vpn detection and blocking', p. 2022.

18. Handel, T. G. & Sandford, M. T. (1996), Hiding data in the osi network model, *in* R. Anderson, ed., 'Information Hiding', Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 23–38.

19. He, Y. & Chen, M. (2016), 'Protocol mimicry technique and its development', **40**, 1–8.

20. Iacovazzi, A. & Baiocchi, A. (2014), 'Internet traffic privacy enhancement with masking: Optimization and tradeoffs', *Parallel and Distributed Systems, IEEE Transactions on* **25**, 353–362.

21. Kailanya, E., Omamo, A. & Mwadulo, M. (2022), 'Deep packet analysis firewall model', *African Journal of Science, Technology and Social Sciences* **1**(1).
URL: *https://journals.must.ac.ke/index.php/AJSTSS/article/view/62*

22. Khattak, S., Javed, M., Anderson, P. D. & Paxson, V. (2013), Towards illuminating a censorship monitor's model to facilitate evasion, *in* '3rd USENIX Workshop on Free and Open Communications on the Internet (FOCI 13)', USENIX Association, Washington, D.C.
URL: *https://www.usenix.org/conference/foci13/workshop-program/presentation/khattak*

23. Kim, Y.-J., Kolesnikov, V., Kim, H. & Thottan, M. (2011), Sstp: a scalable and secure transport protocol for smart grid data collection, pp. 161 – 166.

24. Knapp, E. D. & Langill, J. T. (2015), Chapter 3 - industrial cyber security history and trends, *in* E. D. Knapp & J. T. Langill, eds, 'Industrial Network Security (Second Edition)', second edition edn, Syngress, Boston, pp. 41–57.
URL: *https://www.sciencedirect.com/science/article/pii/B9780124201149000034*

25. Kundur, D. & Ahsan, K. (2003), 'Practical internet steganography: Data hiding in ip'.

26. Lipp, B., Blanchet, B. & Bhargavan, K. (2019), A mechanised cryptographic proof of the wireguard virtual private network protocol, pp. 231–246.

27. Liu, J., Gao, N., Tu, C., Zhang, Y. & Sun, Y. (2023), A pure hardware design and implementation on fpga of wireguard-based vpn gateway, *in* '2023 26th International Conference on Computer Supported Cooperative Work in Design (CSCWD)', pp. 1220–1225.

28. Loshin, P. (2013), Chapter 4 - tor relays, bridges, and obfsproxy, *in* P. Loshin, ed., 'Practical Anonymity', Syngress, Boston, pp. 69–80.
URL: *https://www.sciencedirect.com/science/article/pii/B9780124104044000043*

29. Matalgah, M., Sheikh, K., Thaker, M., Chaudhry, G., Medhi, D. & Qaddour, J. (2002), Performance analysis of ipsec protocol: Encryption and authentication, Vol. 2, pp. 1164 – 1168 vol.2.

30. Munshi, A. (2023), 'Hybrid detection technique for ip packet header modifications associated with store-and-forward operations', *Applied Sciences* **13**(18).
URL: *https://www.mdpi.com/2076-3417/13/18/10229*

31. Naidu, D. & Jha, M. (2023), 'Detection technique to trace ip behind vpn/proxy using machine learning', *International Journal of Next-Generation Computing* .

32. Network Solutions LLC (2021).
URL: *https://openvpn.net/community-resources/how-to/*

33. Pandey, J., Rai, S. & R, S. (2023), 'Assessment of deep packet inspection system of network traffic and anomaly detection', *International Journal of Scientific Research in Science, Engineering and Technology* pp. 680–688.

34. Pang, Y., Jin, S., Li, S., Li, J. & Ren, H. (2013), Openvpn traffic identification using traffic fingerprints and statistical characteristics, *in* Y. Yuan, X. Wu & Y. Lu, eds, 'Trustworthy Computing and Services', Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 443–449.

35. Papadogiannaki, E. & Ioannidis, S. (2021), ‘Acceleration of intrusion detection in encrypted network traffic using heterogeneous hardware’, *Sensors* **21**(4).
URL: *https://www.mdpi.com/1424-8220/21/4/1140*

36. Rathore, M. S., Razzaq, A., Hidell, M. & Sjödin, P. (2009), Site-to-site vpn technologies : A survey.
URL: *https://api.semanticscholar.org/CorpusID:35973654*

37. Sun, S. H. (2011), The advantages and the implementation of ssl vpn, *in* ‘2011 IEEE 2nd International Conference on Software Engineering and Service Science’, pp. 548–551.

38. Trivedi, U. & Patel, M. (2016), A fully automated deep packet inspection verification system with machine learning, *in* ‘2016 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS)’, pp. 1–6.

39. Wang, Y., Ji, P., Ye, B., Wang, P., Luo, R. & Yang, H. (2014), Gohop: Personal vpn to defend from censorship, pp. 27–33.

40. Wang, Z., Cao, Y., Qian, Z., Song, C. & Krishnamurthy, S. V. (2017), Your state is not mine: A closer look at evading stateful internet censorship, *in* ‘Proceedings of the 2017 Internet Measurement Conference’, IMC ’17, Association for Computing Machinery, New York, NY, USA, p. 114–127.
URL: *https://doi.org/10.1145/3131365.3131374*

41. Wang, Z., Zhu, S., Cao, Y., Qian, Z., Song, C., Krishnamurthy, S. V., Chan, K. S. & Braun, T. D. (2020), Symtcp: Eluding stateful deep packet inspection with automated discrepancy discovery, *in* ‘NDSS’.

42. Winter, P. & Lindskog, S. (2012), ‘How the great firewall of china is blocking tor’.

43. Wu, M., Sippe, J., Sivakumar, D., Burg, J., Anderson, P., Wang, X., Bock, K., Houmansadr, A., Levin, D. & Wustrow, E. (2023), How the great firewall of china detects and blocks fully encrypted traffic, *in* ‘32nd USENIX Security Symposium (USENIX Security 23)’, USENIX Association, Anaheim, CA, pp. 2653–2670.
URL: *https://www.usenix.org/conference/usenixsecurity23/presentation/wu-mingshi*

44. Yoo, H. & Ahmed, I. (2019), Control logic injection attacks on industrial control systems.

45. Zhao, Y., Ma, X., Li, J., Yu, S. & Li, W. (2018), Revisiting website fingerprinting attacks in real-world scenarios: A case study of shadowsocks, *in* M. H. Au, S. M. Yiu, J. Li, X. Luo, C. Wang, A. Castiglione & K. Kluczniak, eds, 'Network and System Security', Springer International Publishing, Cham, pp. 319–336.