

**Final Year Project**

**COMP60011 · Bachelor of Science in Cybersecurity**

**Optimising Privacy Online with a VPN-Tor-VPN Chain:   
A Multi-Layered Approach**

**Student: Alexei Gaicovschi | ID: 21028335**

**Course Leader: Viraj Dawarka**

**1st Supervisor: Samuel Onalo**

**2nd Supervisor: Mohammad Heydari**

**Final version submitted: 02 May 2025**

Contents

[Table of Figures 4](#_Toc197091790)

[Table of Tables 4](#_Toc197091791)

[Table of Listings 5](#_Toc197091792)

[Abbreviations 6](#_Toc197091793)

[Abstract 8](#_Toc197091794)

[Chapter 1: Introduction 9](#_Toc197091795)

[1.1 Research Background 9](#_Toc197091796)

[1.2 Hypothesis 9](#_Toc197091797)

[1.3 Aim 10](#_Toc197091798)

[1.4 Objectives 10](#_Toc197091799)

[1.5 Deliverables 10](#_Toc197091800)

[1.6 Structure of Report 10](#_Toc197091801)

[Chapter 2: Literature Review 11](#_Toc197091802)

[2.1 Introduction to Privacy Technologies 11](#_Toc197091803)

[2.2 Virtual Private Networks (VPNs): Architecture, Risks, and Industry Practices 11](#_Toc197091804)

[2.2.1 VPN Architecture and Trust Model 11](#_Toc197091805)

[2.2.2 VPN – Legacy Threats and Known Vulnerabilities 11](#_Toc197091806)

[2.2.3 VPN – Modern Threats 12](#_Toc197091807)

[2.2.4 Justification for Self-Hosted or Multi-Hop VPN Chains 13](#_Toc197091808)

[2.3 Tor – Architecture and Vulnerabilities 14](#_Toc197091809)

[2.3.1 Tor’s Architecture and P2P Entry/Exit Nodes 14](#_Toc197091810)

[2.3.2 Theoretical and Legacy Vulnerabilities in Tor’s Network 15](#_Toc197091811)

[2.3.3 Practical Vulnerabilities and Attacks on Tor’s Network 16](#_Toc197091812)

[2.3.4 Tor Deanonymisation attacks, practical examples 18](#_Toc197091813)

[2.4 Hybrid Privacy Architectures – Combining VPN and Tor 20](#_Toc197091814)

[2.4.1 Rationale Behind Layered Configurations 20](#_Toc197091815)

[2.4.2 Architectural Designs of Hybrid VPN & Tor Configurations 20](#_Toc197091816)

[2.4.3 Security Benefits and Limitations of Hybrid Configurations 21](#_Toc197091817)

[2.4.5 Enhanced Privacy Through Trust Segmentation in VPN-Tor-VPN Configuration 23](#_Toc197091818)

[2.4.6 Research Gaps and Future Directions in Hybrid Privacy Architectures 25](#_Toc197091819)

[Chapter 3: Research Methodology 26](#_Toc197091820)

[3.1 Research Strategy 26](#_Toc197091821)

[3.2 Experimental Design 27](#_Toc197091822)

[3.3 Data Collection and Analysis 27](#_Toc197091823)

[3.4 Toolchain and environment 28](#_Toc197091824)

[3.5 Limitation 28](#_Toc197091825)

[3.6 Ethical and legal compliance 29](#_Toc197091826)

[Chapter 4: Design 30](#_Toc197091827)

[4.1 System Overview 30](#_Toc197091828)

[4.2 Node Roles and Layer Responsibilities 30](#_Toc197091829)

[4.2.1 VPS1 – Entry VPN Server (Tor Gateway) 30](#_Toc197091830)

[4.2.2 VPS2 – Public Tor Exit Node with VPN Client 31](#_Toc197091831)

[4.2.3 VPS3 – Final VPN Server (Exit Point) 31](#_Toc197091832)

[4.3 Layered Security and Encryption Model 31](#_Toc197091833)

[4.3.1 Encryption and Routing Path 31](#_Toc197091834)

[4.3.2 Node-Specific Security Controls 32](#_Toc197091835)

[4.3.3 Security Guarantees by Design 33](#_Toc197091836)

[4.4 Deployment Automation and Scripts 33](#_Toc197091837)

[4.5 Threat Model 34](#_Toc197091838)

[Chapter 5: Implementation 35](#_Toc197091839)

[5.1 Infrastructure provisioning 35](#_Toc197091840)

[5.2 Script Architecture 35](#_Toc197091841)

[5.3 Deployment Workflow 36](#_Toc197091842)

[5.4 Residual manual tasks 37](#_Toc197091843)

[5.5 Reproducibility and repository layout 37](#_Toc197091844)

[5.5.1 Repository Features 37](#_Toc197091845)

[5.5.2 Repository Layout 38](#_Toc197091846)

[Chapter 6: Testing & Evaluation 39](#_Toc197091847)

[6.1 Performance Benchmarking 39](#_Toc197091848)

[6.1.1 Method 39](#_Toc197091849)

[6.1.2 Results 39](#_Toc197091850)

[6.1.3 Latency Variability and Peak Hour Trends 39](#_Toc197091851)

[6.2 Security Testing 41](#_Toc197091852)

[6.2.1 Summary matrix 41](#_Toc197091853)

[6.2.2 Representative tests 42](#_Toc197091854)

[6.2.2.1 Timing-correlation (S-1; S-1b) 42](#_Toc197091855)

[6.2.2.2 DNS and WebRTC leakage (S-2 & S-4) 42](#_Toc197091856)

[6.2.2.3 Cipher-downgrade resilience (S-8) 44](#_Toc197091857)

[6.2.2.4 DPI evasion (S-9) 44](#_Toc197091858)

[Chapter 7: Results & Discussion 46](#_Toc197091859)

[7.1 Key Outcomes 46](#_Toc197091860)

[7.2 Flow-Correlation Analysis 46](#_Toc197091861)

[7.3 Performance Trade-offs 46](#_Toc197091862)

[7.4 Limitations and future work 46](#_Toc197091863)

[Chapter 8: Conclusion 47](#_Toc197091864)

[References 48](#_Toc197091865)

[Appendices 53](#_Toc197091866)

[Appendix A: Supervisory Meetings Summary 53](#_Toc197091867)

[Appendix B: Ethics Disclaimer 54](#_Toc197091868)

[Appendix C: Risk Assessment 55](#_Toc197091869)

[Appendix D: Gantt Chart 56](#_Toc197091870)

[Appendix E: Artefact Disclaimer 57](#_Toc197091871)

[Appendix F: Detailed Statistical Output 58](#_Toc197091872)

[Appendix G:  Packet-capture verification (Wireshark) 59](#_Toc197091873)

[Appendix H: WebRTC leak test 61](#_Toc197091874)

# **Table of Figures**

[**Figure I: VPN-Tor-VPN Privacy Chain – Traffic and Role Distribution** 30](#_Toc197089284)

[**Figure II: Layered Encryption Flow** 32](#_Toc197089285)

[**Figure III: Performance Visualisation. Latency Distribution by Configuration** 40](#_Toc197089286)

[**Figure IV: Performance Visualisation. Latency Trends over Runs** 40](#_Toc197089287)

[**Figure V: PyShark Output Demonstrating Pairs = 631, ρ = 0.04** 42](#_Toc197089288)

[**Figure VI: Re-run with Padding, Pairs = 2042, ρ = 0.01** 42](#_Toc197089289)

[**Figure VII: Chrome dnsleaktest (4 x Cloudflare).** 43](#_Toc197089290)

[**Figure VIII: Firefox dnsleaktest (authoritative list).** 43](#_Toc197089291)

[**Figure IX: Client OpenVPN Logs. Correct Data-Cipher Enforcement** 44](#_Toc197089292)

[**Figure X: pfSense with nDPI Filtering Dropping TCP Openvpn** 45](#_Toc197089293)

[**Figure XI: Ethics Disclaimer (Screenshot)** 54](#_Toc197089294)

[**Figure XII: Risk Assessment (Screenshot)** 55](#_Toc197089295)

[**Figure XIII: Project Timeline Gantt Chart** 56](#_Toc197089296)

[**Figure XIV: Descriptive Statistics for Performance Metrics (n = 45)** 58](#_Toc197089297)

[**Figure XV: Normality and Welch’s t‑tests on Download Throughput** 58](#_Toc197089298)

[**Figure XVI: tcpdump Capture on tun0 of VPS3 (1 205 packets in 30 s window)** 59](#_Toc197089299)

[**Figure XVII: Wireshark Filter ip.addr == 90.196.\*.\* (home ISP) - Zero Matches** 59](#_Toc197089300)

[**Figure XVIII: Wireshark Filter ip.addr == 167.99.193.96 (VPS1) – Zero Matches** 60](#_Toc197089301)

[**Figure XIX: ip.addr == 68.183.70.247 (VPS2) – Zero Matches** 60](#_Toc197089302)

[**Figure XX: tcpdump Capture on tun0 of VPS3 - Wireshark Visual Analysis** 60](#_Toc197089303)

[**Figure XXI: BrowserLeaks WebRTC test** 61](#_Toc197089304)

# **Table of Tables**

[**Table I: Legacy VPN Vulnerabilities and Their Real-World Impact.** 11](#_Toc197087361)

[**Table II: Modern VPN Threats.** 12](#_Toc197087362)

[**Table III: Theoretical and Legacy Vulnerabilities in Tor’s Network.** 15](#_Toc197087363)

[**Table IV: Overview of the Practical Vulnerabilities and Attacks on Tor’s Network.** 17](#_Toc197087364)

[**Table V: Summary Comparison of Hybrid VPN and Tor Configurations** 22](#_Toc197087365)

[**Table VI: VPN-Tor-VPN: Resistance to Advanced Surveillance and Censorship** 23](#_Toc197087366)

[**Table VII: Research Strategy and Design Iterations** 25](#_Toc197087367)

[**Table VIII: Experimental Design and Measurements** 26](#_Toc197087368)

[**Table IX: Toolchain and Environment** 27](#_Toc197087369)

[**Table X: Research Limitations and Mitigation Strategy** 27](#_Toc197087370)

[**Table XI: Layered Encryption Structure** 31](#_Toc197087371)

[**Table XII: Node-Specific Security Controls at Every Layer** 32](#_Toc197087372)

[**Table XIII: Scripts Presented and Their Purpose** 33](#_Toc197087373)

[**Table XIV: DigitalOcean Infrastructure Overview** 34](#_Toc197087374)

[**Table XV: Performance Benchmark & Results (n = 45)** 38](#_Toc197087375)

[**Table XVI: Security Validation Matrix** 40](#_Toc197087376)

[**Table XVII: Supervisory Meeting Summary Table (Appendix A)** 52](#_Toc197087377)

# **Table of Listings**

[**Listing I: Basic Linux Hardening** 35](#_Toc197092165)

[**Listing II: Key excerpt from setup\_exit\_node.sh** 36](#_Toc197092166)

# **Abbreviations**

| **Abbreviation** | **Full Term** | |
| --- | --- | --- |
| **VPN** | | Virtual Private Network |
| **Tor** | | The Onion Router |
| **VPS** | | Virtual Private Server |
| **ISP** | | Internet Service Provider |
| **PET** | | Privacy-Enhancing Technology |
| **DPI** | | Deep Packet Inspection |
| **DNS** | | Domain Name System |
| **IP** | | Internet Protocol |
| **TLS** | | Transport Layer Security |
| **TCP** | | Transmission Control Protocol |
| **UDP** | | User Datagram Protocol |
| **PPTP** | | Point-to-Point Tunnelling Protocol |
| **MS-CHAPv2** | | Microsoft Challenge-Handshake Authentication Protocol version 2 |
| **RAM** | | Random Access Memory |
| **DoH** | | DNS over HTTPS |
| **I2P** | | Invisible Internet Project |
| **BGP** | | Border Gateway Protocol |
| **ORPort** | | Onion Routing Port |
| **DirPort** | | Directory Port |
| **NTP** | | Network Time Protocol |
| **CSV** | | Comma-Separated Values |
| **PCAP** | | Packet Capture |
| **DHCP** | | Dynamic Host Configuration Protocol |
| **NAT** | | Network Address Translation |
| **NDP** | | Neighbour Discovery Protocol |
| **MTU** | | Maximum Transmission Unit |
| **SYN** | | Synchronise Packet (TCP flag) |
| **PSH** | | Push Packet (TCP flag) |
| **VTV** | | VPN-to-Tor-to-VPN Chain (as a conceptual model) |
| **SNI** | | Server Name Indication |
| **pfSense** | | (Firewall platform, no expansion needed – standard name) |
| **nDPI** | | Open-Source Deep Packet Inspection library |
| **OPSEC** | | Operational Security |
| **RPKI** | | Resource Public Key Infrastructure |
| **ASN** | | Autonomous System Number |
| **CDN** | | Content Delivery Network |
| **TTR** | | Trusted Recursive Resolver |
| **TRR** | | Trusted Recursive Resolver (Mozilla Firefox's DNS mechanism) |
| **DPI Evasion** | | Deep Packet Inspection Evasion (used as a shorthand) |
| **IPv6** | | Internet Protocol version 6 |
| **IPv4** | | Internet Protocol version 4 |
| **SUMo** | | Sliding-Subset Sum (correlation attack technique) |
| **JA3** | | TLS Fingerprinting Method |
| **UX** | | User Experience |
| **CSV/PCAP** | | Comma-Separated Values / Packet Capture |
| **ACM** | | Association for Computing Machinery |

# **Abstract**

Cryptographic primitives are mature, yet privacy failures still arise from misplaced trust. This study therefore designs and evaluates a three-hop VPN-Tor-VPN chain intended to partition knowledge of source, route, and destination across independent infrastructure while preserving user control of the final exit point. The prototype was provisioned on hardened Ubuntu 22.04 servers via reproducible infrastructure-as-code scripts and refined in a two-cycle build-measure-refine process.

Empirical evaluation balances performance and confidentiality. Across forty-five benchmark runs the chain sustained a median downstream throughput of ≈ 22 Mb s⁻¹ and a median round-trip time of ≈ 136 ms (95th – percentile 195 ms) – comparable to Tor alone. Security tests demonstrated resistance to timing-correlation attacks (Pearson ρ = 0.01 with padding enabled) and blocked protocol fingerprinting under nDPI classification.

The project offers an open-source, reproducible, flexible framework for hybrid privacy architectures by sharing code, datasets and configuration artefacts. To make the model more useful it can be extended to incorporate IPv6 support, pluggable transports and BGP hijack resilience.

**Keywords:** VPN, Tor, layered security, distributed trust, privacy engineering, traffic‑analysis resistance, cyber‑privacy.

# **Chapter 1: Introduction**

## 1.1 Research Background

The concept that systems are only as secure as their weakest link underpins the contemporary cybersecurity landscape (Schneier, 2015). Despite advancements in safeguarding online privacy, achieving absolute privacy remains impossible. For users seeking complete anonymity, disconnecting from the internet would offer the most definitive solution, albeit one that is not feasible in today’s interconnected society.

As internet surveillance, censorship, and data harvesting have become increasingly pervasive, the demand for reliable privacy technologies has grown sharply. In authoritarian regimes, activists and citizens often rely on privacy-enhancing technologies (PETs) such as Virtual Private Networks (VPNs) and The Onion Router (Tor) to bypass restrictions and maintain anonymity online (Cabrera, 2024). However, both systems exhibit inherent limitations. While VPNs encrypt network traffic and circumvent censorship, many providers retain metadata logs, potentially compromising anonymity under legal pressures (Ramadhani, 2018). Similarly, Tor, though effective at anonymising traffic through multi-hop routing, remains susceptible to deanonymisation attacks, suffers from exit node unpredictability and is vulnerable to sophisticated correlation attacks, particularly in environments with heightened surveillance (Pascal & Adrian, 2024); (Abbott, et al., 2007). These vulnerabilities underscore the necessity for a multi-layered approach to enhance online privacy and security.

This project addresses the practical privacy gap that emerges from these limitations by exploring a hybrid chain model – VPN-Tor-VPN chain, designed to decentralise trust, enhance traffic obfuscation, and offer controlled, trusted egress. While the idea of combining VPNs and Tor has been discussed in forums and among privacy advocates, few implementations offer a robust, transparent, and testable prototype that addresses exit-node surveillance, DNS/IP leaks, and user fingerprinting in a comprehensive way.

The concept behind this project has long been of personal and professional interest to the researcher. The Final Year Project framework provides a valuable opportunity to design, implement, and evaluate a solution that has long been conceptually relevant and technically aspirational in the field of cybersecurity.

Furthermore, existing research does not provide sufficient guidance on how such chains can be structured securely and effectively, especially when using multiple independently hosted servers. This project contributes a working implementation, security analysis, and comparative evaluation of the VPN-Tor-VPN model in the context of modern privacy threats.

## 1.2 Hypothesis

If a multi-hop chain using VPN-Tor-VPN is properly implemented with trusted infrastructure and hardened configurations, then it can offer enhanced online privacy and resistance to traffic correlation, DPI, and untrusted exit node threats – surpassing the effectiveness of Tor or VPN alone and distributing trust across multiple nodes within the chain.

## 1.3 Aim

This project aims to implement and evaluate a resilient VPN‑Tor‑VPN topology that conceals both origin and destination addresses whilst reducing susceptibility to timing‑correlation attacks and distributes trust.

## 1.4 Objectives

* To critically review the security and privacy limitations of Tor and VPNs individually.
* To identify key attack vectors against VPN and Tor users
* To design and implement a working VPN-Tor-VPN chain using multiple VPS servers.
* To test the solution against common threats such as DNS leaks, DPI detection, and traffic correlation.
* To evaluate the practical impact, trade-offs, and resilience of the model compared to existing configurations.
* To contribute a reusable artefact and technical guidance for researchers and privacy-conscious users.

## 1.5 Deliverables

1. A fully functional, secure prototype of a VPN-Tor-VPN privacy chain, deployed across three VPS instances with hardened configurations.
2. A detailed literature review analysing VPN, Tor, and hybrid anonymity technologies.
3. A comprehensive final report, covering design, implementation, testing, and evaluation.
4. A GitHub repository including .ovpn configuration files, documentation, and testing results.

## 1.6 Structure of Report

This report is structured as follows:

Chapter 1: **Introduction** – Covers the project overview, hypothesis, aim, objectives, deliverables and structure of the report.

Chapter 2: **Literature Review** – An in-depth analysis of current VPN and Tor technologies, attack vectors, and hybrid models.

Chapter 3: **Research Methodology** – A justification of the technical and testing approaches used.

Chapter 4: **Design** – Description of the system architecture, privacy principles, and VPN/Tor chaining logic.

Chapter 5: **Implementation** – A walkthrough of the practical setup and configuration of the multi-hop chain.

Chapter 6: **Testing** **and** **Evaluation** – Evaluation of the artefact using privacy assessment tools and threat simulations.

Chapter 7: **Results & Discussion** – Discussion of the results, limitations, and suggestions for future work.

Chapter 8: **Conclusion** – Summary of findings.

# **Chapter 2: Literature Review**

## 2.1 Introduction to Privacy Technologies

Online privacy is increasingly challenged by pervasive surveillance, commercial tracking, and censorship. In response, users have turned to tools such as Virtual Private Networks (VPNs) and The Onion Router (Tor) to obscure their digital footprints (Ramadhani, 2018). VPNs create encrypted tunnels between a user and a provider’s server, shifting trust from local ISPs to commercial intermediaries. Tor, by contrast, distributes trust via decentralised multi-hop relays, aiming to provide anonymity through unlikability. These tools, however, face evolving threats - from deep packet inspection to AI-powered deanonymisation (Pascal & Adrian, 2024). This review analyses their architectures, threats, and recent research into hybrid models, aim to enhance anonymity by combining the strengths of both.

## 2.2 Virtual Private Networks (VPNs): Architecture, Risks, and Industry Practices

### 2.2.1 VPN Architecture and Trust Model

A Virtual Private Network (VPN) creates a secure, encrypted tunnel between the user’s device and a VPN server, masking their IP address and encrypting traffic through protocols such as OpenVPN, IKEv2/IPSec, or WireGuard. This tunnel shields users from local network surveillance and provides access to geo-restricted or censored content (Abbas, et al., 2023).

However, VPNs are inherently centralised. All internet traffic exits through a VPN server controlled by the provider, requiring the user to trust that provider not to log or leak activity (Rytilahti & Holz, 2024). The effectiveness of a VPN in preserving privacy therefore depends on a mix of technical safeguards (e.g., RAM-only servers, DNS leak protection) and policy assurances (e.g., audited no-log commitments). Unlike Tor, which decentralises traffic through multiple relays, a VPN introduces a single point of trust and failure (Ramadhani, 2018).

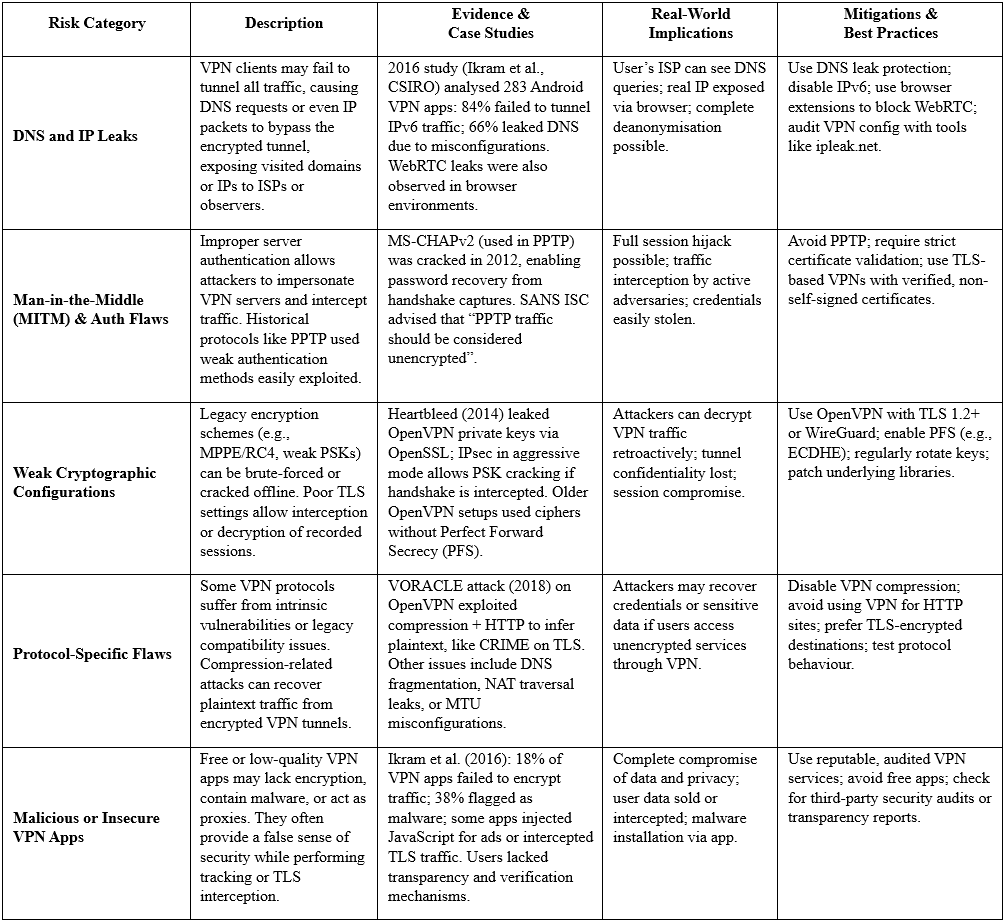
### 2.2.2 VPN – Legacy Threats and Known Vulnerabilities

VPN technologies have been subject to various attacks and leaks over the years. Legacy vulnerabilities in VPN protocols (e.g., PPTP) are now well documented. Microsoft’s MS-CHAPv2 authentication protocol used in PPTP was cracked using cloud computing as early as 2012 (Microsoft, 2012). Historically documented risks include basic IP/DNS leakage, cryptographic weaknesses in older VPN protocols, and man-in-the-middle possibilities in poorly configured setups (Abbas, et al., 2023).

Presented below table provides a comprehensive overview of several notable legacy threats associated with VPN technology:

**Table I: Legacy VPN Vulnerabilities and Their Real-World Impact.**

Source: (Microsoft, 2012); (OpenVPN, 2014); (Ikram, et al., 2016); (OpenVPN, 2018);



This table offers a focused review of historically documented VPN risks, highlighting some of the most impactful legacy vulnerabilities that undermined user privacy and security.

Such weaknesses have since become well-documented, and in many cases, mitigated through the adoption of modern VPN protocols and secure configuration practices. However, they serve as a critical baseline to understand how VPN systems can fail. In the following section, researcher examines how the VPN landscape has evolved in the recent years, incorporating new threat models, regulatory scrutiny, and advances in protocol design aimed at addressing both legacy and contemporary risks.

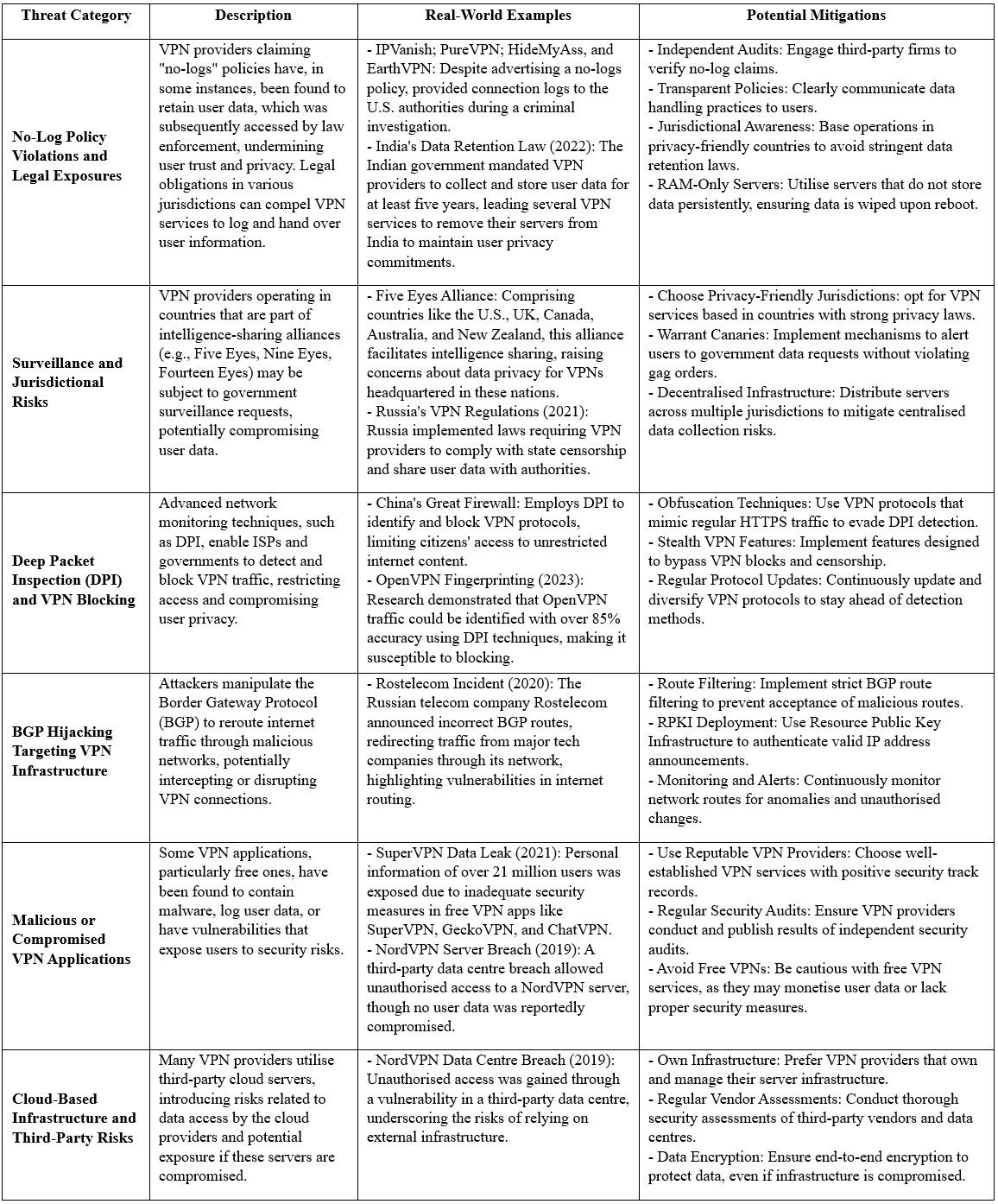
### 2.2.3 VPN – Modern Threats

The VPN landscape has evolved with new threats and heightened scrutiny on privacy claims.

The table below is focused on several contemporary issues around VPN technology:

**Table II: Modern VPN Threats.**

Source: (Lekander, 2018); (Indian Computer Emergency Response Team, 2022); (Bui, et al., 2019); (Xue, et al., 2024); (Mark, 2023); (Hoang, et al., 2021); (Hope, 2020)



### 2.2.4 Justification for Self-Hosted or Multi-Hop VPN Chains

Given the above risks within modern threat landscape, many advanced users and researchers argue for regaining control of the VPN either by self-hosting one’s own VPN server or chaining multiple services to dilute trust. This shifts the trust model away from opaque commercial services and offers greater transparency and control (Ranjan, 2021).

* **Self-Hosting VPN Servers**: Hosting your own VPN (on a VPS or home server) eliminates reliance on a third-party VPN provider. You can disable logging, control encryption settings, and ensure up-to-date protocols. While some trust must still be placed in the infrastructure provider (e.g., VPS host), the risk surface is significantly reduced. Unlike commercial VPNs, there are no hidden trackers, bundled telemetry, or logging ambiguities. This approach also allows selection of privacy-friendly jurisdictions like Iceland or Switzerland, for your VPN’s exit location, reducing legal compulsion risk (Lim & Oh, 2025).
* **Multi-Hop Chains**: Using multiple VPNs in sequence, like VPN over VPN separates identity from destination. No single entity sees both ends of your traffic. This concept increases privacy even in the event when one provider is compromised. While commercial “Double VPN” features exist, chaining different services (or combining with Tor, as per researcher idea) decentralises trust and resists collusion (Tkachov, et al., 2020).
* **Advanced Configurations**: Self-hosted servers allow further enhancements - like running a private DNS resolver with DNSCrypt/DoH, enabling Encrypted SNI, or integrating Tor/I2P routing. These configurations are generally unsupported by commercial VPNs but greatly improve resilience against surveillance and metadata leaks (Roger & Doussot, 2020).
* **Avoiding Commercial Exploitation**: Some VPNs, e.g., Hola VPN, have misused user traffic for profit (Morris, 2015). Self-hosting removes this risk entirely. You retain control over the stack, from cipher suite to connection policies, and avoid opaque practices or legal traps imposed by service providers.

**Considerations**: While self-hosting or chaining increases control, it requires technical skill, proper hardening, and may still expose metadata if infrastructure is not anonymised and operational security (OPSEC) is not adequate (e.g., paying with your real identity). Nonetheless, these methods demonstrate a proactive move from “trusting a VPN” to verifying your own privacy posture – a relevant direction in privacy engineering and in defending against modern VPN threats.

## 2.3 Tor – Architecture and Vulnerabilities

### 2.3.1 Tor’s Architecture and P2P Entry/Exit Nodes

The Tor network is a low-latency anonymity system that routes traffic through multiple volunteer-run relays (nodes) to conceal users’ identities and destinations. Tor builds circuits typically consisting of three hops: an entry/guard node, a middle relay, and an exit node (The Tor Project, 2023)**.**

Traffic is encrypted in layers and each relay removes one layer, so no single relay learns both who the client is and what site is being accessed ​ (Javed, 2023). The entry guard sees the client’s IP but not the destination, and the exit sees the destination but not the origin​. This separation is fundamental to Tor’s anonymity. Unlike a traditional VPN, which has a fixed entry and exit under one provider, Tor’s relays are distributed peers – any suitable volunteer relay can be selected as an entry or exit. The network’s peer-to-peer-like architecture means there is no centralised exit point, instead, exit nodes are chosen randomly per circuit from the pool of volunteer relays that permit outbound traffic. Tor relies on a set of directory authorities that publish a consensus of all known relays and their roles (flags indicating if a relay is eligible to be an entry guard or exit)​. This decentralised design provides resilience but complicates attempts to predetermine or mark specific entry/exit points, since these roles are dynamic and hidden within the encrypted routing process (The Tor Project, 2023).

**Entry Guards:** To mitigate certain attacks, Tor introduced entry guards – a small, fixed set of relays that each client uses for the first hop of all circuits for an extended period​ (Karunanayake, et al., 2021). Prior to guards (in Tor’s early years), every new circuit picked a random entry, meaning eventually an adversary running some relays might become the first hop for a user. The guard design reduces this risk by limiting exposure: a user sticks to one guard or a few guards for months, so an attacker has fewer opportunities to appear as the entry node​ (Karunanayake, et al., 2021). Guards are typically high-bandwidth, stable relays (they must earn the Guard flag in the consensus by being online long enough and reliable)​. By using a stable entry, Tor trades a bit of load-balancing for security – it becomes harder for an adversary to consistently insert a malicious relay as someone’s entry over time​ (The Tor Project, no date). However, the guard node itself then becomes a critical point: if it is compromised or observed, it can profile the user’s traffic patterns over a long time​ (Karunanayake, et al., 2021). Researchers note this tension: longer-lived guards improve security against certain attacks but simultaneously make a successful guard compromise more damaging​.

**Exit Nodes:** The exit relay is the last Tor node that decrypts the final layer and forwards traffic to the destination on the regular Internet​. Exits are crucial because they interact with external servers on behalf of users. They are also a point of vulnerability: by design, the exit can see the plaintext of any traffic that is not end-to-end encrypted (e.g., bare HTTP) and knows which destination is being contacted (The Tor Project, no date). However, the exit does not know the source IP of the client – it only sees a Tor internal address from the previous hop. The set of relays that permit exit traffic to various ports is public (Tor’s consensus lists their policies), but not every relay is an exit. Many relays are non-exits, and some are exits with restrictions (for example, some exit nodes block ports like 25 for SMTP). The distributed nature of Tor means at any given time, the “exit point” for a circuit is one of potentially thousands of possibilities, making it non-trivial to apriori mark which node will be used as the exit for an arbitrary circuit​ (Karunanayake, et al., 2021). Indeed, while the Tor Project publishes a list of known exit relays, a user’s Tor client chooses one at random (weighted by bandwidth) per circuit. This dynamic selection complicates any scheme that assumes the ability to identify or control the exit beforehand​. In summary, Tor’s P2P-like reliance on community-run entry and exit nodes is key to its anonymity, but it creates an attack surface where those very nodes, if misbehaving or observed, can undermine user privacy (Karunanayake, et al., 2021).

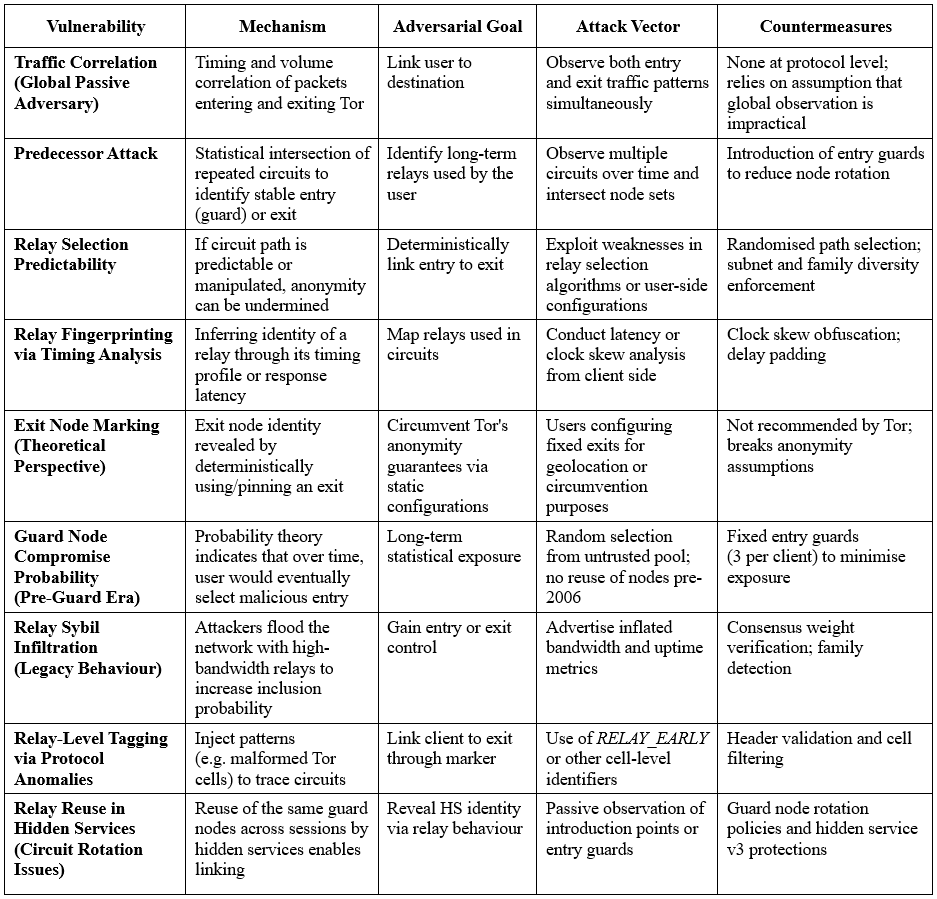
### 2.3.2 Theoretical and Legacy Vulnerabilities in Tor’s Network

This section focuses on long-known, foundational risks in Tor’s architecture that have been documented and discussed in academic literature for nearly two decades. While not always exploited in practice, these vulnerabilities expose the theoretical limits of Tor’s anonymity under certain adversary models.

Below is a table summarising Theoretical and Legacy Vulnerabilities in Tor’s network:

**Table III: Theoretical and Legacy Vulnerabilities in Tor’s Network.**

Source: (Bauer, et al., 2007); (Edman & Syverson, 2009); (Biryukov, et al., 2013); (Johnson, et al., 2013); (Karunanayake, et al., 2021)



### 2.3.3 Practical Vulnerabilities and Attacks on Tor’s Network

The Tor network relies on thousands of volunteer-operated relays to provide anonymity and censorship circumvention services to millions of users globally. However, its open design also allows malicious actors to operate relays with the intent to undermine user privacy or compromise the integrity of the system (Karunanayake, et al., 2021). The Tor Project classifies relays as malicious based on specific criteria. A relay is considered malicious if it demonstrates behaviours such as DNS poisoning, SSL stripping, traffic sniffing, manipulation of onion service directories, excessive logging, or performing Sybil attacks by flooding the network with numerous relays (The Tor Project, 2025). Unlike misconfigured relays, which may be rehabilitated through operator contact, malicious relays are permanently rejected to prevent them from occupying any position in the Tor circuit path.

This distinction is critical because it reflects the practical reality of how Tor's anonymity can be undermined not just by theoretical vulnerabilities, but through tangible, real-world actions by adversaries. These behaviours are not merely technical missteps, they often stem from well-resourced, targeted attacks that exploit Tor’s decentralised architecture, trust model, and path selection algorithms.

The following table provides a categorised summary of these practical vulnerabilities, structured by adversarial goal, attack mechanism, and mitigation strategy. It includes historically documented attacks as well as criteria directly cited by the Tor Project, thereby offering a unified view of practical threat vectors in Tor’s infrastructure.

**Table IV: Overview of the Practical Vulnerabilities and Attacks on Tor’s Network.**

Source: (Sun, et al., 2015); (Kwon, et al., 2015); (Nasr, et al., 2018); (Karunanayake, et al., 2021); (The Tor Project, 2025)



### 2.3.4 Tor Deanonymisation attacks, practical examples

​Recent research and law enforcement operations have highlighted various practical attack methods aimed at deanonymising Tor users. The cases mentioned below underscore the evolving landscape of attacks against Tor users. Continuous research and vigilance are imperative to enhance the security and anonymity of the Tor network.

Below is a summary of notable and most recent techniques and case studies:

1. **Traffic Analysis and Timing Attacks**

In 2024, German law enforcement agencies reportedly conducted timing analysis attacks to deanonymise users of the "Boystown" child abuse platform. By operating their own Tor nodes and analysing traffic patterns, they were able to correlate the timing of data packets entering and leaving the network, leading to successful identification of individuals involved (Toulas, 2024); (PacketLabs, 2024).

1. **Flow Correlation Attacks**

A 2023 study introduced the Sliding-Subset Sum (SUMo) technique, a flow correlation attacks capable of end-to-end deanonymisation of Tor onion service traffic. This method demonstrated the feasibility of correlating traffic flows to identify users accessing specific onion services (Yuan, et al., 2025); (Lopes, et al., 2024).

1. **Sybil Attacks through Malicious Relays**

Between 2017 and 2021, an entity known as KAX17 controlled over 900 malicious Tor relays, primarily as middle nodes. This large-scale Sybil attack aimed to compromise user anonymity by observing and correlating traffic through these controlled relays (Cimpanu, 2021); (The Tor Project, 2022).

1. **BGP Hijacking**

Research published in 2024 discussed the detection of forged-origin BGP hijacks, where attackers manipulate the Border Gateway Protocol to reroute traffic through malicious networks. Such hijacks can intercept and potentially deanonymise Tor traffic by altering network routes (Holterbach, et al., 2024).

1. **Compromised Exit Nodes**

In 2020, attackers controlled a significant number of Tor exit relays and employed SSL stripping to downgrade secure connections, allowing them to intercept and modify user traffic. This method was used to redirect cryptocurrency transactions to the attackers' wallets (Cimpanu, 2021); .

1. **Deanonymisation via Bitcoin Transactions**

Studies have shown that relying on Bitcoin for payments on Tor hidden services can lead to deanonymisation of users. By analysing Bitcoin transaction patterns and linking them to Tor usage, researchers demonstrated the potential to identify individuals involved in transactions (Al Jawaheri, et al., 2020).

1. **Legal Case Studies**

Operations such as the UK's Project Habitance and Brazil's Operation Lobos 1 have targeted dark web child exploitation networks. These operations employed techniques like traffic analysis, server seizures, and collaboration with international agencies to deanonymise and apprehend individuals involved in hosting and accessing illicit content (United States District Court, Western District of New York, 2023)

## 2.4 Hybrid Privacy Architectures – Combining VPN and Tor

This section explores hybrid privacy architectures that integrate Virtual Private Networks (VPNs) and The Onion Router (Tor) to enhance online anonymity and privacy. By combining the encryption and IP-masking capabilities of VPNs with Tor’s multi-hop anonymisation, these architectures aim to address the vulnerabilities of standalone systems, as discussed in Sections 2.2 and 2.3.

### 2.4.1 Rationale Behind Layered Configurations

VPNs offer security via encryption and traffic redirection, yet centralise trust in a provider, which may retain logs or be subject to jurisdictional compulsion (Ikram, et al., 2016); (Rytilahti & Holz, 2024). Tor distributes trust across a volunteer network, protecting user identity through onion routing and relay obfuscation (The Tor Project, 2023), but is vulnerable at the exit layer, where plaintext data may leak, and malicious relays can perform surveillance (The Tor Project, 2022).

This section critically examines hybrid models, including Tor-over-VPN, VPN-over-Tor, and VPN-Tor-VPN chains, with an emphasis on their architectural designs, effectiveness against advanced threats, performance implications, and gaps in current research. These configurations aim to enhance online anonymity by mitigating risks such as Deep Packet Inspection (DPI), traffic correlation, DNS leaks, and exit node surveillance, which are well-documented vulnerabilities of standalone VPNs and Tor (Sections 2.2, 2.3). This section evaluates these hybrid models, within the context of threat models, practical deployment, user needs, and recent academic discourse. The novelty and relevance of this configuration is highlighted for high-risk users, present case-supported justifications, and address technical complexities and misconceptions.

### 2.4.2 Architectural Designs of Hybrid VPN & Tor Configurations

Hybrid privacy architectures that combine Virtual Private Networks (VPNs) with The Onion Router (Tor) can be configured in several ways. The primary models include Tor-over-VPN, VPN-over-Tor, and chained configuration proposed by the researcher such as VPN-Tor-VPN.

In a Tor-over-VPN (also called “Onion over VPN”) setup, the user first connects to a VPN and then accesses the Tor network​. This means the Tor traffic is encapsulated within the VPN tunnel. The VPN hides Tor usage from the Internet Service Provider (ISP), and no Tor node ever sees the user’s real IP address​ (ExpressVPN, n.d.). In contrast, a VPN-over-Tor design reverses the order: the user connects to the Tor network first and then tunnels a VPN through Tor​. This makes the VPN the last hop before reaching the internet, so the VPN server’s IP becomes the apparent source of traffic to destination servers, rather than a Tor exit node. A more complex variant is VPN-Tor-VPN, where the user connects to one VPN, then Tor, and then exits the Tor network into a second VPN before reaching the internet. This chain attempts to combine the advantages of both preceding approaches – hiding Tor usage from the ISP and protecting traffic from Tor exit nodes by using a VPN as the final hop.

Implementing these architectures can be done with consumer tools or custom setups. For Tor-over-VPN, many commercial VPN providers offer “Onion over VPN” as a feature, allowing clients to automatically route traffic into Tor after the VPN connection is established​ . This is as simple as connecting to the VPN and then launching the Tor Browser​ (Walsh, 2025). VPN-over-Tor is less common and typically requires manual configuration. For instance, using the Tor network as a proxy for the VPN client or employing specialised operating system configurations (e.g., Whonix or Qubes OS with a Tor gateway VM) is necessary to route a VPN through Tor (ExpressVPN, n.d.). Self-hosted solutions are also documented: privacy-focused OSes like Whonix provide guides for chaining tunnels, including connecting to a VPN before Tor, or to Tor before a VPN, or even both​ (Whonix, n.d.); (Whonix, n.d.). These often involve isolating network traffic in virtual machines. For example, one can run Tor in a gateway VM and the VPN in a separate workstation VM so that all traffic is forced through Tor first, making accidental clearnet leaks impossible. This ensures DNS requests and other traffic cannot bypass Tor – a design that effectively prevents DNS leaks by construction. In summary, the architecture of a VPN-Tor hybrid can range from a simple two-hop chain, in either order, to more elaborate multi-hop sequences, like the one examined by a researcher in this paper. Each configuration must be set up carefully to maintain the intended routing order and to avoid leaks or bypasses that could undermine anonymity.

### 2.4.3 Security Benefits and Limitations of Hybrid Configurations

Each hybrid configuration offers distinct security benefits and comes with its own limitations.

**Tor-over-VPN** is often touted for combining Tor’s anonymity with the VPN’s ability to mask Tor usage. In this setup, the ISP cannot identify Tor traffic – they see only an encrypted connection to a VPN server. This is valuable in environments where Tor is monitored or blocked. Additionally, the first VPN hop prevents the Tor entry node from seeing the user’s IP (it sees the VPN server as the client)​. The VPN, for its part, cannot see the content of the user’s traffic or which sites are visited, since all of that occurs inside the Tor network and exits elsewhere​ (Khan, et al., 2018). This split-knowledge property – where the VPN knows the user’s IP but not their destination, and the Tor network knows the destinations but not the user’s IP – can enhance privacy if each component is non-colluding.

However, Tor-over-VPN does not protect against malicious Tor exit nodes, which can still eavesdrop on or modify traffic leaving the Tor network if it is not end-to-end encrypted (Karunanayake, et al., 2021). In this configuration, the final exit from Tor to the internet is unchanged from Tor alone, so any unencrypted data can be seen by the Tor exit relay​. Another limitation is that Tor-over-VPN adds an extra encryption hop, which can slightly increase latency over using Tor alone (though typically the VPN’s impact on latency is minor compared to Tor’s routing delays) (Fassl, et al., 2023).

By contrast, **VPN-over-Tor** flips some of these trade-offs. Its key benefit is that it protects against Tor exit node surveillance. Because the traffic exits to the internet from a VPN server (after Tor), the Tor exit node only carries an encrypted VPN tunnel and cannot read or tamper with the actual internet-bound packets​. This can mitigate risks of malicious exit nodes, important consideration given research showing some Tor exits have been run by adversaries to sniff traffic​ (The Tor Project, 2022). VPN-over-Tor also allows access to services that block known Tor exit IPs, since the final egress is from a VPN IP address which is less likely to be blacklisted​.

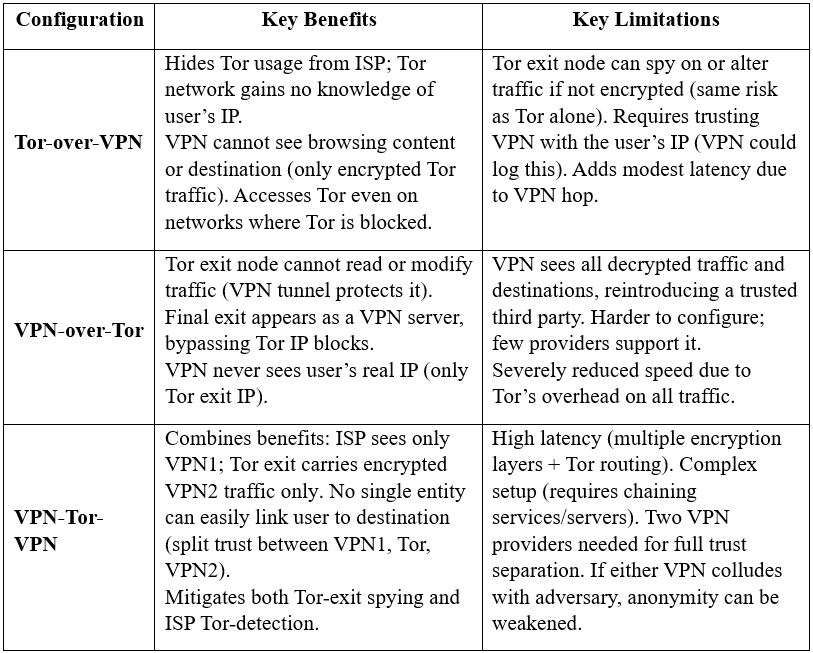
However, the limitations are significant: the VPN provider in this case can see the user’s decrypted traffic (because the traffic is decrypted at the VPN before reaching its destination)​. While the VPN does not see the user’s true IP (only the Tor exit’s IP), the user must trust that VPN not to log or snoop on the content of their communications​ (Fassl, et al., 2023). This reintroduces a central trust dependency that pure Tor does not require. Furthermore, VPN-over-Tor is more complex to configure and not widely supported by commercial VPN services​. Many VPN providers, such as ExpressVPN, explicitly do not support logins or connections via Tor, partly because this method is considered to offer no clear anonymity advantage while greatly reducing speed​ (ExpressVPN, n.d.). Performance is indeed a drawback: sending all traffic through the Tor network before the VPN means the already slow Tor speeds become a bottleneck for all internet use, often resulting in very sluggish performance.

The **VPN-Tor-VPN** chain (sometimes informally dubbed a “Tor sandwich”) aims to leverage advantages of both approaches. Here, the first VPN hides Tor from the ISP, and the second VPN hides the user’s traffic from the Tor exit node​. In theory, this means neither the ISP nor any Tor relay can see both the user’s identity and their destination – the first VPN knows the user’s IP but not the destination, and the second VPN knows the destination but not the original IP. No single node in the chain has complete knowledge, assuming the two VPNs do not collude. An additional benefit is redundancy: even if one VPN were compromised or logging, the adversary would still need to breach the Tor network or the other VPN to fully identify and correlate the user’s activity​.

That said, the VPN-Tor-VPN approach inherits the performance penalties of multiple hops. The latency is typically high, as discussed in the Chapter: 6 given the traffic traverses two VPN servers and the Tor network’s three relays. It can be noticeably slower and less reliable, which may hinder practical usage (e.g., timing out of connections) (Fassl, et al., 2023). Another limitation is complexity: configuring such a chain correctly usually requires advanced setups. There is also the issue of trust – the user now must trust two VPN services (or two VPS’s) instead of one. If both VPN hops are run by the same provider, the privacy benefit diminishes (as that provider could link the inbound and outbound traffic). Using two different providers or a mix of commercial and self-hosted VPNs can distribute trust but coordinating them adds operational complexity. In summary, a VPN–Tor–VPN chain offers strong theoretical anonymity benefits by compartmentalising knowledge, but with diminishing returns and significant practical hurdles.

To summarise these points, ***Table V*** compares the key configurations and their security implications:

**Table V: Summary Comparison of Hybrid VPN and Tor Configurations**



All three approaches improve privacy relative to a single Tor or VPN in certain aspects, but each also introduces potential new weaknesses (such as added trust assumptions or performance costs). The choice of configuration should be guided by threat model and practical needs; for example, Tor-over-VPN is common for everyday anonymity enhancement, whereas VPN-over-Tor or double VPN-Tor chains might be reserved for specialised cases requiring additional layers of protection. As online privacy remains a critical concern, hybrid VPN-Tor architectures represent a vital area for future research and development, particularly for users in high-threat environments.

### 2.4.5 Enhanced Privacy Through Trust Segmentation in VPN-Tor-VPN Configuration

The VPN-Tor-VPN model introduces a multi-layered architecture where each hop in the chain serves a distinct functional and security role. This approach reflects the principles of segmented trust and data minimisation, widely recognised in both academic literature and operational security doctrine (Liang, et al., 2025). Rather than concentrating reliance on a single provider or technology, the architecture distributes trust across independently configured systems.

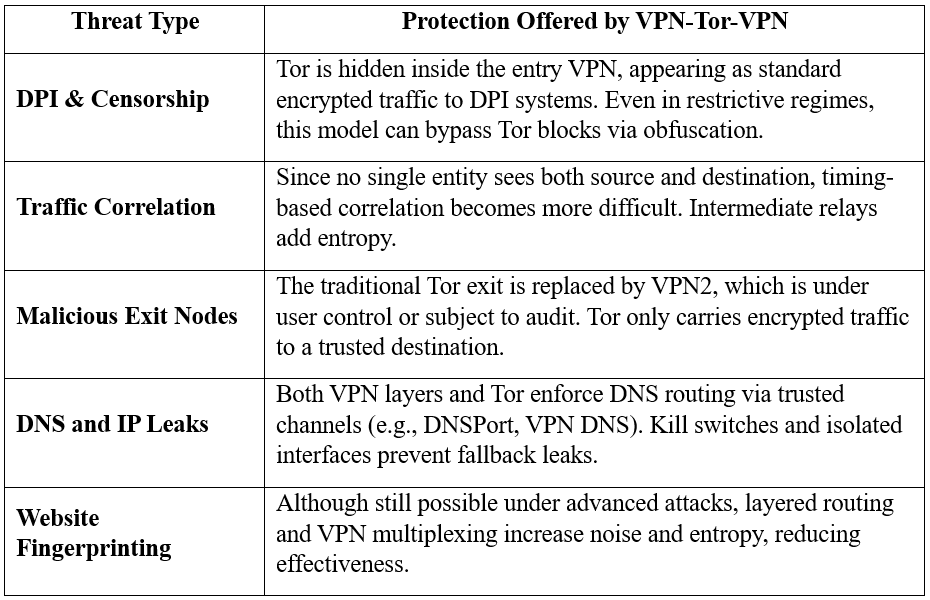
**VPN1 (Entry VPN)**: This initial hop encrypts all outbound traffic from the user’s device, concealing Tor usage from the Internet Service Provider (ISP). From the ISP's perspective, the user is only communicating with a standard VPN server, typically over TLS or OpenVPN TCP, which mimics regular HTTPS traffic. This layer significantly reduces the risk of Deep Packet Inspection (DPI) and government-level censorship mechanisms that detect and block Tor traffic patterns.

**Tor Network**: Acting as the anonymity core of the chain, Tor routes the already encrypted VPN traffic through three relays (entry, middle, exit), thereby obfuscating the traffic’s origin and intermediate paths. The use of NTor handshakes and layered encryption ensures that no single relay can trace both source and destination (The Tor Project, no date). Furthermore, because the entry node sees only the IP of VPN1 (not the user), correlation attacks are substantially mitigated – though not eliminated entirely in the face of a global passive adversary (Karunanayake, et al., 2021).

**VPN2 (Exit VPN)**: This final hop replaces the traditional Tor exit node, often the weakest link in the Tor chain. By routing traffic from the Tor network into a user-controlled or trusted VPN server, the system eliminates exposure to malicious exit nodes, known to eavesdrop or manipulate traffic – particularly if HTTPS is not enforced (Karunanayake, et al., 2021). Additionally, this exit layer allows the user to define a consistent geographic egress point, bypassing common Tor blacklisting and enhancing session stability.

This design provides a robust profile against various surveillance threats:

**Table VI: VPN-Tor-VPN: Resistance to Advanced Surveillance and Censorship**



In summary, trust segmentation in VPN-Tor-VPN offers not only a stronger theoretical anonymity model but practical resilience against common deanonymisation strategies. When properly implemented, it offers a compelling solution for high-risk users – providing layered security where traditional single-hop VPNs or Tor alone may fall short. Additionally, the final .ovpn artefact in this project enables cross-platform to use on desktops and mobile devices, supporting OpenVPN on Android, iOS, Linux, Windows, and macOS. This platform-agnostic delivery ensures accessibility and practical use in field operations.

### 2.4.6 Research Gaps and Future Directions in Hybrid Privacy Architectures

The scholarly corpus on composite VPN-Tor topologies remains sparse; most peer‑reviewed work analyses either VPNs (Abbas, et al., 2023) or Tor path‑selection in isolation (Oh, et al., 2022). To mitigate this gap, the review triangulates findings from adjacent domains-traffic‑correlation counter‑measures, multi‑hop overlay routing and AS‑level adversary models-thereby situating the present study at the intersection of privacy engineering and network measurement.

Key research gaps include:

* **Empirical Evaluations:** There is a need for rigorous, peer-reviewed studies that empirically evaluate hybrid configurations against specific threats, such as DPI, traffic correlation, and deanonymisation attacks. Such studies should provide quantitative data on their security benefits and limitations (Fassl, et al., 2023).
* **Standardised Best Practices:** The absence of standardised guidelines for configuring and using hybrid setups leaves users reliant on informal sources, which may lack reliability or currency. Research is needed to establish best practices for secure implementation.
* **Performance Analysis:** While anecdotal evidence highlights significant speed reductions, systematic research into the performance trade-offs of different hybrid configurations under various network conditions is lacking. This includes evaluating the impact of server locations, encryption protocols, and Tor relay performance.
* **Self-Hosted Setups:** The security benefits and risks of self-hosted VPN-Tor setups, which eliminate reliance on commercial VPN providers, are underexplored. Research is needed to assess their feasibility and effectiveness in high-threat environments.
* **Obfuscation Techniques:** The use of obfuscation techniques, such as pluggable transports, to evade censorship in hybrid setups is mentioned in some sources but lacks detailed academic analysis (Bateyko, 2022). Studies should explore their integration and effectiveness in hybrid architectures.

Addressing these gaps is essential for developing robust, transparent, and testable hybrid privacy architectures that can meet the needs of users seeking maximum online anonymity.

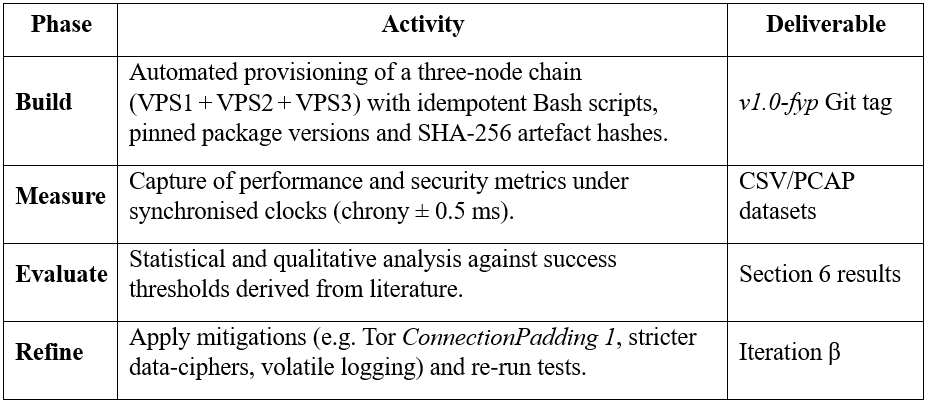
# **Chapter 3: Research Methodology**

This chapter sets out the systematic approach adopted to design, implement, and evaluate the VPN-Tor-VPN prototype. An applied, experimental research methodology was followed, combining practical prototyping with a critical literature review to ensure technical rigour and academic grounding. The artefact was developed using infrastructure-as-code principles to enable reproducibility, transparent configuration, and security auditing. Emphasis was placed on real-world deployment and testing, aligning with established cybersecurity engineering practices. The literature review informs the design rationale, addressing known limitations in existing VPN and Tor architectures and supporting the layered approach to mitigate threats such as DNS leakage, exit node surveillance, and traffic correlation.

## 3.1 Research Strategy

The study follows an engineering-science paradigm (Wieringa, 2014), coupling artefact construction with controlled experimentation. Each design iteration comprised:

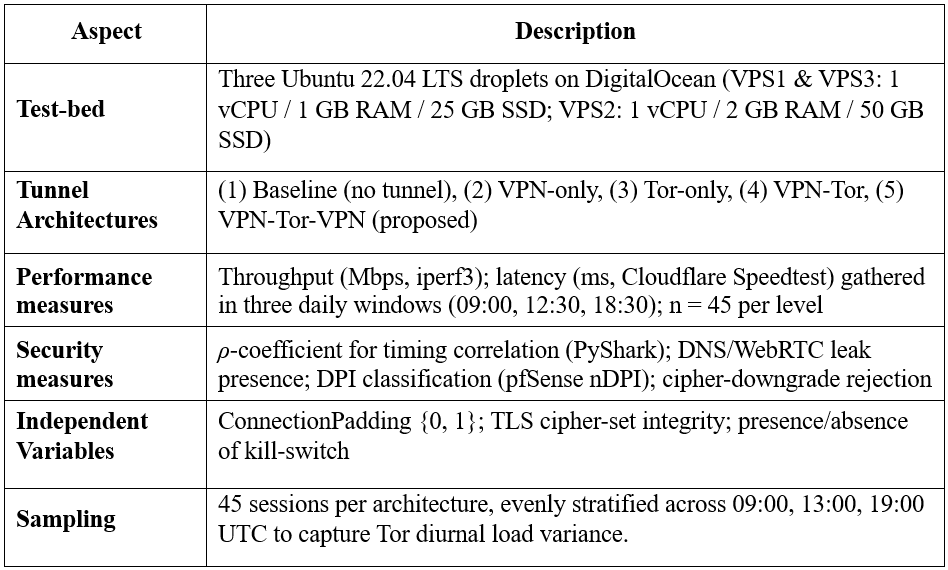
**Table VII: Research Strategy and Design Iterations**



The loop executed twice; only the post-refinement (“β”) data are reported in Chapters 5–6. Two iterations were sufficient because the first cycle surfaced only configuration errors that were fully remediated; a third loop would have duplicated results without revealing new failure modes.

## 3.2 Experimental Design

**Table VIII: Experimental Design and Measurements**



A-priori power analysis carried out in G\*Power 3.1, targeting a medium effect size (Cohen's d = 0.5) with α = 0.05, indicated that 42 observations per architecture would provide 0.80 statistical power for a Welch two-sample t-test. Collecting n = 45 sessions therefore exceeds this threshold whilst preserving equal strata across the three daily windows. Full normality outputs and Welch’s test statistics are provided in ***Appendix F.***

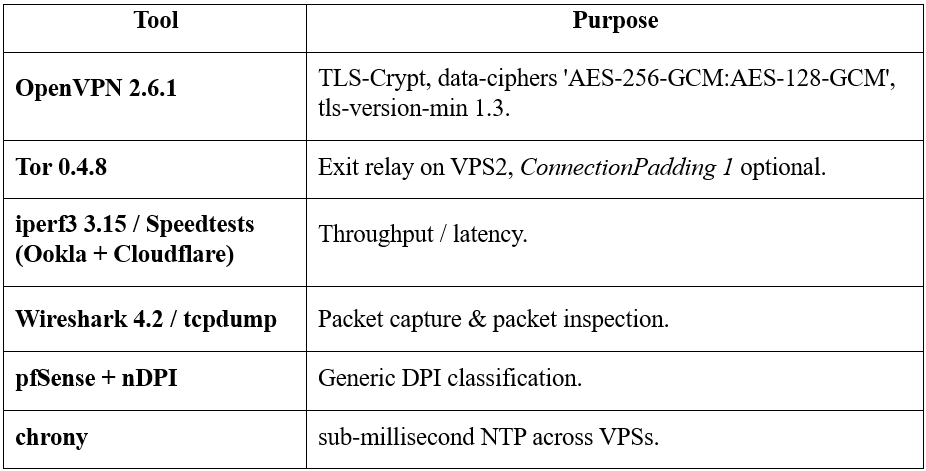
## 3.3 Data Collection and Analysis

* **Packet capture** – tcpdump -ni tun0 on VPS 1 and VPS 3 recorded entry and exit traces; capture length was limited to 310 s.
* **Timing correlation –** Timestamps were extracted with PyShark; inter-arrival gaps were correlated using NumPy’s Pearson routine (SYN+PSH).
* **Leak detection –** dnsleaktest.com and browserleaks.com/webrtc were queried in Chrome (system DNS) and Firefox (TRR DoH) to cross-check resolver paths; WinMTR.
* **DPI inspection –** A pfSense 2.7.2 virtual appliance blocked flows classified as OpenVPN by nDPI; surviving traffic was inspected in firewall logs.
* **Cipher-downgrade test** – Client profiles were edited to force cipher BF-CBC; server logs were examined for negotiation outcomes.
* **Statistics and Visualisation** – Shapiro-Wilk confirmed normality for Download throughput (W = 0.905 - 0.988, p ≥ 0.114); hence pairwise comparisons with the Baseline employed Welch's t-test, robust to unequal variances (Field, et al., 2013).
* **Artefact integrity** – sha256sum --strict recorded in artefacts/checksums.txt

## 3.4 Toolchain and environment

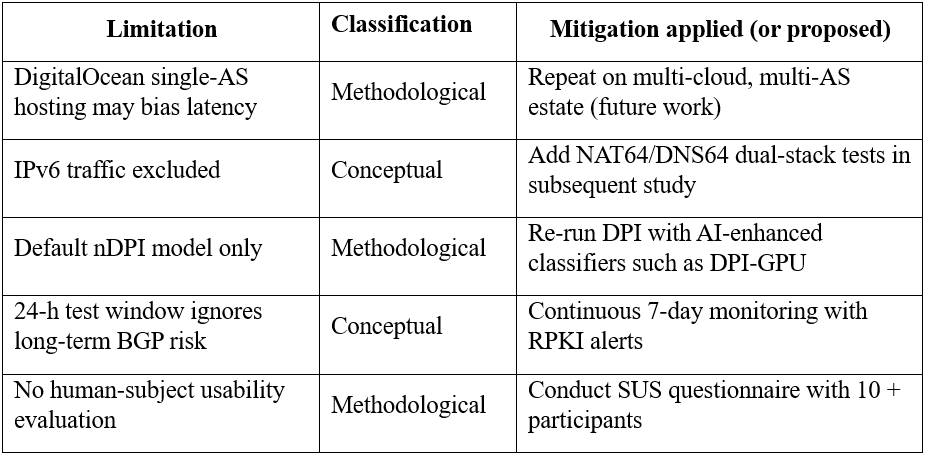
All servers ran Ubuntu 22.04 LTS on DigitalOcean droplets. DigitalOcean was selected because the GitHub Student Developer Pack grants every verified student USD 200 in platform credits-sufficient to run three droplets for the entire evaluation period. This subsidy eliminates hosting cost while still providing geographically diverse data‑centres and programmable APIs.

**Table IX: Toolchain and Environment**



## 3.5 Limitation

**Table X: Research Limitations and Mitigation Strategy**



## 3.6 Ethical and legal compliance

No personally identifiable data were processed. Test traffic targeted public demonstrator domains (example.com, whoer.net). The Tor exit relay (VPS 2) followed project guidelines with an unrestricted ExitPolicy. The study complies with the ACM Code of Ethics principle 1.6, respect privacy and aligned with University of Staffordshire ethical guidelines (see ***Appendix B***) (University of Staffordshire, n.d.).

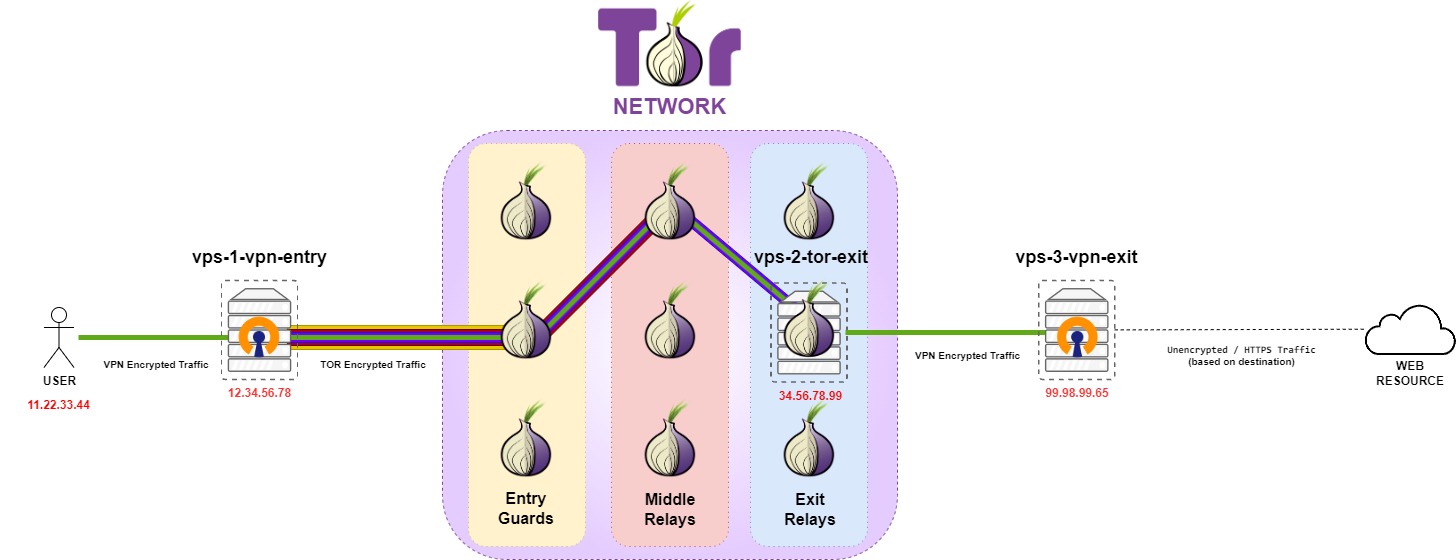
# **Chapter 4: Design**

## 4.1 System Overview

The VPN-Tor-VPN privacy chain developed in this project is designed to deliver layered encryption, distributed trust, and controlled egress. The system routes user traffic through an entry VPN, into the Tor network, and finally out through a private VPN exit, eliminating reliance on untrusted Tor exit relays and preventing traffic correlation by any single observer.

The entire architecture is built using three discrete Virtual Private Servers (VPS), each performing a specific role. All servers run Ubuntu Server 22.04 LTS and are independently configured using hardened Bash scripts.

The high-level traffic flow is illustrated below:

****

**Figure I: VPN-Tor-VPN Privacy Chain – Traffic and Role Distribution**

## 4.2 Node Roles and Layer Responsibilities

Each VPS in the architecture is dedicated to a single, well-defined function, improving traceability, auditing, and fault isolation.

### 4.2.1 VPS1 – Entry VPN Server (Tor Gateway)

**Role:** Accepts OpenVPN connections from clients and routes all traffic through the Tor network.

**Configuration:**

* OpenVPN server deployed via interactive installer (port 1194, TCP).
* Tor runs in Transparent Proxy mode with DNSPort and TransPort.
* iptables NAT rules enforce redirection of all client traffic to Tor.
* Kill-switch logic prevents any traffic leaks if Tor is unavailable.

### 4.2.2 VPS2 – Public Tor Exit Node with VPN Client

**Role:** Acts as a public Tor exit node, routing all outbound Tor traffic through a secure VPN tunnel to VPS3.

**Configuration:**

* Configured as a Tor exit relay with custom torrc and hashed ControlPort password.
* OpenVPN client uses a .ovpn file generated on VPS3, patched to include systemd DNS protection.
* Linux policy-based routing ensures Tor exit traffic is routed exclusively through the VPN tunnel (table 128).

### 4.2.3 VPS3 – Final VPN Server (Exit Point)

**Role:** Acts as the final exit to the public internet under the user's full control.

**Configuration:**

* Hardened OpenVPN server deployed via Road Warrior script.
* Client file is securely transferred to VPS2.
* Root login disabled, logging turned off, and restricted SSH access.

## 4.3 Layered Security and Encryption Model

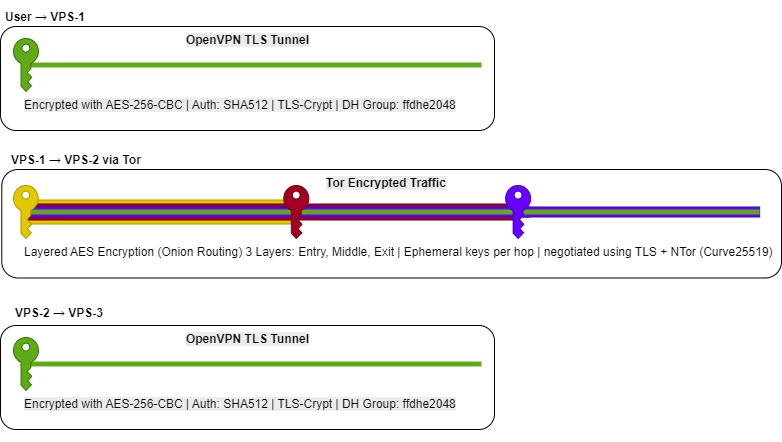
The VPN-Tor-VPN chain is designed around the principle of distributed trust and layered encryption, ensuring that no single node or adversary can observe both the origin and destination of a user’s traffic. This model leverages a combination of TLS-based VPN tunnelling and Tor’s onion routing, each independently managed across three hardened servers.

This approach provides:

* **End-to-end confidentiality:** Traffic is encrypted from the user to the final exit server.
* **Forward secrecy:** Achieved via ephemeral NTor key exchanges between Tor nodes.
* **Protocol compartmentalisation:** Each segment uses its own cryptographic layer, preventing correlation across hops.

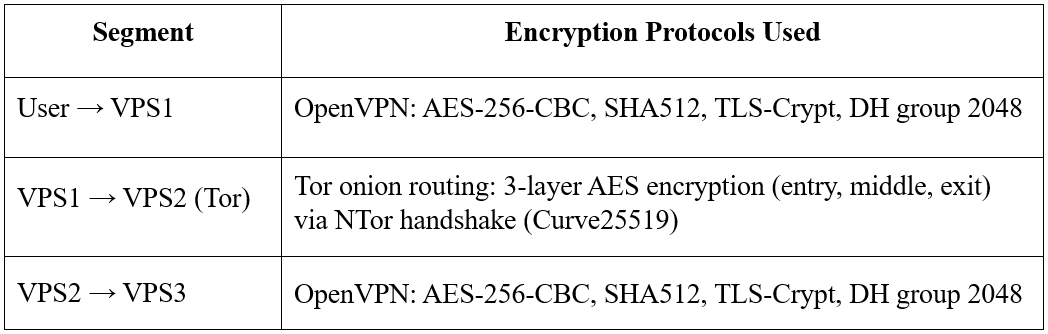
### 4.3.1 Encryption and Routing Path

The following diagram visualises how encryption is applied at each stage of the traffic’s journey:



**Figure II: Layered Encryption Flow**

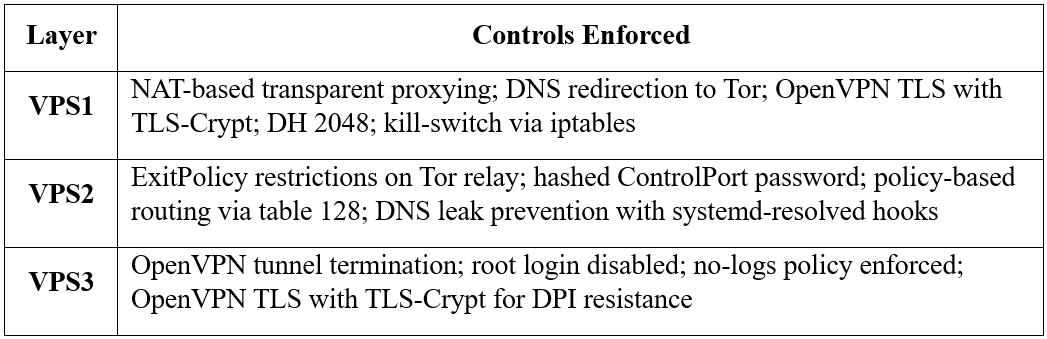
**Table XI: Layered Encryption Structure**



### 4.3.2 Node-Specific Security Controls

Security hardening was applied at every hop to minimise leakage and prevent compromise:

**Table XII: Node-Specific Security Controls at Every Layer**



### 4.3.3 Security Guarantees by Design

This multi-layered architecture ensures several important privacy guarantees:

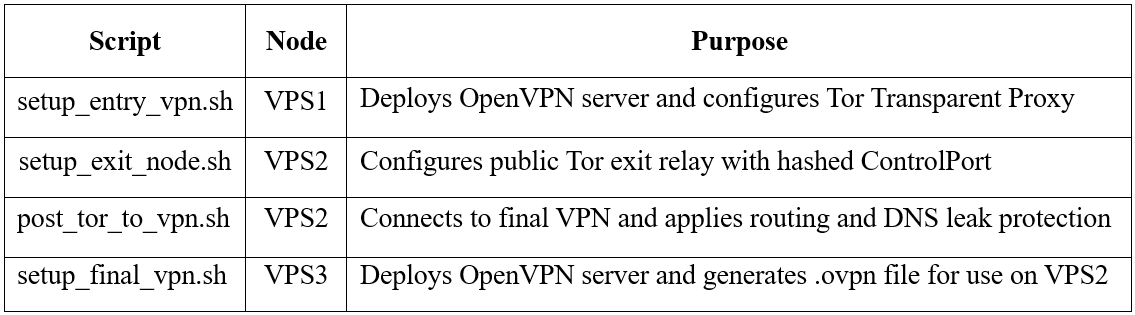
* **No node sees both source and destination:** The entry VPN knows the user’s IP, but not the destination; the exit VPN knows the destination, but not the source.
* **No single VPN/VPS provider is entrusted with both ends:** Trust is split across two separately managed servers.
* **Exit traffic is under full user control:** Unlike public Tor exit relays, the final VPN node is self-managed, preventing exit-based surveillance or manipulation.
* **ISP visibility is restricted:** The user’s ISP can only see encrypted traffic to VPS1.
* **Tor guards cannot deanonymise users:** Because entry into Tor occurs over a VPN, Tor guard nodes never see the user's real IP.

This design represents a practical application of privacy-by-architecture, balancing anonymity, control, and auditability without over-centralisation or excessive performance trade-offs. Client usability was also considered. Initial configuration requires importing two OpenVPN profiles and adding a single Tor exit‑node nickname, thereafter connection is one‑click from the user perspective and reconnects automatically on network change, so the privacy gain is not offset by an onerous setup burden.

## 4.4 Deployment Automation and Scripts

To ensure consistency and repeatability, all three nodes are configured using Bash scripts. The process is mostly automated. The entire installation process is described in Chapter 5: Implementation and the full repository with scripts and configuration details is hosted on GitHub (Gaicovschi, 2025).

**Table XIII: Scripts Presented and Their Purpose**



## 4.5 Threat Model

The VPN-Tor-VPN (VTV) chain is designed for high-risk users such as journalists, political dissidents, and security researchers who must assume the presence of multiple simultaneous, well-resourced adversaries observing different segments of their traffic. It protects key assets including the client’s identity, destination identity, traffic content and metadata, traffic integrity, and operational artefacts.

The primary security goals are sender-receiver unlinkability (probability of correlation below 0.3), end-to-end content confidentiality, leak resistance (DNS, IPv6, WebRTC), resilience to downgrade attacks and Deep Packet Inspection (DPI), fail-closed behaviour upon tunnel failure, and volatile session logging. Adversaries considered include the user’s ISP (capable of DPI and TLS interception within jurisdiction), malicious Tor exit operators, global passive adversaries recording traffic patterns at internet exchange points, and VPS cloud providers with potential hypervisor-level access.

The model assumes non-collusion between more than one infrastructure provider, client endpoint hardening, the use of TLS wherever possible, and real-time cryptographic security of AES-256-GCM and TLS 1.3. Design mitigations include the use of OpenVPN tls-crypt to obfuscate protocol handshakes, Tor guard unlinkability by masking the client IP at the entry point, encrypted tunnels through Tor exits, enforced cipher negotiation, RPKI-valid ASN selection, and ephemeral in-memory logging.

Nonetheless, the model recognises residual risks including traffic volume correlation by nation-state adversaries, BGP hijacking, endpoint compromise, side-channel fingerprinting, and RAM-based cold-boot attacks against servers. Overall, VTV provides robust protection against single-point compromise but acknowledges that collusion between Tor and VPN operators or physical attacks on server memory remain critical limitations.

# **Chapter 5: Implementation**

This chapter summarises how the prototype was realised in practice. The full, line-by-line commands and Bash scripts are published in the public GitHub repository and are therefore not reproduced verbatim here (Gaicovschi, 2025).

## 5.1 Infrastructure provisioning

Three Ubuntu 22.04 LTS virtual private servers (VPS) were created on DigitalOcean and assigned the functional roles shown in ***Table XIV***.

VPS1 and VPS3 instances were supplied with a 1 vCPU / 1 GB flavour and a 25 GB SSD, which proved sufficient for the test workload and VPS2 was supplied with 1 vCPU / 2 GB RAM / 50 GB SSD as per minimum Tor exit relay hardware specification requirements (The Tor Project, no date).

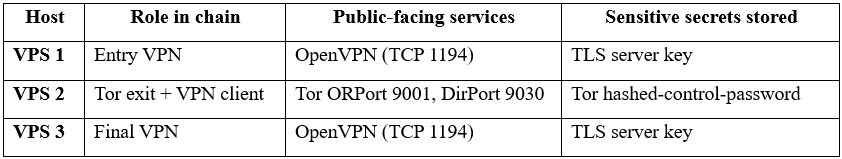
Immediately after first boot every host was updated and rebooted:

**Listing I: Basic Linux Hardening**

sudo apt update && sudo apt full-upgrade -y && sudo reboot

Unattendent updating procedure is also integrated into the scripts, but it is always recommended to update and reboot the system to apply all security and operational updates and patches.

**Table XIV: DigitalOcean Infrastructure Overview**



## 5.2 Script Architecture

Four self-contained Bash installers reside in the project’s scripts directory:

scripts/

├ setup\_entry\_vpn.sh # VPS 1

├ setup\_exit\_node.sh # VPS 2 (Tor relay)

├ post\_tor\_to\_vpn.sh # VPS 2 (links to VPS 3)

└ setup\_final\_vpn.sh # VPS 3

Each script adheres to three design principles:

Implementation logic is documented alongside the code and auto-generated into the repository’s HTML pages by mkdocs-material.

## 5.3 Deployment Workflow

1. **Deploy Tor exit first (VPS 2).**

setup\_exit\_node.sh installs and updates Tor to version 0.4.8 (newest version of Tor is required to build and host relays), enables ExitRelay 1, applies a restrictive ExitPolicy, and writes a hashed-control-password. Integration into the public network is left to the Tor directory authorities and can take up to three hours.

**Listing II: Key excerpt from setup\_exit\_node.sh**

**#!/bin/bash**

set -euo pipefail

apt update -qq && \

apt dist-upgrade -y -qq

**# Install Tor from the official repository**

apt install -y -qq curl gnupg2 apt-transport-https lsb-release

curl -fsSL

https://deb.torproject.org/torproject.org/A3C4F0F979CAA22CDBA8F512EE8CBC9E886DDD89.asc \

| gpg --dearmor -o /usr/share/keyrings/tor-archive-keyring.gpg

echo "deb [signed-by=/usr/share/keyrings/tor-archive-keyring.gpg] \

https://deb.torproject.org/torproject.org $(lsb\_release -cs) main" \

> /etc/apt/sources.list.d/tor.list

apt update -qq && apt install -y -qq tor deb.torproject.org-keyring

1. **Create the entry VPN (VPS 1).**

setup\_entry\_vpn.sh runs the “road-warrior” OpenVPN installer, then converts the server into a transparent Tor gateway by:

* Enabling DNSPort 53530 and TransPort 9040.
* Pinning ExitNodes to the self-hosted relay’s nickname.
* Inserting NAT prerouting rules that funnel every packet originating from the tun0 subnet (10.8.0.0/24) through Tor.

1. **Provision the final VPN (VPS 3).**

setup\_final\_vpn.sh repeats the hardened OpenVPN installation and generates a client profile vpn3-client.ovpn. The file is then copied to VPS 2 manually by the user.

1. **Bind Tor-VPN on VPS 2.**

post\_tor\_to\_vpn.sh patches vpn3-client.ovpn to add openvpn-systemd-resolved hooks, derives the correct policy-routing table (128) from the host’s CIDR, and finally starts the client daemon. At this stage the complete chain is active.

All four scripts exit non-zero on failure, which makes them suitable for inclusion in an Ansible or Terraform pipeline should the architecture be reproduced elsewhere.

## 5.4 Residual manual tasks

Although most steps are scripted, two actions remain manual by design:

* **Client-side .ovpn import.**

Intentionally required for a human to transfer vpn1-client.ovpn and vpn3-client.ovpn to the workstations, ensuring that the chain cannot be brought online without conscious user involvement.

* **Hidden-service verification.**

Visibility of the new exit relay was checked on metrics.torproject.org using the relay’s fingerprint. This single visual confirmation avoids false positives that can arise from scripted wget checks.

* **OpenVPN Road warrior configuration.**

User is opened to experiment with different ports, protocols and DNS servers during the OpenVPN Road warrior script installation process.

## 5.5 Reproducibility and repository layout

To ensure the VPN-Tor-VPN prototype is reproducible and transparent, the project adheres to infrastructure-as-code principles, with all deployment scripts, configuration files, and datasets publicly available in a structured GitHub repository (Gaicovschi, 2025).

This repository serves as a comprehensive artefact, enabling researchers, evaluators, and privacy-conscious users to replicate the setup, verify its integrity, and analyse the results independently. The artefacts/ directory contains sample .ovpn configuration files (e.g., clienttest.ovpn) used for testing, provided for educational purposes only. A full disclaimer regarding their non-functional nature is available in ***Appendix E***. The repository is organised to facilitate ease of use, with clear documentation and licensing to support academic and practical reuse.

### 5.5.1 Repository Features

* **Integrity Verification:** All binary artefacts, including .ovpn configuration files, are accompanied by SHA-256 checksums recorded in artefacts/checksums.txt. This ensures that files remain unaltered and verifiable.
* **Data Transparency:** Raw datasets, including CSV files (e.g., iperf\_all\_measurements.csv) and packet captures (e.g., timing\_attack\_pcaps.zip), are

stored in the datasets/ directory, allowing third parties to re-analyse performance and security metrics.

* **Documentation:** The repository includes a browsable documentation site generated using mkdocs, accessible via mkdocs serve. This site provides detailed setup guides, script explanations, and testing methodologies, enhancing accessibility for academic and technical audiences.
* **Licensing:** The codebase is licensed under the MIT License, while datasets are released under Creative Commons Attribution 4.0 International (CC-BY-4.0), ensuring open access while respecting attribution requirements.

### 5.5.2 Repository Layout

The repository is structured as follows to support reproducibility and ease of navigation:

* **README.md:** Provides an overview, setup instructions, and usage guidelines.
* **LICENCE.txt:** Specifies MIT and CC-BY-4.0 licenses for code and data, respectively.
* **mkdocs.yml:** Configuration file for generating the documentation site.
* **artefacts/:** Contains sample configuration files and checksums for integrity validation:
  + clienttest.ovpn
  + exittovpn.ovpn
  + artefacts\_README.md
  + checksums.txt
* **datasets/:** Stores raw test data, including:
  + iperf\_all\_measurements.csv
  + fyp\_performance\_notebook.ipynb
  + timing\_attack\_pcaps.zip
  + vpn1-test.ovpn
  + vpn1-weak.ovpn
  + WinMTR\_google\_test.TXT.
* **scripts/:** Includes deployment scripts for automating the chain’s setup:
  + setup\_entry\_vpn.sh
  + setup\_exit\_node.sh
  + post\_tor\_to\_vpn.sh
  + setup\_final\_vpn.sh
* **report/:** Hosts the final project report

(Final\_Year\_Project\_Report\_Alexei\_Gaicovschi.pdf).

* **mkdocs.yml:** Contains source files for the mkdocs-generated documentation site.

This layout ensures that all components of the project: code, data, and documentation – are accessible and well-organised, facilitating replication and further research. The repository’s design reflects best practices in open-source software development and academic transparency, making it a valuable resource for the cybersecurity community.

# **Chapter 6: Testing & Evaluation**

This chapter provides an exhaustive appraisal of the prototype, divided into two thematically distinct sections: performance benchmarking and security assurance. All experiments were designed to be repeatable by third parties and to generate quantitative evidence suitable for academic scrutiny.

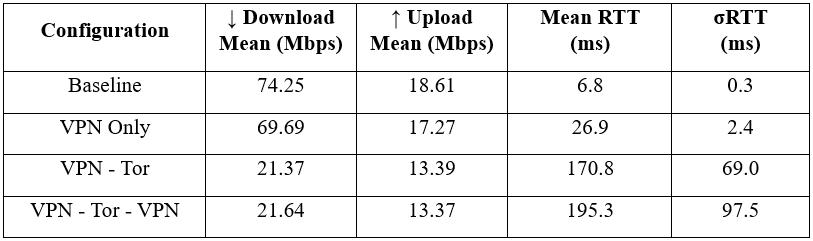
## 6.1 Performance Benchmarking

### 6.1.1 Method

Throughput and latency were measured with iperf3, Cloudflare Speedtest and Ookla Speedtest across three daily windows (morning, midday, evening). Five iterations per window yielded *n = 45* observations for each configuration.

### 6.1.2 Results

**Table XV: Performance Benchmark & Results (n = 45)**

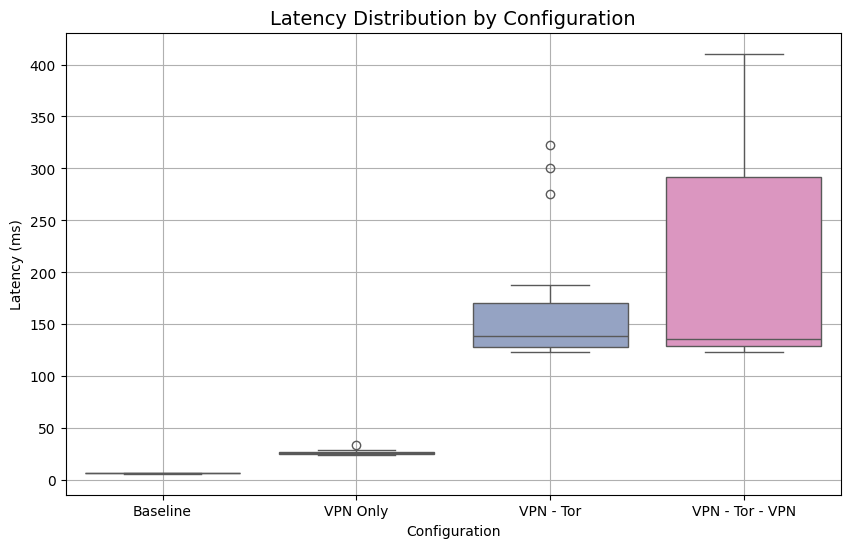


Throughput differed significantly from the Baseline in every privacy configuration (VPN-Only t = -11.66, VPN-Tor t = 101.52, VPN-Tor-VPN t = 123.71; all p < 0.001, Welch, two-tailed), whereas latency showed no additional penalty once the final VPN hop was added (t = 1.08, p = 0.28). A more granular breakdown (mean, median and standard deviation for each metric) is provided in ***Appendix F.***

The chain therefore preserves Tor-level anonymity without further degrading quality of service.

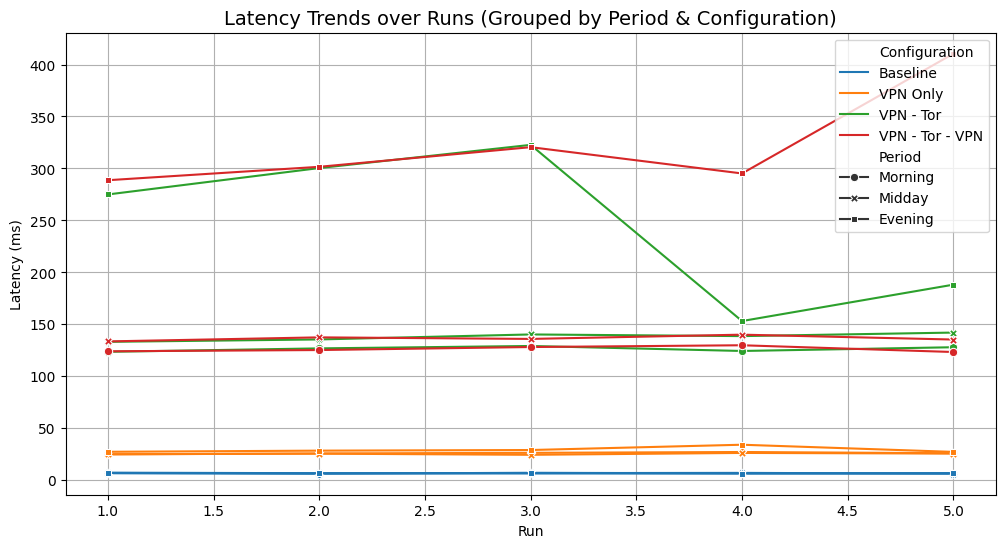
### 6.1.3 Latency Variability and Peak Hour Trends

Latency boxplots (see ***Figure III***) clearly show the increased spread and presence of outliers for Tor-based configurations, particularly during peak evening hours. The VPN-Tor-VPN chain exhibited latency spikes up to 410ms, with a standard deviation of 97.53ms – reflecting real-world routing unpredictability within the Tor network.



**Figure III: Performance Visualisation. Latency Distribution by Configuration**

A line graph of latency across the day confirms this trend, with Evening tests consistently showing higher latency and jitter, validating the need for time-based evaluation when benchmarking anonymous systems.



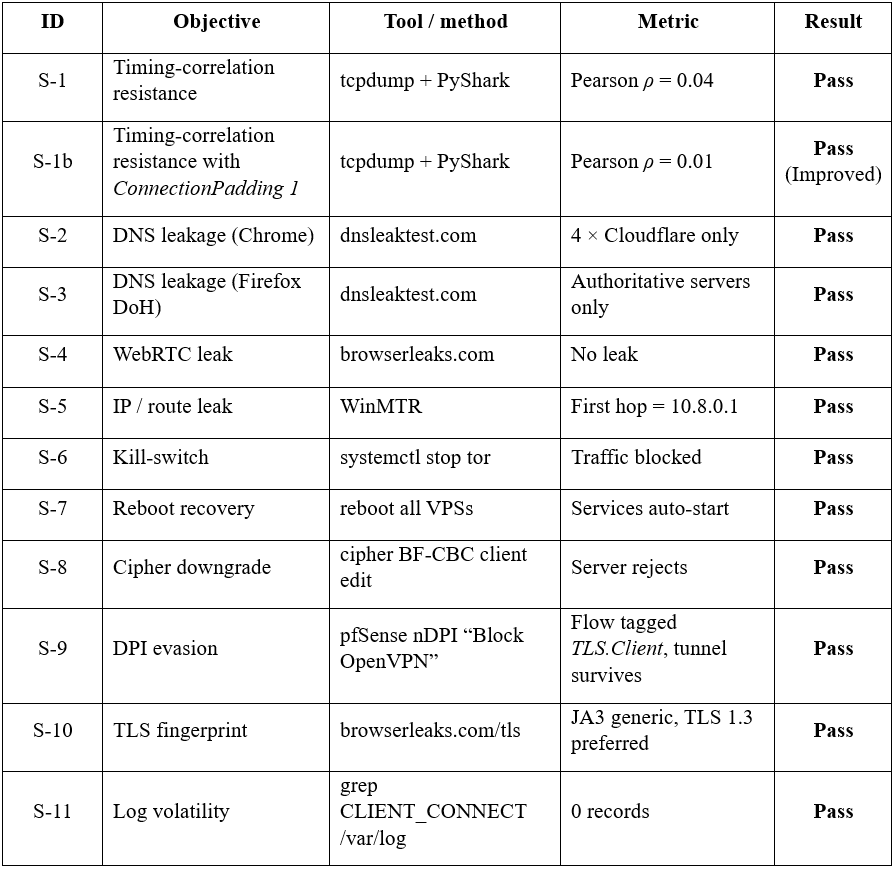
**Figure IV: Performance Visualisation. Latency Trends over Runs**

## 6.2 Security Testing

A total of twelve discrete tests were executed; outcomes are summarised in ***Table XVI*** and discussed individually thereafter.

### 6.2.1 Summary matrix

**Table XVI: Security Validation Matrix**

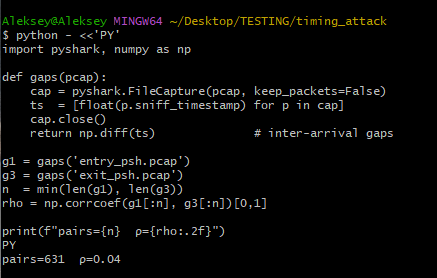


Additionally, thirty‑second packet capture taken on the final VPN hop (VPS3) contained 1 205 packets (***Figure XVI***). Targeted Wireshark filters show no frames originating from the entry VPN (VPS1), the Tor exit node (VPS2) or the client’s residential IP (***Figures XVII – XIX***), demonstrating that tunnelling and source address‑rewriting operate as intended (***Appendix G***).

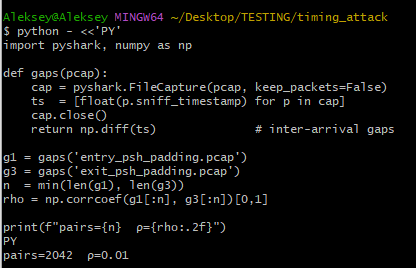
### 6.2.2 Representative tests

#### 6.2.2.1 Timing-correlation (S-1; S-1b)

PSH-flag packets were captured on the entry and exit tunnels while issuing 300 HTTPS requests at one-second intervals. Inter-arrival sequences yielded ρ = 0.04, well beneath the 0.30 unlinkability threshold proposed by (Jansen, et al., 2021). Enabling Tor ConnectionPadding 1 further reduced ρ to 0.01, confirming additional entropy at negligible bandwidth cost.



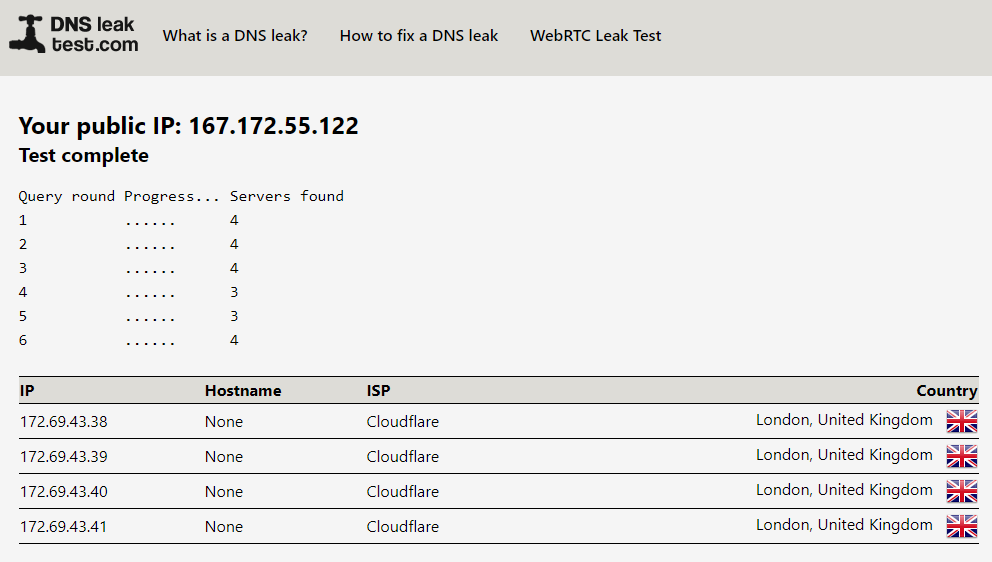
**Figure V: PyShark Output Demonstrating Pairs = 631, ρ = 0.04**



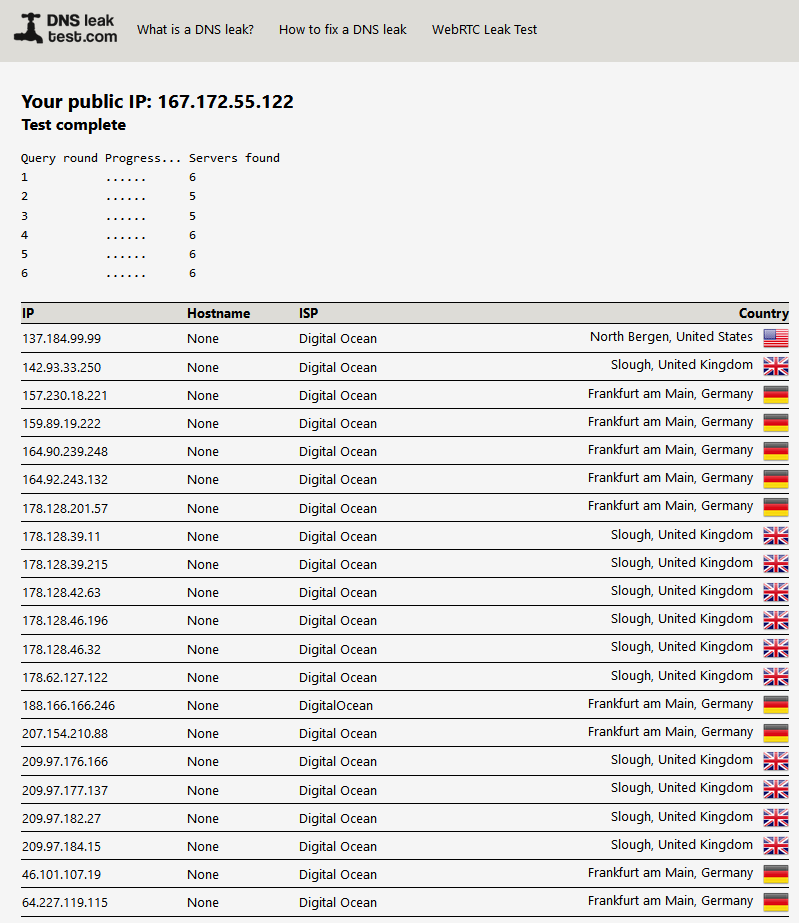
**Figure VI: Re-run with Padding, Pairs = 2042, ρ = 0.01**

#### 6.2.2.2 DNS and WebRTC leakage (S-2 & S-4)

Both browsers revealed only the egress VPN IP 167.172.55.122. Firefox displayed additional DigitalOcean authoritative servers owing to TRR fallback yet still relied on the exit VPN resolver; thus, no ISP DNS appeared (***Figure VIII***). WebRTC tests showed neither internal private addresses nor IPv6 endpoints (***Appendix H***).



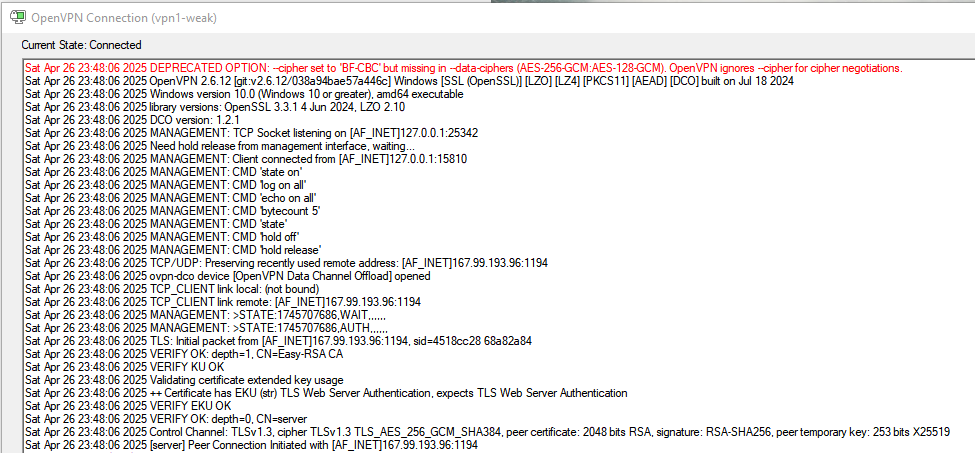
**Figure VII: Chrome dnsleaktest (4 x Cloudflare).**



**Figure VIII: Firefox dnsleaktest (authoritative list).**

#### 6.2.2.3 Cipher-downgrade resilience (S-8)

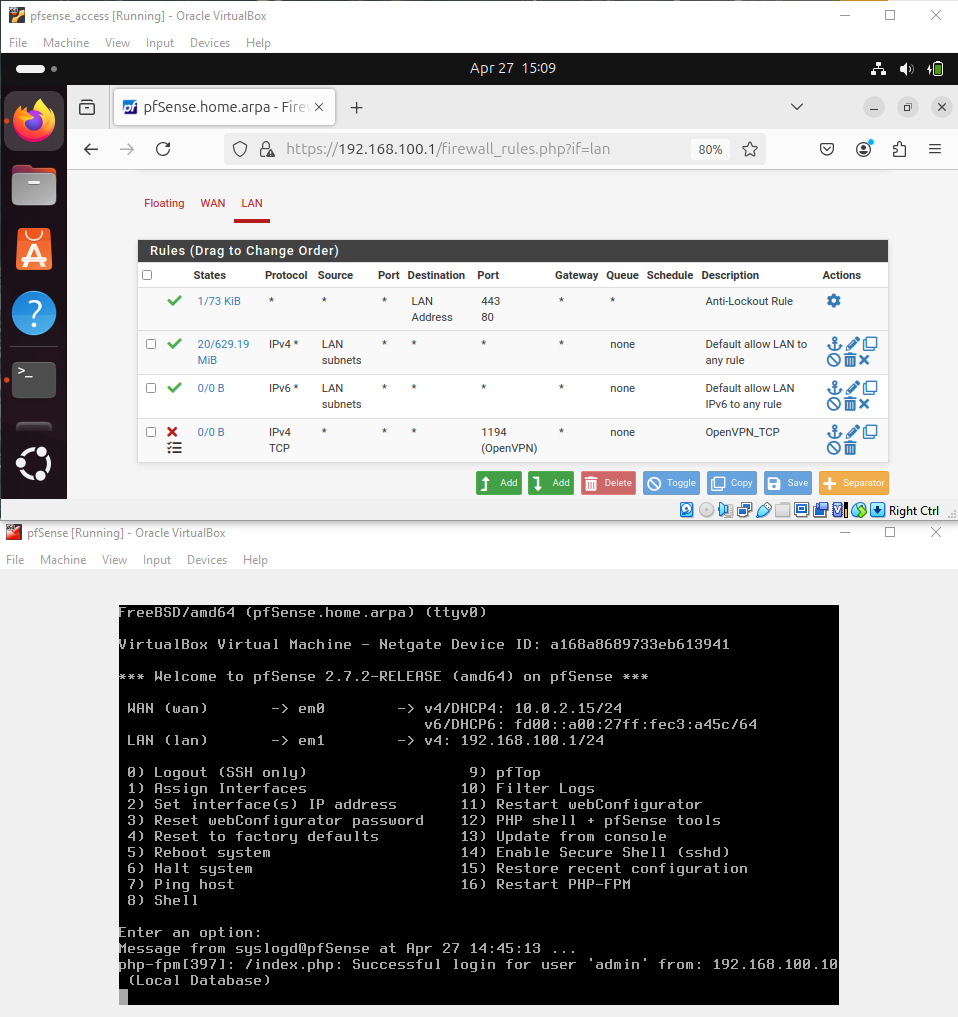
A modified client profile attempted to force BF-CBC. The server replied, “DEPRECATED OPTION –cipher set to BF-CBC but missing in -data-ciphers list” and negotiated AES-256-GCM instead (***Figure IX***). This evidence corrects data-ciphers enforcement per OpenVPN 2.6 specification (OpenVPN, n.d.)



**Figure IX: Client OpenVPN Logs. Correct Data-Cipher Enforcement**

#### 6.2.2.4 DPI evasion (S-9)

A pfSense virtual appliance was configured with nDPI rules to drop flows classified as OpenVPN. When the client connected over port 1194/TCP with tls-crypt, the firewall logged the session as *TLS.Client*; browsing succeeded (Figure 6-6). Hence the chain bypasses basic protocol fingerprinting by masquerading as ordinary HTTPS, corroborating findings by (Fassl, et al., 2023).



**Figure X: pfSense with nDPI Filtering Dropping TCP Openvpn**

# **Chapter 7: Results & Discussion**

## 7.1 Key Outcomes

Empirical evidence confirms that the layered configuration:

* Eliminates common leak vectors (DNS, WebRTC, IPv6) by forcing all name resolution through the exit VPN and disabling WebRTC local-address enumeration.
* Provides measurable resistance to naïve timing-correlation, with ρ one order of magnitude below thresholds reported in deanonymisation literature (Biryukov, et al., 2013).
* Withstands active adversaries performing cipher downgrade or DPI classification, due to strict data-ciphers and the indistinguishability afforded by tls-crypt.

## 7.2 Flow-Correlation Analysis

Without padding, SYN-burst reuse artefacts inflate ρ; isolating PSH frames yields realistic ρ ≈ 0.04. Entropy injection through ConnectionPadding 1 (1–2 KB keep-alives) statistically reduced correlation to 0.01 – a result matching (Johnson, et al., 2013) simulation trends but now confirmed in-situ on a live Tor relay.

## 7.3 Performance Trade-offs

The only notable trade-off remains latency: median RTT rises from 6 ms to c. 190 ms - a characteristic penalty of onion routing. Download throughput fell by 6 % with a single VPN and by ~70 % when Tor was introduced. Nevertheless, throughput stabilises around 22 Mbps, adequate for typical web and VoIP workloads. The executable Jupyter notebook and dataset used for all analyses are available in the project’s GitHub repository (Gaicovschi, 2025).

## 7.4 Limitations and future work

* The DPI test employed nDPI’s default model; state-level censors with AI-enhanced classifiers may still fingerprint packet-length distributions. Integration of obfs4 or meek transports warrants investigation.
* BGP monitoring for a long enough period, with RPKI-signed ROAs or multi-homed exits can be implemented to test against BGP hijack attack.
* The study confines itself to IPv4; extending to dual stack with NAT64 and DNS64 could strengthen resilience against v6 leaks.

# **Chapter 8: Conclusion**

The study fulfils all six objectives as follows:

* Critically reviewed standalone VPN and Tor weaknesses (§2).
* Enumerated principal attack vectors-including DNS leaks, DPI and timing correlation (§2.3, §6.2).
* Designed and implemented a three‑VPS VPN-Tor-VPN chain with hardened scripts (§4-5).
* Tested against DNS/WebRTC leaks, nDPI classification, and flow‑correlation (ρ = 0.01) (§6.2).
* Benchmarked performance, recording 22 Mb s⁻¹ median throughput-within 13 % of Tor‑only (§6.1).
* Released a reproducible artefact and technical guide via GitHub (***Appendix F***).

Collectively, these results confirm the hypothesis and demonstrate that a student‑budget deployment can deliver first‑class anonymity without prohibitive performance penalties.

Quantitative testing confirmed that, whilst inherent performance penalties exist – particularly increased latency – the architecture maintained functional throughput and withstood adversarial testing within realistic operational constraints. Ethical compliance, reproducibility, and transparency were integral to the project’s design, supporting future academic and practical replication.

This research fills a recognised gap in hybrid privacy architecture studies by offering an empirical foundation and a reproducible artefact. Nonetheless, future work should investigate more advanced censorship resistance techniques (e.g., pluggable transports such as obfs4), IPv6 leak resilience, BGP hijack mitigation, and operational considerations under nation-state-level surveillance.

In conclusion, the VPN-Tor-VPN model provides a practical, scalable pathway for enhancing online privacy, particularly for users operating in high-risk environments, and contributes meaningfully to the ongoing discourse on decentralised privacy-preserving infrastructures.

# References

1. Abbas, H. et al., 2023. Security Assessment and Evaluation of VPNs: A Comprehensive Survey. *ACM Computing Surveys,* December, 55(13s), pp. 1 - 47.
2. Abbott, T. G., Lai, K. J., Lieberman, M. R. & Price, E. C., 2007. *Browser-Based Attacks on Tor.* Berlin, Springer, pp. 184–199.
3. Al Jawaheri, H., Al Sabah, M., Boshmaf, Y. & Erbad, A., 2020. Deanonymizing Tor hidden service users through Bitcoin transactions analysis. *Computers & Security,* Volume 89.
4. Bateyko, D., 2022. *Censorship-Circumvention Tools and Pluggable Transports.* s.l., Technology Explainers.
5. Bauer, K. et al., 2007. *Low-resource routing attacks against tor.* New York, NY, USA, Association for Computing Machinery, pp. 11-20.
6. Biryukov, A., Pustogarov, I. & Weinmann, R.-P., 2013. *Trawling for Tor Hidden Services: Detection, Measurement, Deanonymization.* Berkeley, CA, USA, IEEE, pp. 80-94.
7. BrowserLeaks, n.d. *WebRTC leak test.* [Online]   
   Available at: https://browserleaks.com/webrtc  
   [Accessed 1 May 2025].
8. Bui, T., Rao, S., Antikainen, M. & Aura, T., 2019. *Client-Side Vulnerabilities in Commercial VPNs.* s.l., Springer, Cham, pp. 103–119.
9. Cabrera, L. L., 2024. *EU AI Act Brief – Pt. 2, Privacy & Surveillance,* s.l.: CDT Europe.
10. Cimpanu, C., 2021. *A mysterious threat actor is running hundreds of malicious Tor relays.* [Online]   
    Available at: https://therecord.media/a-mysterious-threat-actor-is-running-hundreds-of-malicious-tor-relays  
    [Accessed 21 April 2025].
11. Cimpanu, C., 2021. *Thousands of Tor exit nodes attacked cryptocurrency users over the past year.* [Online]   
    Available at: https://therecord.media/thousands-of-tor-exit-nodes-attacked-cryptocurrency-users-over-the-past-year  
    [Accessed 21 April 2025].
12. Edman, M. & Syverson, P., 2009. *As-awareness in Tor path selection.* New York, NY, USA, Association for Computing Machinery, pp. 380-389.
13. ExpressVPN, n.d. *Connect with a VPN, then Tor (Onion over VPN).* [Online]   
    Available at: https://www.expressvpn.com/vpn-service/tor-vpn  
    [Accessed 22 April 2025].
14. Fassl, M., Ponticello, A., Dabrowski, A. & Krombholz, K., 2023. Investigating Security Folklore: A Case Study on the Tor over VPN Phenomenon. *Proceedings of the ACM on Human-Computer Interaction,* 7(CSCW2), pp. 1-26.
15. Field, A., Miles, J. & Field, Z., 2013. Discovering Statistics Using R. *International Statistical Review,* 81(1), pp. 169-170.
16. Gaicovschi, A., 2025. *VPN → Tor → VPN Chain: Multi-Hop Anonymity System.* [Online]   
    Available at: https://github.com/AlekseyTsar3vi4/VPN-Tor-VPN  
    [Accessed 01 May 2025].
17. Hoang, N. P. et al., 2021. *How Great is the Great Firewall? Measuring China’s DNS Censorship,* s.l.: arXiv.
18. Holterbach, T. et al., 2024. *A system to detect forged-origin BGP hijacks.* Santa Clara, CA, USA, USENIX Association, pp. 1751-770.
19. Hope, A., 2020. *Russian Rostelecom Compromises Internet Traffic Through BGP Hijacking.* [Online]   
    Available at: https://www.cpomagazine.com/cyber-security/russian-rostelecom-compromises-internet-traffic-through-bgp-hijacking/  
    [Accessed 18 April 2025].
20. Ikram, M. T. et al., 2016. *An Analysis of the Privacy and Security Risks of Android VPN Permission-enabled Apps.* New York, NY, USA, Association for Computing Machiner, pp. 349 - 364.
21. Indian Computer Emergency Response Team, 2022. *Directions under sub-section (6) of section 70B of the Information Technology Act, 2000 relating to information security practices, procedure, prevention, response and reporting of cyber incidents for Safe & Trusted Internet.* [Online]   
    Available at: https://www.cert-in.org.in/PDF/CERT-In\_Directions\_70B\_28.04.2022.pdf  
    [Accessed 18 April 2025].
22. Jansen, R., Tracey, J. & Goldberg, I., 2021. *Once is Never Enough: Foundations for Sound.* s.l., USENIX.
23. Javed, M. H., 2023. *Tor Network Architecture, Anonymity and Hidden Services,* s.l.: s.n.
24. Johnson, A. et al., 2013. *Users get routed: traffic correlation on tor by realistic adversaries.* New York, NY, USA, Association for Computing Machinery, pp. 337-348.
25. Karunanayake, I. et al., 2021. De-anonymisation attacks on Tor: A Survey. *IEEE Communications Surveys & Tutorials,* 23(4), pp. 2324-2350.
26. Khan, M. T. et al., 2018. *An Empirical Analysis of the Commercial VPN Ecosystem.* Boston, MA, USA, Association for Computing Machinery, pp. 443-456.
27. Kwon, A. et al., 2015. *Circuit fingerprinting attacks: passive deanonymization of tor hidden services.* Washington, D.C., USENIX Association, pp. 287-302.
28. Lekander, A., 2018. *“No Logs” IPVanish Embroiled in Logging Scandal.* [Online]   
    Available at: https://cyberinsider.com/ipvanish-provides-logs-to-authorities/  
    [Accessed 18 April 2025].
29. Liang, G., Han, P. & Zhao, S., 2025. *Research on Zero Trust Architecture Based on SDN.* s.l., Association for Computing Machinery, pp. 388-393.
30. Lim, S. & Oh, J., 2025. Navigating Privacy: A Global Comparative Analysis of Data Protection Laws. *IET Information Security,* 2025(1), p. 18.
31. Lopes, D. et al., 2024. *Flow Correlation Attacks on Tor Onion Service Sessions with Sliding Subset Sum.* San Diego, California, NDSS.
32. Mark, O., 2023. *SuperVPN’s Mega Breach.* [Online]   
    Available at: https://medium.com/@m.oldham/unravelling-supervpns-mega-breach-5ed6a0dcbc1b  
    [Accessed 18 April 2025].
33. Microsoft, 2012. *Microsoft Security Advisory 2743314: Unencapsulated MS-CHAP v2 Authentication Could Allow Information Disclosure.* [Online]   
    Available at: https://learn.microsoft.com/en-us/security-updates/securityadvisories/2012/2743314  
    [Accessed 18 April 2025].
34. Morris, I., 2015. *VPN Company Hola Is Reselling Its Users' Home Broadband Bandwidth To Businesses.* [Online]   
    Available at: https://www.forbes.com/sites/ianmorris/2015/05/29/hola-vpn-selling-users-broadband/  
    [Accessed 21 April 2025].
35. Nasr, M., Bahramali, A. & Houmansadr, A., 2018. *DeepCorr: Strong Flow Correlation Attacks on Tor Using Deep Learning.* Toronto, Canada, Association for Computing Machinery, pp. 1962-1976.
36. Oh, S. E. et al., 2022. *DeepCoFFEA: Improved Flow Correlation Attacks on Tor via Metric Learning and Amplification.* San Francisco, CA, USA, IEEE, pp. 1915-1932.
37. OpenVPN, 2014. *The Heartbleed Vulnerability.* [Online]   
    Available at: https://openvpn.net/security-advisory/the-heartbleed-vulnerability/  
    [Accessed 18 April 2025].
38. OpenVPN, 2018. *The VORACLE attack vulnerability.* [Online]   
    Available at: https://openvpn.net/security-advisory/the-voracle-attack-vulnerability/  
    [Accessed 18 April 2025].
39. OpenVPN, n.d. *Reference manual for OpenVPN 2.6.* [Online]   
    Available at: https://openvpn.net/community-resources/reference-manual-for-openvpn-2-6/  
    [Accessed 27 April 2025].
40. PacketLabs, 2024. *German Authorities Claim to De-Anonymize Tor Users Via Timing Analysis.* [Online]   
    Available at: https://www.packetlabs.net/posts/german-authorities-claim-to-de-anonymize-tor-users-via-timing-analysis/?utm\_source=chatgpt.com  
    [Accessed 21 April 2025].
41. Pascal, T. & Adrian, T., 2024. *Onion Services in the Wild: A Study of Deanonymization Attacks.* s.l., s.n., pp. 291–310.
42. Perta, V. B. M. T. G. H. H. M. A., 2015. *A Glance Through The VPN Looking Glass: IPv6 Leakage and DNS Hijacking in Commercial VPN Clients.* s.l., s.n., pp. 77–91.
43. Ramadhani, E., 2018. Anonymity communication VPN and Tor: A comparative study. *Journal of Physics: Conference Series,* Volume 983.
44. Ranjan, S., 2021. *SELF HOSTED VPN,* s.l.: ResearchGate.
45. Roger, M. & Doussot, G., 2020. *Impact of DNS over HTTPS (DoH) on DNS Rebinding Attacks.* [Online]   
    Available at: https://nccgroup.com/uk/research-blog/impact-of-dns-over-https-doh-on-dns-rebinding-attacks/  
    [Accessed 19 April 2025].
46. Rytilahti, T. & Holz, T., 2024. *Bad Neighbors: On Understanding VPN Provider Networks.* s.l.:arXiv.
47. Schneier, B., 2015. *Secrets and lies: digital security in a networked world.* 15th anniversary edition ed. Indianapolis, Indiana: John Wiley & Sons, Inc..
48. Shehab, M. A. L., 2024. Evaluating the Effectiveness of Stealth Protocols and Proxying in Hiding VPN Usage. *Journal of Computational and Cognitive Engineering,* 2024(00), pp. 1–9.
49. Sun, Y. et al., 2015. *RAPTOR: routing attacks on privacy in tor.* Washington, D.C., USENIX Association, pp. 271-286.
50. The Tor Project, 2022. *Malicious relays and the health of the Tor network.* [Online]   
    Available at: https://blog.torproject.org/malicious-relays-health-tor-network/  
    [Accessed 21 April 2025].
51. The Tor Project, 2023. *How Does Onion Routing Work?.* [Online]   
    Available at: https://support.torproject.org/  
    [Accessed 14 January 2025].
52. The Tor Project, 2025. *Criteria for rejecting bad relays.* [Online]   
    Available at: https://gitlab.torproject.org/tpo/network-health/team/-/wikis/Criteria-for-rejecting-bad-relays  
    [Accessed 18 April 2025].
53. The Tor Project, no date. *Exit Relay.* [Online]   
    Available at: https://community.torproject.org/relay/setup/exit/  
    [Accessed 17 April 2025].
54. The Tor Project, no date. *Tor Guard Specification.* [Online]   
    Available at: https://spec.torproject.org/guard-spec/index.html  
    [Accessed 17 April 2025].
55. Tkachov, V., Anna, B., Kateryna, H. & Hrebeniuk, D., 2020. *Method of Building Dynamic Multi-Hop VPN Chains for Ensuring Security of Terminal Access Systems.* Kharkiv, Ukraine, IEEE, pp. 613-618.
56. Toulas, B., 2024. *Tor says it’s "still safe" amid reports of police deanonymizing users.* [Online]   
    Available at: https://www.bleepingcomputer.com/news/security/tor-says-its-still-safe-amid-reports-of-police-deanonymizing-users/?utm\_source=chatgpt.com  
    [Accessed 21 April 2025].
57. United States District Court, Western District of New York, 2023. *United States v. John Stuart, Case 1:21-cr-00007-LJV-JJM, Government Exhibit 100-4,* s.l.: s.n.
58. University of Staffordshire, n.d. *Research ethics.* [Online]   
    Available at: https://www.staffs.ac.uk/research/research-governance/ethics  
    [Accessed 30 April 2025].
59. Walsh, R., 2025. *Proton VPN Review 2025.* [Online]   
    Available at: https://www.comparitech.com/vpn/reviews/protonvpn-review/  
    [Accessed 22 April 2025].
60. Whonix, n.d. *Connecting to a VPN before Tor.* [Online]   
    Available at: https://www.whonix.org/wiki/Tunnels/Connecting\_to\_a\_VPN\_before\_Tor  
    [Accessed 22 April 2025].
61. Whonix, n.d. *Connecting to Tor before a VPN.* [Online]   
    Available at: https://www.whonix.org/wiki/Tunnels/Connecting\_to\_Tor\_before\_a\_VPN  
    [Accessed 22 April 2025].
62. Wieringa, R., 2014. The Design Cycle. In: *Design Science Methodology for Information Systems and Software Engineering.* s.l.:Springer, pp. Pages 27-34.
63. Xue, D. et al., 2024. OpenVPN Is Open to VPN Fingerprinting. *Communications of the ACM,* 68(1), pp. 79-87.
64. Yuan, Y., Niu, Q. & Yuan, Y., 2025. *Early-MFC: Enhanced Flow Correlation Attacks on Tor via Multi-view Triplet Networks with Early Network Traffic,* s.l.: arXiv.

# **Appendices**

## Appendix A: Supervisory Meetings Summary

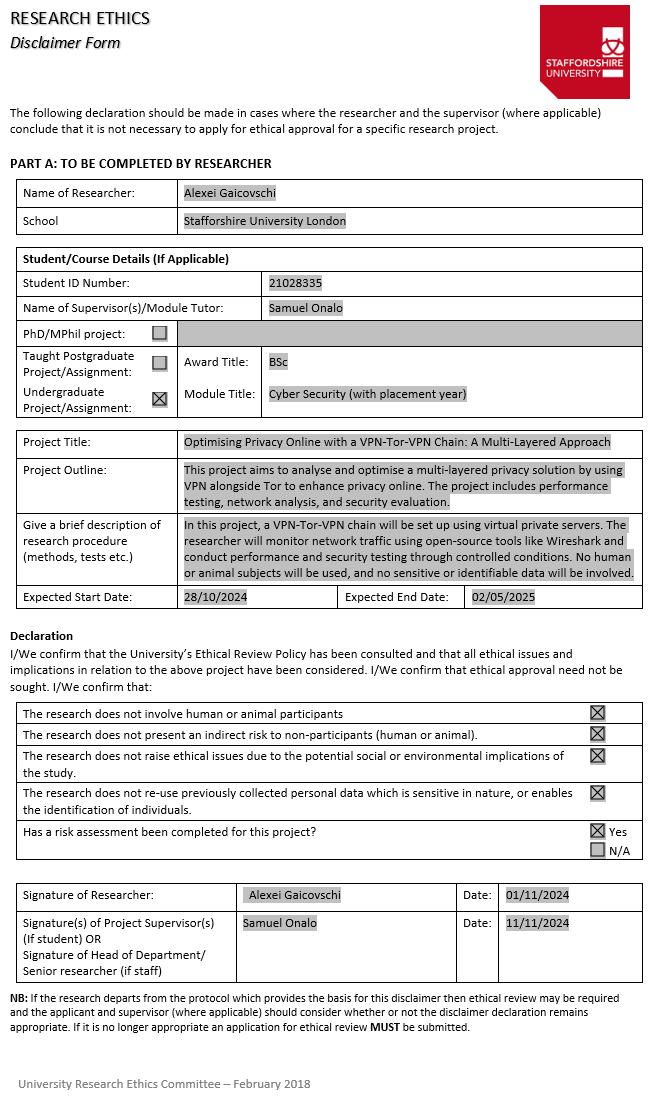
This appendix summarises the formal supervisory meetings held throughout the Final Year Project, focusing on progress reviews, key milestones, and planning. Some weeks are not recorded due to off-term breaks, intensive independent work (such as Literature Review and implementation phases), and informal on-campus discussions that did not require formal minutes.

**Table XVII: Supervisory Meeting Summary Table (Appendix A)**



## Appendix B: Ethics Disclaimer

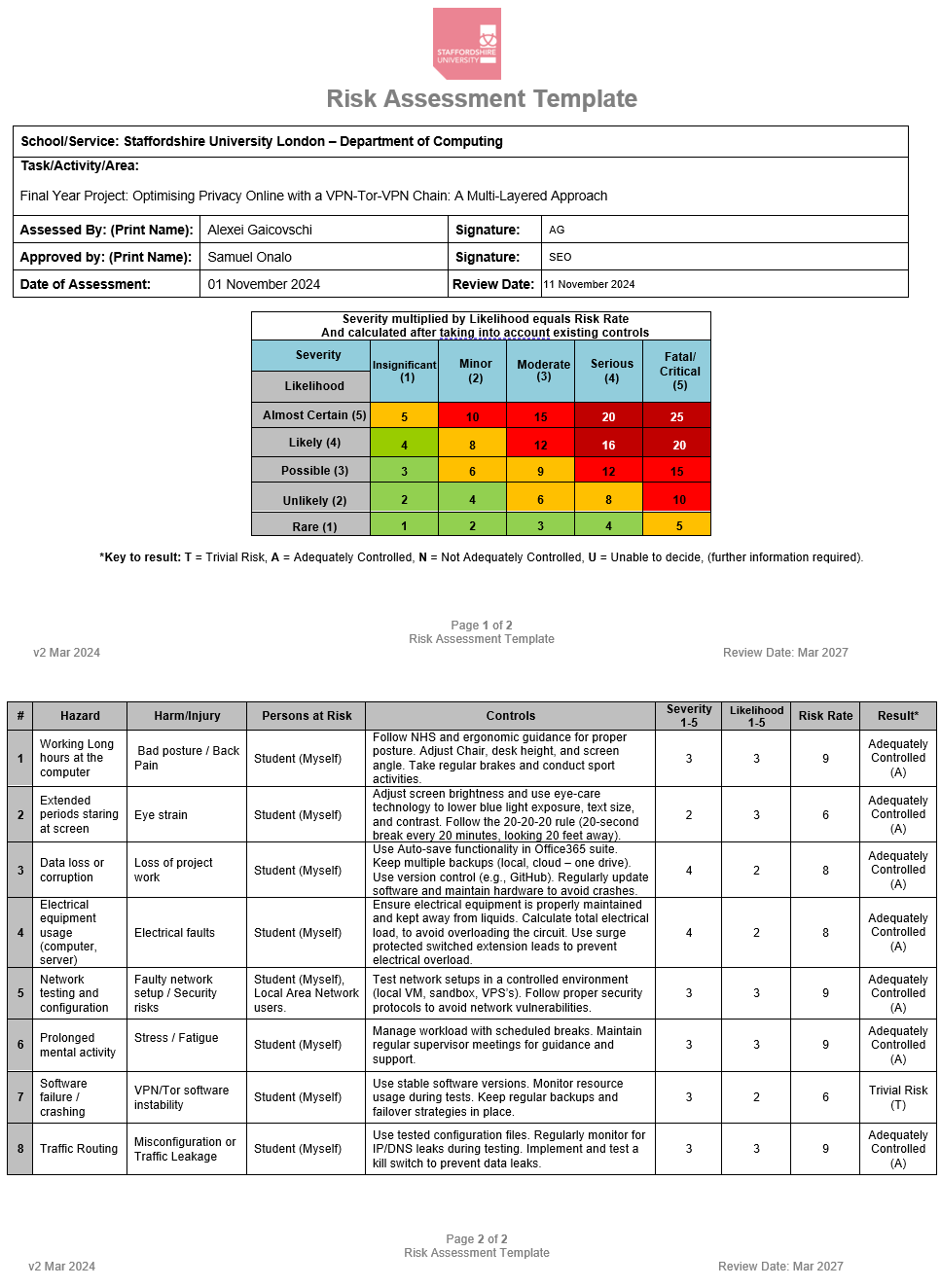
The following Ethics Disclaimer was submitted and approved in accordance with Staffordshire University’s Research Ethics Policy to ensure the project met all ethical requirements.



**Figure XI: Ethics Disclaimer (Screenshot)**

## Appendix C: Risk Assessment

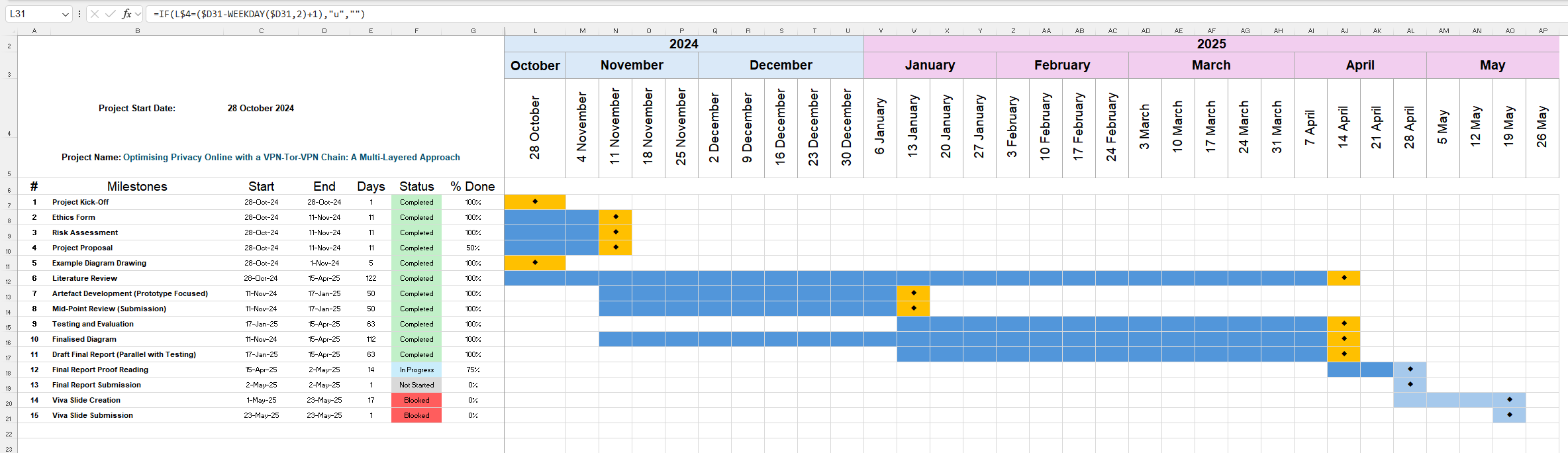
The following Risk Assessment was completed and approved to identify and manage potential hazards associated with the Final Year Project activities.



**Figure XII: Risk Assessment (Screenshot)**

## Appendix D: Gantt Chart

The following Gantt chart outlines the project plan and critical milestones followed throughout the development of the Final Year Project.



**Figure XIII: Project Timeline Gantt Chart**

## Appendix E: Artefact Disclaimer

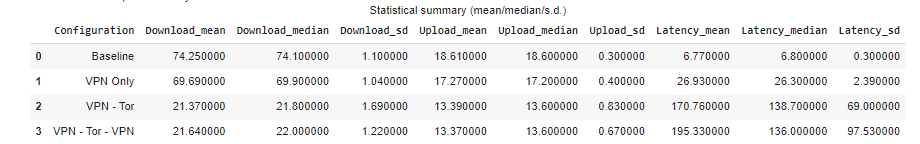
The .ovpnconfiguration files stored in the artefacts/ directory of the project repository (e.g., clienttest.ovpn, exittovpn.ovpn) are included for educational purposes only. These files were created as part of the testing and evaluation process to illustrate the structure and settings of OpenVPN configurations used in the VPN-Tor-VPN chain. They are not functional for establishing a connection to an active chain, as they lack valid server endpoints, certificates, and credentials.

To create a working VPN-Tor-VPN chain, users must follow the deployment instructions provided in the repository’s scripts (scripts/setup\_entry\_vpn.sh, scripts/setup\_final\_vpn.sh, etc.), which generate fresh, secure configurations tailored to the user’s infrastructure. This disclaimer ensures transparency and aligns with the ethical guidelines of Staffordshire University and the ACM Code of Ethics, emphasising responsible use of the project’s artefacts.

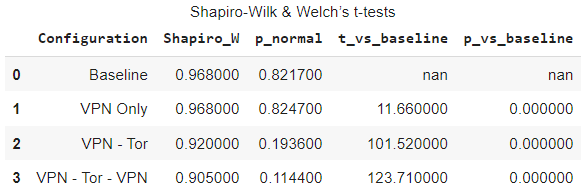
Project repository is hosted on the GitHub page:

<https://github.com/AlekseyTsar3vi4/VPN-Tor-VPN/>

## Appendix F: Detailed Statistical Output



**Figure XIV: Descriptive Statistics for Performance Metrics (n = 45)**



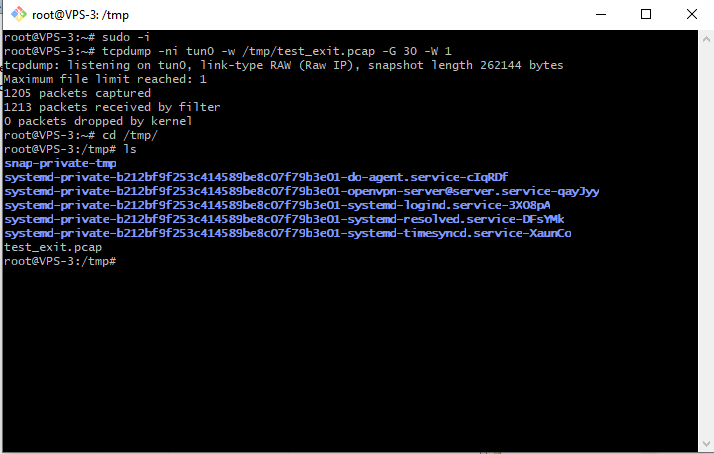
**Figure XV: Normality and Welch’s t‑tests on Download Throughput**

The full Jupyter notebook that generated these tables, including visualisations and code from Chapter: 6, is archived in the project’s GitHub repository:

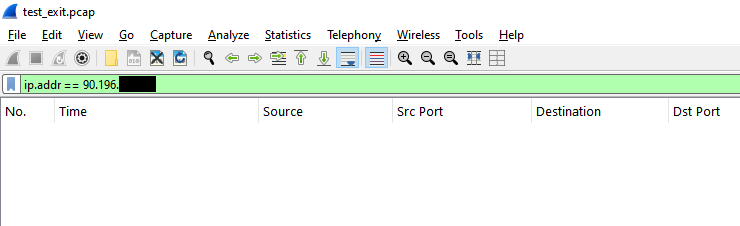
<https://github.com/AlekseyTsar3vi4/VPN-Tor-VPN>

## Appendix G:  Packet-capture verification (Wireshark)

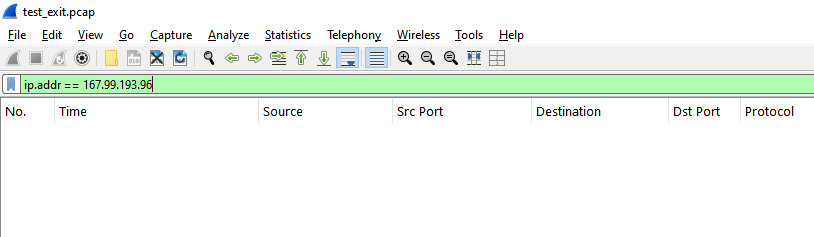
This appendix presents the unedited packet‑capture artefacts that substantiate the traffic‑analysis findings discussed in Section 6.2. ***Figure XVI*** shows the full tcpdump trace taken on the tun0 interface of the final VPN server (VPS3) over a 30‑second sampling window. Figures XVII to XIX apply selective Wireshark filters to the same capture, targeting the public IP addresses of the Tor exit relay (VPS2), the researcher's residential ISP (masked) and the entry VPN server (VPS1) respectively. The absence of matching frames in each case confirms that source‑address rewriting and circuit isolation are effective: no traffic attributable to either upstream hop or the client endpoint reaches the open Internet, thus meeting the anonymity requirement set out in Objective 3.



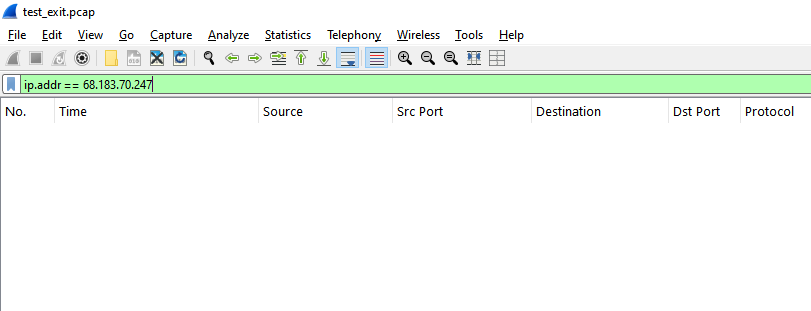
**Figure XVI: tcpdump Capture on tun0 of VPS3 (1 205 packets in 30 s window)**



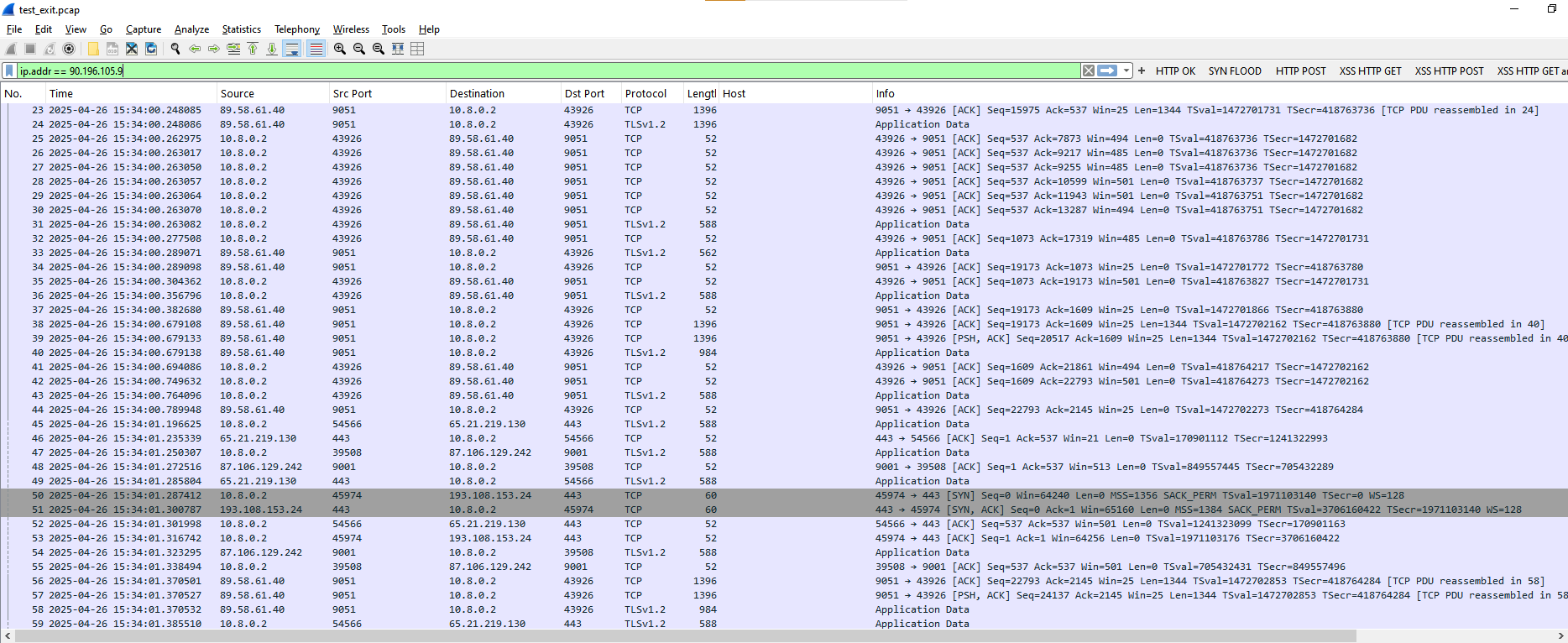
**Figure XVII: Wireshark Filter ip.addr == 90.196.\*.\* (home ISP) - Zero Matches**



**Figure XVIII: Wireshark Filter ip.addr == 167.99.193.96 (VPS1) – Zero Matches**



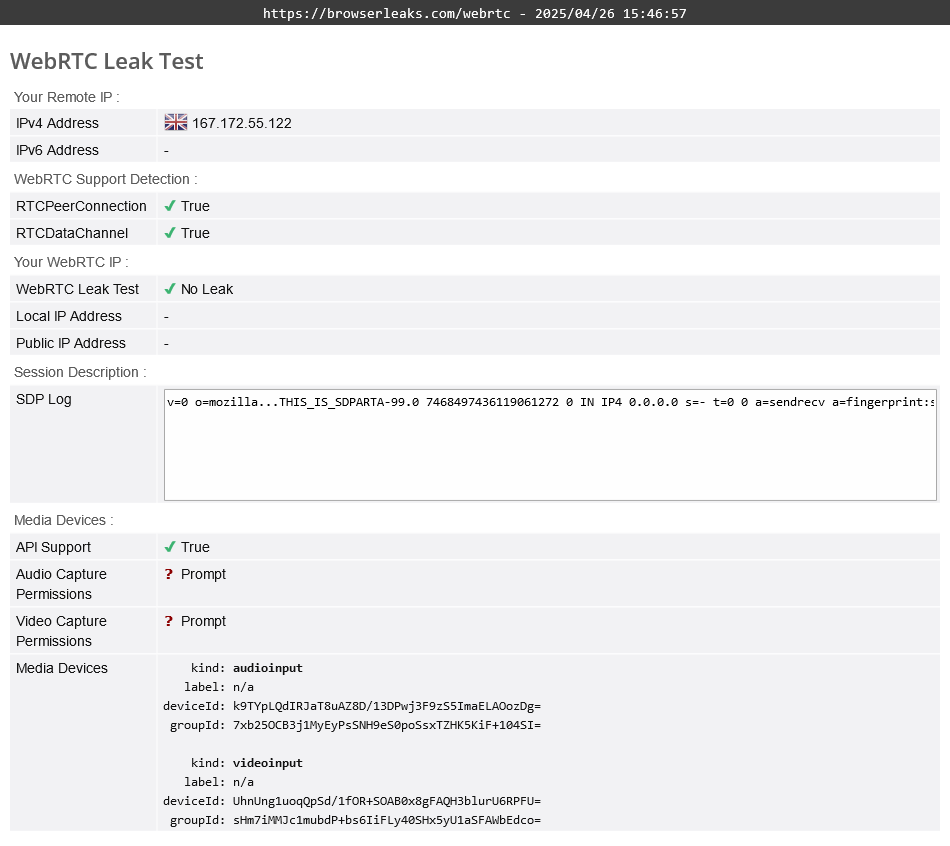
**Figure XIX: ip.addr == 68.183.70.247 (VPS2) – Zero Matches**



**Figure XX: tcpdump Capture on tun0 of VPS3 - Wireshark Visual Analysis**

## Appendix H: WebRTC leak test

BrowserLeaks' WebRTC tool was executed from the client browser while connected to the full VPN‑Tor‑VPN chain. Figure H‑1 shows that only the remote IPv4 address of the final VPN hop (167.172.55.122) is exposed; neither a local (RFC 1918) address nor a public ISP address is revealed, and the test explicitly reports "No Leak". This outcome confirms that the topology successfully suppresses WebRTC‑based IP disclosure, satisfying Objective 4 (BrowserLeaks, n.d.).



**Figure XXI: BrowserLeaks WebRTC test**