Detecting VPN Traffic through Encapsulated TCP Behavior

Michelina Hanlon
Stanford University

Gerry Wan
Stanford University

Anna Ascheman
Stanford University

Zakir Durumeric
Stanford University

ABSTRACT

Virtual Private Networks (VPNs) are increasingly being used to protect online users' privacy and security. However, there is an ongoing arms race between censors that aim to detect and block VPN usage, and VPN providers that aim to obfuscate their services from these censors. In this paper, we explore the feasibility of a simple, protocol-agnostic VPN detection technique based on identifying encapsulated TCP behaviors in UDP-based tunnels. We derive heuristics to distinguish TCP-over-UDP VPN traffic from plain UDP traffic using RFC-defined TCP behaviors. Our evaluations on real-world traffic show that this technique can achieve a false positive rate (FPR) of 0.11%, an order of magnitude lower than existing machine learning-based VPN detection methods. We suggest defenses to evade our detection technique and encourage VPN providers to proactively defend against such attacks.

KEYWORDS

VPN detection, censorship, traffic tunneling

1 INTRODUCTION

Virtual Private Networks (VPNs) are widely used to protect online privacy and bypass censorship [34]. Despite their popularity, the effectiveness of VPNs continues to be challenged by Internet Service Providers (ISPs), advertisers, and governments that seek to monitor, tamper with, or block network traffic [5, 11, 16, 19, 20, 26, 38, 43]. These entities are known to employ strategies such as IP tracking, VPN provider blocking, and deep packet inspection (DPI) to detect and block VPN usage [15, 27]. In response, users and providers alike have countered with techniques to prevent detection, including rotating IP addresses [8], using non-default ports [21], and obfuscating VPN traffic [22, 25, 30]. However, even commercial obfuscated VPN services that are touted as robust to detection have been shown to be straightforward for censors and ISPs to block using protocol fingerprinting [42].

While a large body of work has emerged on how to detect VPN traffic using machine learning (ML) [4, 10, 17, 18, 24, 36], in practice, VPN detection methods used by real-world censors still rely on simpler heuristics like protocol fingerprints, the fraction of set bits, and the number, fraction, and position of ASCII characters [40]. Building on this observation, we take an adversarial perspective and explore a simple yet understudied protocol-agnostic approach to VPN detection that exploits the leakage of tunneled (inner) protocol behaviors in the observed (outer) VPN connection. Specifically, we consider whether the RFC-defined characteristics of the TCP transport protocol inadvertently leak that a UDP connection is being used to tunnel traffic. Our approach is based on a fundamental feature of VPNs—the encapsulation of one protocol within another—and uses detection heuristics that are grounded in the intrinsic behaviors of TCP. This method avoids biases from empirical heuristics derived from particular VPN/non-VPN traffic datasets, which allows for a more generalizable method.

We find that several fundamental aspects of TCP behavior (e.g., the presence of TCP three-way handshakes and acknowledgement packets) can be detected via simple threshold-based heuristics derived from RFC specifications. We evaluate the effectiveness of our approach on real traffic from a large university campus network. Our results suggest that TCP-over-UDP VPN detection is feasible with a false positive rate (FPR) of 0.11%, which is an order of magnitude lower than those of proposed ML-based VPN detection techniques [4, 10, 18]. Our findings motivate that when building future VPN protocols, the security community needs to actively consider how to mask the protocol characteristics of inner transport-layer and application-layer protocols.

2 RELATED WORK

VPN Detection. Numerous VPN detection methods have been proposed in prior work, which range from using information databases to identify known VPN providers and proxies [13] to ML-based classification of flow statistics [4, 10, 17, 18, 24, 36]. While ML-based VPN detection research has grown in popularity, many of these ML-based classifiers are evaluated on synthetic datasets [6] in offline settings, and few have demonstrated practical usage in real-world environments [42]. Indeed, sophisticated censors like the Great Firewall (GFW) have been shown to apply "crude but efficient heuristics" to detect and block encrypted traffic, rather than relying on more complex and less interpretable ML-based strategies [40].

We build upon prior works that use simple and lightweight VPN detection methods. Webb et al. [39] proposed the use of network delay measurements to distinguish proxy and VPN traffic from direct traffic, which relies on the assumption that the round-trip time (RTT) of proxied connections are different from those of direct connections. Xue et al. [42] demonstrated the ability to fingerprint OpenVPN connections using byte patterns, packet sizes, and server responses, achieving negligible false positive rates. Xue et al. [41] further demonstrated a protocol-agnostic approach to VPN/proxy detection by fingerprinting encapsulated TLS handshakes. In our work, we expand on this idea of detecting encapsulated features, instead focusing on more general encapsulated TCP characteristics.

Free and Open Communications on the Internet 2024 (2), 77–82

Obfuscated VPNs. As ISPs and censors have improved at VPN detection, some VPN providers have begun to offer "obfuscated" VPN services that can supposedly bypass censorship [22, 25, 30]. However, many of these obfuscation techniques primarily defend against deep packet inspection (DPI) and are not designed to protect against attacks that exploit packet size and timing characteristics. Xue et al. [42] investigated 20 obfuscated VPN configurations deployed by 14 VPN providers, and found that most of these tunnels do not add random padding to packet payloads. As a result, many obfuscated VPN variants are susceptible to traffic analysis attacks.

3 TCP-OVER-UDP VPN DETECTION

Our VPN detection method identifies the presence of TCP traffic tunneled within UDP-based VPN connections. This approach considers that RFC-mandated TCP characteristics can leak into the outer VPN connection. In particular, TCP connections, even when wrapped inside an encrypted UDP tunnel, can exhibit distinct and easily measurable patterns in the size, direction, and timing of packets. By identifying UDP connections that exhibit strong TCP-like behavior, we can infer the presence of tunneled TCP traffic that is likely to be indicative of VPN usage. While this technique is limited to UDP-based VPNs tunneling TCP traffic, we note that many popular VPN services (e.g., ExpressVPN, NordVPN, Surfshark, and Private Internet Access) use UDP-based protocols because the outer tunnel does not need to provide any TCP features [1, 9, 12, 31].

The advantages of our approach are threefold: (1) it relies on characteristics of traffic tunneling, a fundamental feature of VPNs, and thus eliminates the need to fingerprint or account for specific VPN protocols or providers, (2) its use of intrinsic, RFC-mandated, traits of TCP avoids biases that may be present in particular datasets used to train ML-based VPN detection systems, and (3) many TCP behaviors are easily recognizable, even when packet payloads are encrypted. We choose three distinguishing characteristics of TCP that are required by RFC 9293 (Transmission Control Protocol) [33]:

- **3WHS:** The presence of a three-way SYN, SYN-ACK, ACK handshake to open the connection (RFC 9293, Section 3.5).
- **500msACK:** The presence of an ACK packet generated within 500 ms of the arrival of a data segment (RFC 9293, Section 3.8.6.3).
- **2RMSS:** The presence of an ACK packet generated after the receival of 2×RMSS bytes of data, where RMSS is the maximum segment size (MSS) specified by the TCP endpoint receiving the segments (RFC 9293, Section 3.8.6.3).

While RFC 9293 lists multiple TCP requirements that "must" or "should" be implemented, we select these characteristics for their ease of inference using only the size, direction, and timing of packets in a connection. For instance, the presence of exponential backoff, slow start, and congestion avoidance can also be strong indicators of TCP, but are less trivial to detect without access to TCP sequence numbers. Additionally, connection closing behaviors may be less practical to use since they only occur at the end of the connection.

3.1 Practical Heuristics

A passive observer of VPN traffic cannot access unencrypted TCP-header fields (e.g., whether a packet has the ACK flag set) within the VPN tunnel, so we derive threshold-based heuristics for each of our three TCP characteristics that take VPN encapsulation into account. We evaluate the efficacy of our derived heuristics in Section 4.

Encapsulated 3WHS. Since TCP flags are not visible inside encrypted tunnels, we propose a heuristic for the presence of a 3WHS based on the size, direction, and sequence of packets in the outer UDP connection. We consider an encapsulated 3WHS to be a sequence of three packets in alternating directions with non-increasing size, with the first packet having a UDP payload size of 60–160 bytes (indicating no TCP payload). This size is derived from lower and upper bounds on the size of IP headers, TCP headers, and VPN metadata [41], and confirmed through empirical analysis of anonymized, real-world TCP handshakes captured on a large university campus network. [1] Because some VPN protocols send initial connection establishment packets that are not part of the tunneled connection, we loosen the requirement of the presence of a 3WHS to occur within the first 20 packets of the outer UDP connection.

Encapsulated 500msACK. We approximate a TCP data segment as any UDP packet with size greater than the estimated size of tunneled packets that do not contain a TCP payload. We consider one instance of 500msACK as the observation of a packet sent in one direction within 500 ms plus the RTT after observing a packet containing a TCP payload sent in the opposite direction. RTT is estimated by the time elapsed during the 3WHS, or a default 500 ms if a handshake is not detected.

Encapsulated 2RMSS. Since passive observers are unable to view the MSS values specified in the inner TCP connection, we define a default RMSS based on the maximum transmission unit (MTU) that standard Ethernet supports. We consider one instance of 2RMSS as the observation of a packet sent in one direction within the estimated RTT after observing 2×RMSS bytes of TCP data sent in the opposite direction. TCP data length is computed as the difference between the size of the outer UDP packet and the size of the headers and VPN metadata. The size of the headers and VPN metadata is estimated by the size of the first 3WHS packet, or a default 160 bytes if a handshake is not detected.

3.2 VPN/Non-VPN Classifier

We use these heuristics to build a simple classifier that distinguishes between TCP-over-UDP VPN traffic and plain UDP traffic. To classify an observed UDP connection as VPN, we require (1) the presence of an encapsulated 3WHS, and (2) a rate of compliance with encapsulated 500msACK and 2RMSS of least t, where t is a tunable threshold. Rate of compliance is defined as the number of times an expected acknowledgement packet is observed divided by the number of times an acknowledgement packet is expected.

Similar to prior research on the practical detection of VPN and proxy traffic [41, 42], we only observe the first W packets of the connection (the observation window) before making a decision. As in recent work on detecting OpenVPN traffic [42], we evaluate our classifier at $W = 100$. However, note that we do not need to observe all 100 packets if the absence of an encapsulated 3WHS suggests

[1] From 19,468 successful real-world TCP handshakes captured on a large university campus network, we observe that 97% of handshakes do not contain data and have non-increasing packet sizes across the three packets.

that it is not a VPN connection prior to the full observation window. Our choice to use a threshold-based classifier is based on previous work demonstrating that even advanced censorship systems rely on simple heuristics [40].

4 EVALUATION

We evaluate our classifier and compare its performance to alternative VPN detection techniques. We demonstrate that the identification of encapsulated TCP behaviors in UDP traffic can be an effective technique for VPN detection.

4.1 Dataset

We collected a dataset of VPN and non-VPN traffic to evaluate our classifier. Unfortunately, we found that existing public VPN/non-VPN datasets were not suitable for our evaluation. Most notably, the popular ISCXVPN2016 dataset [6] contains discrepancies that indicate the network was tapped prior to packets going through the VPN client.[2] Therefore, these traces may not be representative of VPN traffic a real-world censor might observe.

VPN (TCP-over-UDP). We generate and collect VPN traffic across three different traffic categories: web browsing (Chrome), file transfer (SCP and SFTP), and remote access (SSH). These categories are informed by other VPN traffic datasets [6, 14]. We connect to a VPN server managed by ExpressVPN using the default OpenVPN (UDP mode) configuration. To generate file transfer traffic, we upload/download random text files to/from a remote server using either SCP or SFTP. To generate SSH traffic, we randomly choose commands to remote execute from a list of 13 shell commands (listed in Table 1 in Appendix A). To generate web browsing traffic, we use Puppeteer to send requests to the top 1,000 websites, as defined by Google CrUX [29] on Github [7], one website per packet trace. Per Ruth et al., this accounts for about 60% of website requests [28]. We note that the web browsing VPN traffic is multiplexed, as it could include multiple TCP streams as well as DNS traffic. This is representative of real-world web browsing traffic, and is useful in evaluating the robustness of our classifier to connection multiplexing. We collect 10,000 packet traces for each category of traffic, resulting in a total 30,000 VPN flows for classification. All packet traces are at least 85 packets in length.

Non-VPN (Plain UDP). For non-VPN traffic, we capture anonymized real-world UDP traffic from a large university campus network using Retina [35]. Some of the UDP traffic we capture may be VPN traffic, but we minimize this by filtering out flows on ports commonly associated with VPNs (e.g., 1194/1195 for OpenVPN). However, there still could be VPN connections within this UDP dataset, so our reported false positive rates are upper bounds, and true positive rates are lower bounds. We parse the collected UDP traffic into individual flows, defining a single UDP flow as a unique combination of source IP address, source port, destination IP address, and destination port. Additionally, we only consider flows that have 20 or more packets. In total, we collect 27,382 UDP flows.

[2]We downloaded a portion of the ISCXVPN2016 dataset and found that the packet traces were majority TCP traffic, despite the dataset website [23] stating that OpenVPN (UDP mode) was used to capture traffic. Jorgensen et al. [14] also came to the same conclusion that the network may have been tapped prior to the packets going through the VPN client.

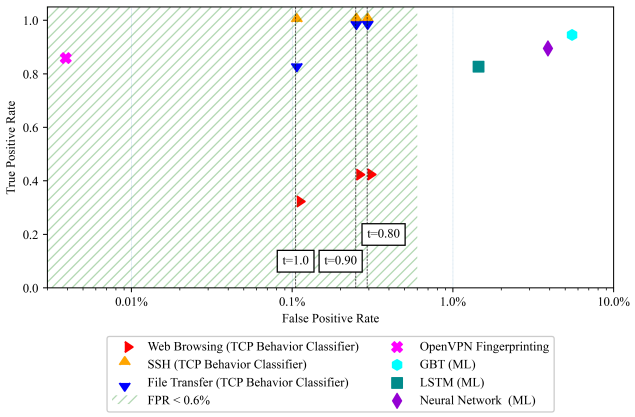

Figure 1: Classifier Results ($t = 1.00, 0.90, 0.80$; $W = 100$). The TPR against the FPR of our classifier (broken down by traffic category) and alternative VPN detection methods. Our protocol-agnostic classifier achieves a FPR an order of magnitude lower than ML detection techniques.

4.2 Results

Overall Classifier Results. We classify 27,382 UDP connections and 30,000 TCP-over-UDP VPN connections. Figure 1 presents the true positive rate (TPR) against the false positive rate (FPR), broken down by traffic category, under varying rate-of-compliance thresholds $t = 1.00, 0.90, 0.80$. We omit results for $t < 0.80$, as $t = 0.80$ achieves within 1% of the maximum TPR our classifier can achieve, given that it requires the detection of an encapsulated 3WHS to be classified as VPN, regardless of the value of t. The TPR is calculated as the number of true VPN flows classified as VPN divided by the total number of VPN flows; the FPR is calculated as the number of non-VPN flows classified as VPN divided by the total number of non-VPN flows.

Figure 1 also shows the TPR versus FPR for alternative VPN detection methods. We note that we do not use the same dataset for evaluation, and even among the prior work, different datasets are used. For a more meaningful comparison, we show the best results, defined by the lowest FPR, reported by the following works: gradient boosting trees (GBT) [4], bidirectional LSTM network with attention mechanism [10], flow statistic-based classification using a multi-layered perceptron neural network [18], and OpenVPN fingerprinting [42]. OpenVPN fingerprinting is a threshold-based classifier; all other classifiers are ML-based. We note that the FPRs of the ML-based classifiers (1.4–5.5%) are greater than the estimated FPR of the inferred algorithm used by the GFW to block fully encrypted proxies as of 2023 (0.6%) [40], indicating their limited practicality for a real-world censor. OpenVPN fingerprinting reports a lower FPR (0.0039%) but is limited to OpenVPN-based connections.

Our classifier achieves a FPR of 0.11–0.29%. This FPR is an order of magnitude lower than the FPRs of proposed ML-based techniques (1.4–5.5%) and is on par with the estimated FPR of the inferred algorithm used by the GFW (0.6%) [40]. The TPR, broken down by traffic category, is as follows: for SSH traffic, 99.43–99.56%; for file transfer traffic, 83.95–99.73%; and for web browsing traffic, 32.35–42.44%.

The significantly lower TPR for web browsing traffic is due to failure in detecting an encapsulated 3WHS in many of the web browsing VPN flows. On investigation of these traffic flows prior to being encapsulated by the VPN client, we found that many of the 3WHSs were interrupted by DNS queries, a symptom of multiplexing multiple connections within a single VPN tunnel. This demonstrates that connection multiplexing does impact the accuracy of our classifier, and therefore, could be an effective defense.

Feature Importance. We perceive the presence of a 3WHS based on our heuristic in only 0.33% of plain UDP flows, compared to 80.59% of VPN flows (and 99.65% of VPN flows not including multiplexed web browsing traffic). Notably, this feature on its own can distinguish between non-VPN and VPN traffic with a FPR of 0.33%. We measure plain UDP and VPN flow compliance with our 500msACK and 2RMSS heuristics per flow length (in packets). Our results are shown in Table 2 in Appendix A. Although the distinguishing power of both features increase with flow length, even at 500 packets, neither of these features on their own would be sufficient to effectively differentiate between TCP-over-UDP and UDP traffic. However, they are useful in negating false positives and including them in our classifier results in a 67% reduction in false positives for $t = 1.0$ (89 false positives to 29 false positives).

Robustness of 3WHS. Because we find the presence of an encapsulated 3WHS to be a strong distinguishing feature of encapsulated TCP traffic, we investigate its robustness if it is modified to account for two potential VPN detection defense tactics: random padding and connection multiplexing. To account for connection multiplexing, we modify the encapsulated 3WHS feature to allow for disruption of the handshake. The packets of the handshake are not required to be sequential; they are only required to occur within 500 ms. To account for random padding, we consider a simple padding scheme used by some proxy protocols (e.g., *vmess* [32]), adding between 0 and 63 bytes to each packet. We modify the encapsulated 3WHS feature to not require the three handshake packets to have non-increasing sizes, and increase the maximum expected size of a TCP packet without data by 100 bytes. Table 3 in Appendix A presents the results for our classifier using these modified encapsulated 3WHS features. As expected, both TPR and FPR increase with our modified heuristics, but we observe that only using random padding *or* multiplexing does not significantly increase the FPR (0.11% to 0.15% for random padding, and 0.11% to 0.53% for multiplexing). However, a combination of these defenses increases the FPR more significantly (0.11% to 2.57%).

5 DISCUSSION

Limitations and Future Work. Our work focuses on detecting UDP-based VPNs tunneling TCP traffic. We do not evaluate TCP-based VPNs or UDP-based VPNs tunneling other types of traffic. However, our results provide evidence that this *style* of attack—detecting the presence of tunneled connections based on protocol-agnostic, RFC-mandated behaviors—may be an effective method of VPN detection. Additionally, although we expect that our attack would generalize to other VPN protocols besides OpenVPN, we leave this to future work. Finally, we recognize that our dataset is not necessarily representative of the distribution of VPN/non-VPN traffic in the real world and is relatively small compared to the volume of traffic a censor must analyze. If VPN traffic accounts for a small percentage of the total traffic, even a low FPR of 0.11% (Section 4.2) could block more legitimate traffic than VPN traffic. We leave a full-scale evaluation of the practicality of our methodology for real-world censors to future work.

Suggested Defenses. We urge VPN providers that aim to bypass censorship to proactively defend against attacks that exploit encapsulated traffic behaviors. Our work confirms the findings of prior work [41] that while random padding slightly increases the difficulty of encapsulated-behavior attacks, multiplexing shows more promise to be a robust defense. In the short term, we suggest VPN providers at the very least add random padding and timing obfuscation to increase the difficulty of traffic metadata attacks. Drawing packet sizes from certain distributions, similar to the *obsf4* [2] proxy protocol, would likely offer a more robust defense against attacks that exploit packet size analysis. Connection multiplexing in tunnels can further increase the difficulty of these types of attacks, and is a natural form of timing obfuscation that does not artificially insert delays between packets. However, as Xue et al. [41] noted, multiplexing and random padding could introduce other detectable behavior (e.g., a larger than "normal" packet size distribution). As another potential defense to obscure distinctive encapsulated protocol characteristics, VPN providers could consider transporting packets from the same flow on different random ports to make flow analysis more difficult, a strategy used by the VPN tool GoHop [37].

Ethical Considerations. There is a risk that real-world adversaries could implement our attack. We note that prior work has already proposed the use of encapsulated TLS handshakes to detect proxy traffic, and alludes to the potential of encapsulated TCP handshakes for obfuscated VPN detection [41]. We emphasize that the ultimate goal of our work is to better understand the feasibility of such an attack, and more importantly, provide insight into how it could be defended against. To minimize risk to users, our campus traffic analysis was performed in coordination with our university's privacy and security office. Investigations into anonymized network traffic unrelated to user behavior has been determined by our campus IRB as being non-human subjects research. We immediately anonymized all IP addresses with format-preserving IP encryption [3], and zeroed all packet payloads. Our experiments only involved the packet sizes and timings within connections.

6 CONCLUSION

In this work, we showed the efficacy of identifying RFC-defined TCP behavior within UDP connections as a means of detecting VPN tunnels. Our simple, protocol-agnostic classifier achieved a FPR as low as 0.11% on production network traffic, which is an order of magnitude lower than those of proposed ML-based VPN detection techniques. We emphasize that the methodology of detecting the presence of encapsulated protocols based on RFC specifications could be adapted to detect other types of tunneled traffic and we encourage VPN providers to consider this type of attack when developing protocols to be robust against passive analysis.

ACKNOWLEDGMENTS

This work was supported by the National Science Foundation under contracts CNS-2124424 and CNS-2319080 as well as by a Sloan Research Fellowship. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of their employers or the sponsors. In addition, we thank the Stanford network and security teams for their assistance.

REFERENCES

[1] Aurelija Andriekutė. 2023. *TCP vs UDP: A comparison of the protocols and their differences.* NordVPN. https://nordvpn.com/blog/tcp-or-udp-which-is-better/.
[2] Yawning Angel. 2019. *obfs4.* https://gitlab.com/yawning/obfs4.
[3] Jean-Philippe Aumasson. 2018. *Rust ipcrypt.* https://github.com/stbuehler/rust-ipcrypt
[4] Sikha Bagui, Xingang Fang, Ezhil Kalaimannan, Subhash Bagui, and Joseph Sheehan. 2017. Comparison of machine-learning algorithms for classification of VPN network traffic flow using time-related features. *Journal of Cyber Security Technology* 1 (06 2017), 1–19. https://doi.org/10.1080/23742917.2017.1321891
[5] Marcel Dischinger, Alan Mislove, Andreas Haeberlen, and Krishna P. Gummadi. 2008. Detecting Bittorrent Blocking. In *Proceedings of the 8th ACM SIGCOMM Conference on Internet Measurement* (Vouliagmeni, Greece) *(IMC '08)*. Association for Computing Machinery, New York, NY, USA, 3–8. https://doi.org/10.1145/1452520.1452523
[6] Gerard Draper-Gil, Arash Habibi Lashkari, Mohammad Saiful Islam Mamun, and Ali A Ghorbani. 2016. Characterization of encrypted and vpn traffic using time-related features. In *Proceedings of the 2nd international conference on information systems security and privacy (ICISSP)*. 407–414.
[7] Zakir Durumeric. 2023. *Cached Chrome Top Million Websites.* https://github.com/zakird/crux-top-lists.
[8] Aurelija Einorytė. 2023. *What is a rotating IP address, and how does IP rotation work?* NordVPN. https://nordvpn.com/blog/what-is-ip-rotation/.
[9] ExpressVPN. 2024. *VPN protocols: OpenVPN (TCP vs. UDP).* https://www.expressvpn.com/what-is-vpn/protocols/openvpn.
[10] Peipei Fu, Chang Liu, Qingya Yang, Zhenzhen Li, Gaopeng Gou, Gang Xiong, and Zhen Li. 2020. NSA-Net: A NetFlow Sequence Attention Network for Virtual Private Network Traffic Detection. 430–444. https://doi.org/10.1007/978-3-030-62005-9_31
[11] Thiago Garrett, Ligia E. Setenareski, Leticia M. Peres, Luis C. E. Bona, and Elias P. Duarte. 2018. Monitoring Network Neutrality: A Survey on Traffic Differentiation Detection. *IEEE Communications Surveys & Tutorials* 20, 3 (2018), 2486–2517. https://doi.org/10.1109/COMST.2018.2812641
[12] Kristin Hassel. 2018. *TCP vs UDP: Protocols, Ports, and Practical Applications.* Private Internet Access. https://www.privateinternetaccess.com/blog/tcp-vs-udp-understanding-the-difference/.
[13] Hans Hoogstraaten. 2018. *Evaluating server-side internet proxy detection methods.* Master's thesis. Leiden University.
[14] Steven Jorgensen, John Holodnak, Jensen Dempsey, Karla de Souza, Ananditha Raghunath, Vernon Rivet, Noah DeMoes, Andrés Alejos, and Allan Wollaber. 2023. Extensible Machine Learning for Encrypted Network Traffic Application Labeling via Uncertainty Quantification. *IEEE Transactions on Artificial Intelligence* (2023), 1–15. https://doi.org/10.1109/TAI.2023.3244168
[15] Renuka Kumar, Apurva Virkud, Ram Sundara Raman, Atul Prakash, and Roya Ensafi. 2022. A Large-scale Investigation into Geodifferences in Mobile Apps. In *31st USENIX Security Symposium (USENIX Security 22)*. USENIX Association, Boston, MA, 1203–1220. https://www.usenix.org/conference/usenixsecurity22/presentation/kumar
[16] Fangfan Li, Arian Akhavan Niaki, David Choffnes, Phillipa Gill, and Alan Mislove. 2019. A Large-Scale Analysis of Deployed Traffic Differentiation Practices. In *Proceedings of the ACM Special Interest Group on Data Communication* (Beijing, China) *(SIGCOMM '19)*. Association for Computing Machinery, New York, NY, USA, 130–144. https://doi.org/10.1145/3341302.3342092
[17] Mohammad Lotfollahi, Ramin Shirali Hossein Zade, Mahdi Jafari Siavoshani, and Mohammdsadegh Saberian. 2017. Deep Packet: A Novel Approach For Encrypted Traffic Classification Using Deep Learning. *CoRR* abs/1709.02656 (2017). arXiv:1709.02656 http://arxiv.org/abs/1709.02656
[18] Shane Miller, Kevin Curran, and Tom Lunney. 2018. Multilayer Perceptron Neural Network for Detection of Encrypted VPN Network Traffic. In *2018 International Conference On Cyber Situational Awareness, Data Analytics And Assessment (Cyber SA)*. 1–8. https://doi.org/10.1109/CyberSA.2018.8551395
[19] Alan Mislove, Massimiliano Marcon, Krishna P Gummadi, Peter Druschel, and Bobby Bhattacharjee. 2007. The differentiation of viral marketing through email: The role of content and network structure. In *ACM SIGCOMM Conference on Internet Measurement*. ACM, 35–50.
[20] Arash Molavi Kakhki, Abbas Razaghpanah, Anke Li, Hyungjoon Koo, Rajesh Golani, David Choffnes, Phillipa Gill, and Alan Mislove. 2015. Identifying Traffic Differentiation in Mobile Networks. In *Proceedings of the 2015 Internet Measurement Conference* (Tokyo, Japan) *(IMC '15)*. Association for Computing Machinery, New York, NY, USA, 239–251. https://doi.org/10.1145/2815675.2815691
[21] Hunain Muhammad. 2023. *Why is my ISP blocking VPNs? Top reasons & fixes.* PureVPN. https://www.purevpn.com/blog/isp-blocking-vpn/.
[22] NordVPN. 2024. *What are obfuscated servers?* https://nordvpn.com/features/obfuscated-servers/.
[23] University of New Brunswick. 2024. *VPN-nonVPN dataset (ISCXVPN2016).* https://www.unb.ca/cic/datasets/vpn.html.
[24] Yi Pang, Shuyuan Jin, Shicong Li, Jilei Li, and Hao Ren. 2012. OpenVPN Traffic Identification Using Traffic Fingerprints and Statistical Characteristics. In *International Standard Conference of Trustworthy Computing and Services.* https://api.semanticscholar.org/CorpusID:49617625
[25] ProtonVPN. 2022. *Defeat censorship with Stealth, our new VPN protocol.* https://protonvpn.com/blog/stealth-vpn-protocol/.
[26] Ram Sundara Raman, Leonid Evdokimov, Eric Wurstrow, J. Alex Halderman, and Roya Ensafi. 2020. Investigating Large Scale HTTPS Interception in Kazakhstan. In *Proceedings of the ACM Internet Measurement Conference* (Virtual Event, USA) *(IMC '20)*. Association for Computing Machinery, New York, NY, USA, 125–132. https://doi.org/10.1145/3419394.3423665
[27] Reethika Ramesh, Ram Sundara Raman, Matthew Bernhard, Victor Ongkowijaya, Leonid Evdokimov, Anne Edmundson, Steven Sprecher, Muhammad Ikram, and Roya Ensafi. 2020. Decentralized control: A case study of russia. In *Network and Distributed Systems Security (NDSS) Symposium 2020.*
[28] Kimberly Ruth, Aurore Fass, Jonathan Azose, Mark Pearson, Emma Thomas, Caitlin Sadowski, and Zakir Durumeric. 2022. A world wide view of browsing the world wide web. In *ACM Internet Measurement Conference.*
[29] Kimberly Ruth, Deepak Kumar, Brandon Wang, Luke Valenta, and Zakir Durumeric. 2022. Toppling top lists: Evaluating the accuracy of popular website lists. In *Proceedings of the 22nd ACM Internet Measurement Conference.* 374–387.
[30] Surfshark. 2024. *How does VPN obfuscation work?* https://surfshark.com/features/obfuscated-servers.
[31] Surfshark. 2024. *What protocols can I use with Surfshark?* https://support.surfshark.com/hc/en-us/articles/360010324739-What-protocols-can-I-use-with-Surfshark.
[32] V2Fly. 2021. *VMess protocol.* https://www.v2fly.org/en_US/developer/protocols/vmess.html.
[33] Ed W. Eddy. 2021. *Transmission Control Protocol (TCP).* RFC 9293. RFC Editor. https://datatracker.ietf.org/doc/html/rfc9293
[34] Preeti Wadhwani. 2023. *Virtual Private Network (VPN) Market size.* Global Market Insights. https://www.gminsights.com/industry-analysis/virtual-private-network-vpn-market.
[35] Gerry Wan, Fengchen Gong, Tom Barbette, and Zakir Durumeric. 2022. Retina: analyzing 100GbE traffic on commodity hardware. In *ACM SIGCOMM Conference.*
[36] Wei Wang, Ming Zhu, Jinlin Wang, Xuewen Zeng, and Zhongzhen Yang. 2017. End-to-end encrypted traffic classification with one-dimensional convolution neural networks. In *2017 IEEE International Conference on Intelligence and Security Informatics (ISI)*. 43–48. https://doi.org/10.1109/ISI.2017.8004872
[37] Yuzhi Wang, Ping Ji, Borui Ye, Pengjun Wang, Rong Luo, and Huazhong Yang. 2014. GoHop: Personal VPN to defend from censorship. In *16th International Conference on Advanced Communication Technology.* IEEE, 27–33.
[38] Nicholas Weaver, Christian Kreibich, and Vern Paxson. 2011. Redirecting DNS for Ads and Profit. In *USENIX Workshop on Free and Open Communications on the Internet (FOCI 11).*
[39] Allen T Webb and AL Narasima Reddy. 2016. Finding proxy users at the service using anomaly detection. In *2016 IEEE Conference on Communications and Network Security (CNS)*. IEEE, 82–90.
[40] Mingshi Wu, Jackson Sippe, Danesh Sivakumar, Jack Burg, Peter Anderson, Xiaokang Wang, Kevin Bock, Amir Houmansadr, Dave Levin, and Eric Wustrow. 2023. How the Great Firewall of China Detects and Blocks Fully Encrypted Traffic. In *32nd USENIX Security Symposium (USENIX Security 23)*. USENIX Association, Anaheim, CA, 2653–2670. https://www.usenix.org/conference/usenixsecurity23/presentation/wu-mingshi
[41] Diwen Xue, Michalis Kallitsis, Amir Houmansadr, and Roya Ensafi. 2024. Fingerprinting Obfuscated Proxy Traffic with Encapsulated TLS Handshakes. In *33st USENIX Security Symposium (USENIX Security 24).*
[42] Diwen Xue, Reethika Ramesh, Arham Jain, Michalis Kallitsis, J. Alex Halderman, Jedidiah R. Crandall, and Roya Ensafi. 2022. OpenVPN is Open to VPN Fingerprinting. In *31st USENIX Security Symposium (USENIX Security 22)*. USENIX Association, Boston, MA, 483–500. https://www.usenix.org/conference/usenixsecurity22/presentation/xue-diwen
[43] Ying Zhang, Zhuoqing Morley Mao, and Ming Zhang. 2009. Detecting Traffic Differentiation in Backbone ISPs with NetPolice. In *Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement* (Chicago, Illinois, USA) *(IMC '09)*. Association for Computing Machinery, New York, NY, USA, 103–115. https://doi.org/10.1145/1644893.1644905

A APPENDIX

Table 1: Shell commands used to generate SSH traffic. We generate SSH VPN traffic by randomly choosing commands to run from the list below until our packet trace is sufficiently long (at least 85 packets).

ls
touch myfile.txt
cat myfile.txt
rm myfile.txt
mkdir mydir
cd mydir
cd ..
echo "this is a command"
clear
pwd
grep "hello" myfile.txt
echo "hello" > myfile.txt
date

Table 2: VPN/non-VPN compliance with encapsulated 500msACK and 2RMSS heuristics ($t = 0.99$). The percentages represent the percent of flows that complied with the heuristic at least 99% percent of the time. We observe that both features increase in distinguishability as flow length increases. However, neither of these features on their own would be sufficient to distinguish between TCP-over-UDP and UDP traffic.

		20 packets	100 packets	500 packets
UDP	500msACK	81.5%	77.4%	70.9%
	2RMSS	97.6%	72.3%	56.1%
VPN	500msACK	89.4%	88.3%	97.6%
	2RMSS	100%	89.4%	81.5%

Table 3: Robustness of 3WHS ($t = 1.00$; $W = 100$). TPRs are averaged across all VPN traffic categories. Modifying the 3WHS feature to be robust to multiplexing (MUX) *or* simple random padding results in a FPR that may be practical for a real-world censor. Modifying the 3WHS feature to be robust to *both* of these defenses combined results in a FPR likely too high to be practical.

Modification	FPR	TPR
N/A (original)	0.11%	71.84%
Account for MUX	0.53%	93.36%
Account for random padding	0.15%	72.42%
Account for random padding & MUX	2.57%	93.40%