OpenVPN is Open to VPN Fingerprinting

Diwen Xue* Reethika Ramesh* Arham Jain* Michalis Kallitsis†
J. Alex Halderman* Jedidiah R. Crandall‡ Roya Ensafi*
*University of Michigan †Merit Network, Inc.
‡ Arizona State University/Breakpointing Bad

Abstract

VPN adoption has seen steady growth over the past decade due to increased public awareness of privacy and surveillance threats. In response, certain governments are attempting to restrict VPN access by identifying connections using "dual use" DPI technology. To investigate the potential for VPN blocking, we develop mechanisms for accurately fingerprinting connections using OpenVPN, the most popular protocol for commercial VPN services. We identify three fingerprints based on protocol features such as byte pattern, packet size, and server response. Playing the role of an attacker who controls the network, we design a two-phase framework that performs passive fingerprinting and active probing in sequence. We evaluate our framework in partnership with a million-user ISP and find that we identify over 85% of OpenVPN flows with only negligible false positives, suggesting that OpenVPN-based services can be effectively blocked with little collateral damage. Although some commercial VPNs implement countermeasures to avoid detection, our framework successfully identified connections to 34 out of 41 "obfuscated" VPN configurations. We discuss the implications of the VPN fingerprintability for different threat models and propose short-term defenses. In the longer term, we urge commercial VPN providers to be more transparent about their obfuscation approaches and to adopt more principled detection countermeasures, such as those developed in censorship circumvention research.

1 Introduction

ISPs, advertisers, and national governments are increasingly disrupting, manipulating, and monitoring Internet traffic [16, 22, 27, 47, 69]. As a result, virtual private network (VPN) adoption has been growing rapidly, not only among activists and journalists with heightened threat models but also among average users, who employ VPNs for reasons ranging from protecting their privacy on untrusted networks to circumventing censorship. As a recent example, with the passage of Hong Kong's new national security law, popular VPN providers observed a 120-fold surge in downloads due to fears of escalating surveillance and censorship [62].

In response to the growing popularity of VPNs, numerous ISPs and governments are now seeking to track or block VPN traffic in order to maintain visibility and control over the traffic within their jurisdictions. Binxing Fang, the designer of the Great Firewall of China (GFW) said there is an "eternal war" between the Firewall and VPNs, and the country has ordered ISPs to report and block personal VPN usage [60,61]. More recently, Russia and India have proposed to block VPN services in their countries, both labeling VPNs a national cybersecurity threat [44, 59]. Commercial ISPs are also motivated to track VPN connections. For example, in early 2021, a large ISP in South Africa, Rain, Ltd., started throttling VPN connections by over 90 percent in order to enforce quality-of-service restrictions in their data plans [64].

ISPs and censors are known to employ a variety of simple anti-VPN techniques, such as tracking connections based on IP reputation, blocking VPN provider (provider from hereon) websites, and enacting laws or terms of service forbidding VPN usage [46,53,60]. Yet, these methods are not robust; motivated users find ways to access VPN services in spite of them. However, even less-powerful ISPs and censors now have access to technologies such as carrier-grade deep packet inspection (DPI) with which they can implement more sophisticated modes of detection based on protocol semantics [43, 48].

In this paper, we explore the implications of DPI for VPN detection and blocking by studying the fingerprintability of OpenVPN (the most popular protocol for commercial VPN services [6]) from the perspective of an adversarial ISP. We seek to answer two research questions: *(1) can ISPs and governments identify traffic flows as OpenVPN connections in real time*? and *(2) can they do so at-scale without incurring significant collateral damage from false positives?* Answering these questions requires more than just identifying fingerprinting vulnerabilities; although challenging, we need to demonstrate practical exploits under the constraints of how ISPs and nation-state censors operate in the real world.

We build a detection framework that is inspired by the architecture of the Great Firewall [1, 11, 71], consisting of *Filter* and *Prober* components. A *Filter* performs passive filtering over passing network traffic in real time, exploiting protocol quirks we identified in OpenVPN's handshake stage. After a flow is flagged by a *Filter*, the destination address is passed

Figure 1: OpenVPN Session Establishment (TLS mode).

to a Prober that performs active probing as confirmation. By sending probes carefully designed to elicit protocol-specific behaviors, the Prober is able to identify an OpenVPN server using side channels even if the server enables OpenVPN's optional defense against active probing. Our two-phase framework is capable of processing ISP-scale traffic at line-speed with an extremely low false positive rate.

In addition to core or "vanilla" OpenVPN, we also include commercial "obfuscated" VPN services in this study. In response to increasing interference from ISPs and censors, obfuscated VPN services have started to gain traction, especially from users in countries with heavy censorship or laws against the personal usage of VPNs. Obfuscated VPN services, whose operators often tout them as "invisible" and "unblockable" [5, 49, 54], typically use OpenVPN with an additional obfuscation layer to avoid detection [2, 66].

Partnering with Merit (a mid-size regional ISP that serves a population of 1 million users), we deploy our framework at a monitor server that observes 20 Gbps of ingress and egress traffic mirrored from a major Merit point-of-presence. (Refer to § 5 for ethical considerations.) We use PF_RING [38] in zero-copy mode for fast packet processing by parallelized Filters. In our tests, we are able to identify 1718 out of 2000 flows originating from a control client machine residing within the network, corresponding to 39 out of 40 unique "vanilla" OpenVPN configurations.

More strikingly, we also successfully identify over two-thirds of obfuscated OpenVPN flows. Eight out of the top 10 providers offer obfuscated services, yet all of them are flagged by our Filter. Despite providers' lofty unobservability claims (such as: "[...] even your Internet provider can't tell that you're using a VPN" [49]), we find most implementations of obfuscated services resemble OpenVPN masked with the simple XOR-Patch [36], which is easily fingerprintable. Lack of random padding at the obfuscation layer and co-location with vanilla OpenVPN servers also make the obfuscated services more vulnerable to detection.

In a typical day, our single-server setup analyzes 15 TB of traffic and 2 billion flows. Over an eight-day evaluation, our framework flagged 3,638 flows as OpenVPN connections. Among these, we are able to find evidence that supports our detection results for 3,245 flows, suggesting an upper-bound false-positive rate three orders of magnitude lower than previous ML-based approaches [3, 14, 26].

We conclude that tracking and blocking the use of OpenVPN, even with most current obfuscation methods, is straightforward and within the reach of any ISP or network operator, as well as nation-state adversaries. Unlike circumvention tools such as Tor or Refraction Networking [8, 74], which employ sophisticated strategies to avoid detection, robust obfuscation techniques have been conspicuously absent from OpenVPN and the broader VPN ecosystem. For average users, this means that they may face blocking or throttling from ISPs, but for high-profile, sensitive users, this fingerprintability may lead to follow-up attacks that aim to compromise the security of OpenVPN tunnels [40, 51]. We warn users with heightened threat models not to expect that their VPN usage will be unobservable, even when connected to obfuscated services. While we propose several short-term defenses for the fingerprinting exploits described in this paper, we fear that, in the long term, a cat-and-mouse game similar to the one between the Great Firewall and Tor is imminent in the VPN ecosystem as well. We implore VPN developers and providers to develop, standardize, and adopt robust, well-validated obfuscation strategies and to adapt them as the threats posed by adversaries continue to evolve.

2 Background & Related Work

VPN tools create private networks across the public Internet through encrypted tunneling. Although many VPN protocols are being used, such as IPSec and WireGuard, OpenVPN remains the most supported and trusted protocol among commercial VPN providers [6]. Due to its versatility and open-source nature, OpenVPN has been used as the underlying protocol in numerous VPN products, which often advertise the protocol for its proven security [66]. In addition, OpenVPN's popularity continues to rise with the trend of users choosing to self-host open-source VPN tools [65].

OpenVPN Protocol. OpenVPN was first released in 2002 with the aim of creating a tunneling protocol focusing on security, while also being free and fast over the standard TCP and UDP [34]. When the OpenVPN tunnel is active, raw IP packets being sent to or from the tunnel to the final destination are encapsulated inside OpenVPN packets. To achieve secure communication, OpenVPN leverages the OpenSSL library as its cryptographic layer. Two methods for authentication and key exchange are provided to establish trust with peers: either pre-shared static key(s) or TLS-based negotiations. The latter has been adopted by the majority of commercial VPN

services. Two separate channels are used for key exchange and data transfer, both sharing a single multiplexed TCP/UDP stream. In the control channel, the client and server engage in a TLS-style exchange of key materials. As TLS is designed to operate over a reliable transport, OpenVPN provides its control channel with a sequential, reliable layer based on an explicit acknowledgement and re-transmission mechanism. The negotiated key from the control channel will be used to encrypt packets transferred in the data channel, which does not provide any reliability guarantee. Figure 1 presents a typical initialization sequence of OpenVPN packets leading to a fully encrypted data channel.

Tor, Proxy, and VPN Detection. The ongoing arms-race between the GFW and Tor has been extensively studied and is most representative of the con ict between censorship & surveillance and circumvention tools [9, 11, 12, 55, 56, 71]. Censors started by blocking Tor's website and public relays, which Tor responded to by deploying website mirrors and private, unpublished bridges. Next, censors moved to blocking with DPI by ngerprinting Tor's TLS handshake, e.g. cipher suites. Tor used Pluggable Transports (PT) obfuscators, such as Obfsproxy and meek [39], to mask the handshake. In response, censors deployed active probing to complement DPI-based ngerprinting to detect Tor and certain obfuscators.

There is limited previous work focusing on VPN traf c detection. Hoogstraaten [19] explored server-side VPN detection methods, ranging from using existing information databases (e.g. WHOIS, rDNS) to ngerprinting TCP options (e.g. advertised MSS). Webb et al. [70] proposed detecting proxies and VPNs based on traf c timing and latency. Their approach relied on the hypothesis that when a service is accessed through a proxy, the RTT measurement will be different from the RTT of a direct connection. Another class of previous work uses computational and machine learning models to passively detect VPN traf c [3, 14, 15, 17, 24, 26, 68], leveraging ow-level statistics such as connection duration and packet interval. Most of this work uses the same synthetic ISCXVPN2016 dataset [17]—which contains a balanced mixture of VPN and non-VPN traf c—to train and test a variety of machine learning and neural network classi ers in an of- ine, lab-setting. In contrast, our work primarily focuses on whether ISP-level adversaries can identify OpenVPN ows in near real time, and whether they can do so at scale, under practical constraints, and with minimal collateral damage. For this reason, we omit a full analysis of ML-based work, and only compare them with our approach in terms of false positives (falsely blocking legitimate traf c).

Obfuscated (Open)VPN. Various traf c obfuscation techniques have been examined in previous work. Wang et al. examined the detectability of Obfsproxy, FTE, and meek [67]. Using attacks based on protocol semantics, packet entropy, and timing-related features, they concluded that a determined censor could detect all three obfuscators reliably. Houmansadr et al. demonstrated that popular mimicry-based obfuscation tools failed to achieve unobservability because seamlessly simulating another protocol is extremely challenging [20]. Previous studies have suggested censors can use active probing to detect proxies that obfuscate traf c [1, 11, 71]. In response, "probe-resistant" proxies were developed, which remain silent when being probed by an unauthenticated adversary. However, researchers have demonstrated that carefully designed probes could still identify these proxies [13].

There is a marked demand for an emerging class of services called "stealth" or "obfuscated" VPN, especially from users in countries with heavy censorship or laws against personal VPN usage [60, 63]. Most obfuscated VPN services use OpenVPN as the underlying protocol for security and routing, with an obfuscation layer overlaid to avoid detection [2, 66][1]. OpenVPN's core developers prefer that obfuscation remains a separate project operating alongside the vanilla/core protocol, as they "do not want to play the cat-and-mouse game [as Tor]" [35]. The absence of a standardized obfuscation solution has led to a plethora of obfuscators implemented by different VPN providers, who often claim that their obfuscated services can remain undetected by ISPs and censors alike. For example, TorGuard introduces their obfuscated VPN service as "Engineered from the ground up to be impossible to detect" [54]. BolehVPN claims that their VPN obfuscation "...keeps you out of trouble, even in China" [5]. Common obfuscation strategies adopted by commercial VPNs include employing XOR-based scramblers, wrapping OpenVPN inside encrypted tunnels, or using proprietary protocols.

OpenVPN XOR Patch: Originally developed by Clayface as a patch for vanilla OpenVPN, the XOR patch scrambles a packet by either xor-ing bytes with a pre-shared key, reversing the order of the bytes, xor-ing each byte with its position, or a combination of these steps [36]. Notably, OpenVPN developers discourage its use due to the lack of code audit [57].

OpenVPN over Encrypted Tunnels: Some VPN services wrap OpenVPN traf c inside encrypted tunnels to prevent DPI ngerprinting. Some of the adopted obfuscation tunnels are Obfsproxy (obfs{2/3/4}), Stunnel, Websocket Tunnel, and encrypted proxies (shadowsocks, V2Ray).

Proprietary Protocols: A few VPN providers have developed proprietary obfuscated protocols, some of which are built on top of OpenVPN with a proprietary obfuscation layer added, such as VyprVPN or Astrill [2, 66].

To the best of our knowledge, we are the rst to explore the ngerprintability of commercial and/or obfuscated OpenVPN services on real traf c. Our unique study highlights the practicality of such ngerprinting, which has profound real-world security implications on end-users expecting certain privacy and anonymity guarantees from using these services.

[1] There are discussions on obfuscating WireGuard [72, 73], but to the best of our knowledge, they have yet to be deployed by any commercial VPNs

Figure 2: Framework Deployment on Merit Steps: (1) Client connects to VPN servers. (2) VPN connections, along with passing traf c, are being mirrored to the Filter. (3) Filter forwards server IP of suspected connections to the probing system. (4) Targets are sent to each dedicated Probers (5) Proberssend probes asynchronously. (6) Connections con rmed by probing are logged.

3 Challenges in Real-world VPN Detection

Effective investigation of ngerprintability requires incorporating perspectives of how ISPs and censors operate in practice. It is not enough to simply identify ngerprinting vulnerabilities, we need to demonstrate realistic exploits to illustrate the practicality of exploiting the vulnerability, while taking into consideration the ISP and censors' capabilities and constraints [56]. For instance, previous academic works considered using ow-level features to train ML classi ers for VPN detection [3, 14, 17, 24, 26, 68]. Yet, it remains unclear how practical these detection approaches are for ISPs and censors, and we know of no rigorous studies that examine real-world deployment of an ML-based censorship system [56]. Furthermore, previous works test on the ISCXVPN2016 dataset [17] with balanced OpenVPN and non-VPN traf c. However, we note that due to the low base rate of VPN traf c in the wild, even the best-performing ML system has false positive rates that can be economically impractical for real-world censors sensitive to collateral damage [67].

However, investigations adopting the viewpoint of ISPs and censors can be challenging. First, such investigation requires collaboration with real-world ISPs and access to their network traf c. We need to install monitors inside an ISP's network, while ensuring our analysis will not affect ISP's normal routing operations. Furthermore, analyzing traf c from real users raises ethical concerns. Processing raw network data may violate the privacy of users, in particular VPN users who often have a heightened threat model. Finally, deploying a system that performs ad-hoc traf c analysis in real time poses signi cant engineering challenges. We need to ensure the entire analysis framework (including processing and logging) keeps pace with the packet arrival rate and take into consideration the effect of potential asymmetric routing or packet loss on the analysis and results.

4 Adversary Model and Deployment

We assume a realistic censor (ISP) capability model based on knowledge from previous measurement studies on the arms race between censors and circumventors [1, 11, 56, 71]. We outline a censor-controlled on-path lter that passively observes and examines passing network traf c. The lter is stateful, but has limited resources and can maintain a limited amount of per-connection states for a short time. The lter is also constrained by long-term data storage and computational resources. In addition to lters installed inside the monitored networks, we assume the censor also operates measurement machines that can send protocol-speci c probes to further con rm the detection result. Such two-phase systems have already been adopted by real-world censors such as the GFW against Tor and Shadowsocks [1, 71]. Finally, we expect the censor is familiar with the protocol of interest and has access to the different obfuscators deployed by VPN providers (e.g., as a paid customer). We emphasize that this threat model corresponds to censor's capabilities as observed in practice today, rather than future capabilities.

To investigate the ngerprintability of OpenVPN and existing obfuscated solutions, we set up a two-phase detection framework in order to answer our key questions: 1) whether real-world censors are capable of performing such detection, and 2) whether it is economical to do this at scale Figure 2 shows an overview of our framework deployment. Partnering with Merit, we instantiate a Filter on a Monitoring Station overseeing mirrored traf c from a router that handles 20% of the ISP's traf c. The Filter performs passive ngerprinting over raw packets, exploiting traf c features unique to OpenVPN. IP and port information of ows agged by the Filter are forwarded to a probing system and then distributed to dedicated Probers. The Proberssend a set of pre-de ned probes speci cally designed to ngerprint an OpenVPN server. Finally, probed servers that are con rmed as OpenVPN are logged for manual analysis. Such a two-phase framework resembles how

real-world censors operate: lightweight filtering followed up by more expensive, but also more accurate, active probing. This framework is capable of processing massive traffic in real-time while also preventing excessive collateral damage.

5 Ethics, Privacy, and Responsible Disclosure

Raw network traffic that contains real users' data is highly sensitive, and this is especially true for traffic related to privacy-oriented services such as VPNs. Here we describe how we consider the security and privacy risks and ethical issues raised by our work, and we detail the procedural and technical steps we take to mitigate the risks.

Foremost among the ethical concerns associated with this work is our Filter deployment inside Merit's network to analyze user traffic. Merit, which has extensive previous experience collaborating with universities and has well-defined ethics and privacy rules to govern such projects, supervised the deployment. We also cleared our research plan with our university legal counsel and IRB. Although the IRB determined that the work is not regulated, we take extensive measures to minimize potential risks for end-users.

Our framework is fine-tuned on both real and lab-generated traffic data, and it is evaluated on live ISP traffic. For controlled fine-tuning, a small traffic snapshot (the ISP Dataset in section 7) was used to calibrate parameters, e.g., the size of observation window. The traffic snapshot, sampling 1/30 of all flows for 45 minutes on July 28, 2021, was generated and analyzed entirely on Merit systems, with security mechanisms limiting access to select members of the team. As with the design described in Section 5, Filter analyzed only the first payload byte, completely ignoring the remainder of the payload, and it recorded only the observed degree of variation. The raw snapshot was never inspected by humans and was deleted after the fine-tuning concluded.

For deployment and evaluation on live ISP traffic, the Filter architecture is designed to minimize risks of disrupting or modifying user traffic. The Monitoring Station only receives a copy of the traffic, so even if our software were to malfunction, network service would be unaffected. In addition, to reduce privacy risks, the Filter collects only the minimum information necessary for the subsequent probing operation. It records only the server IP addresses and ports of matching connections, which are bucketed into 5-minute internals to inhibit time correlation. These logs are stored and analyzed on a server that is securely maintained by Merit and is accessible only to a few members of our research team on a least-privilege basis. Merit reviewed our source code prior to deploying it on their network. During deployment and evaluation, no packet payloads or client IP addresses are ever recorded to disk or inspected by humans.

Based on the Filter log, the Probers send probes to candidate VPN servers. To minimize the risk of disrupting server operations, we design the probes to be non-invasive and make

Figure 3: OpenVPN Header in TCP and UDP modes (TLS only)

information available to assist operators in debugging any problems we inadvertently cause. Each server receives only 2–10 innocuous connection attempts, similar to those commonly used in Internet measurement tools like Nmap. The probes originate from two dedicated machines that we provisioned with web pages that explain the nature of the experiment and provide our contact information. We did not receive any inquiries, complaints, or problem reports. Since the server IP addresses themselves may sometimes be non-public, we only report aggregate statistics (e.g., the false positive rate) and will not publish any of the addresses that we collect. Any data requests will be referred to Merit.

As with all attack-oriented research, there is a risk that our work developing VPN fingerprinting techniques will be adopted by real attackers. To minimize this risk, we are in the process of responsibly disclosing our findings to the VPN operators whose obfuscated servers we successfully identified in our evaluation. We believe that the security of the VPN ecosystem is best advanced by having these problems surfaced by responsible researchers. Our work will help accurately inform users about the VPN services they rely on, and we hope it will enable more robust countermeasures to be developed and deployed.

6 Identifying Fingerprintable Features

In this section, we identify three features that fingerprint OpenVPN, exploiting byte pattern, packet length, and server behaviors, respectively.

6.1 Opcode-based Fingerprinting

As shown in Figure 3, each OpenVPN packet has a header of 24 bits in TCP mode or 8 bits in UDP mode, which is not part of the encrypted payload. Each OpenVPN header starts with an opcode that specifies the message type of the current packet and a key ID that refers to a (new) TLS session. The opcode field can take over 10 defined values, corresponding to message types transmitted during different communication stages. A typical OpenVPN session starts with the client sending a Client_Reset packet. The server then responds with a Server_Reset packet, and a TLS handshake follows. OpenVPN packets that carry TLS ciphertexts have P_Control as their message type. Since OpenVPN can run over UDP but has to provide a reliable channel for TLS, each P_Control packet is explicitly acknowledged by P_ACK packets. Finally,