VPN Leak Tester Tool

Nathan Leakeas, Matt Hiatt

GitHub Link — Video Demo

Abstract

VPN usage has become extremely prevalent in the United States and around the world. In 2025, 46% of Americans are using VPNs for one reason for another [1] and that portion has grown every year for the last several years. These VPNs come in a variety of forms, there are many free and open source versions available but also a booming industry for commercial VPN applications. Free and paid services alike have been the subject of reporting and academic investigation that has revealed ethics scandals, technical vulnerabilities, or other malfeasance that may cause the platforms not to live up their stated security and privacy guarantees [2]. We seek to provide an overview of the types of security concerns VPN users should be aware of and to develop a lightweight testing tool to help them ensure that their VPN is performing as they expect. Our tool reveals a common form of VPN vulnerability, DNS leaks. These leaks can exist in low-quality VPNs, or arise in otherwise secure VPNs due to misconfiguration by an inexperienced user.

1 Introduction

Virtual Private Networks (hereafter VPNs) are used to protect user privacy, evade censorship, access geo-filtered content, and secure public internet access to name a few. Historically, both commercial and free VPN offerings have suffered vulnerabilities. For example, the CVE database returns 879 results for the keyword 'VPN' [3].

These vulnerabilities can arise due to company malfeasance, bugs introduced in the normal development life cycle, or misconfiguration of an otherwise secure implementation. Vulnerabilities arising from company malfeasance are some of the most sinister as they may happen despite a perfectly functioning application and therefore be impossible for the end-user to detect. For example, in 2024, WIRED discovered that Big Mama VPN was selling user's data for profit [4]. This kind of breach is not unique to just one company, many VPNs have been found to keep logs of user data despite pledges not to do so, sparking concerns that user data is at risk [5].

The vulnerabilities arising from bugs or user error are the types that we focus on detecting with our tool because they are the most feasible to detect. These issues take several different forms, but three of the most common are DNS leaks, WebRTC leaks, and IPv6 leaks.

The most apparent type of leak is the DNS leak, and it is what our tool is focused on detecting. A Domain Name System (hereafter DNS) leak occurs when a user's DNS queries are sent outside the secure VPN tunnel, typically to the user's default DNS provider (often their ISP). This defeats the privacy guarantees of the VPN, as it allows third parties, including network operators or eavesdropping attackers, to observe the domains a user accesses. Even though the rest of the traffic is encrypted and tunneled through the VPN, the leakage of DNS requests can reveal browsing behavior and compromise the anonymity that users are expecting.

A second type of leak is a WebRTC leak. Web Real-Time Communication (hereafter WebRTC) [6] is a browser feature that enables peer-to-peer communication such as video calls or file sharing directly between users. It is common on popular applications like Microsoft Teams or WhatsApp. However, WebRTC can expose a user's real IP address to websites even when a VPN is active. This kind of leak occurs because WebRTC bypasses the VPN's routing, allowing websites to detect the user's actual IP address using Session Traversal Utilities for NAT (hereafter STUN) requests. SO even with a VPN enabled, the user's identity and location may be compromised. Some, but not all VPNs, claim to identify and block these leaks.

The final type of leak we describe is the a type of IP leak that results from the VPN failing to properly process IPv6 traffic. IPv6 leaks occur when a user's device supports IPv6 and sends traffic over the IPv6 protocol while the VPN only routes IPv4 traffic. If the VPN is not configured to handle or block IPv6 traffic, this can lead to data bypassing the VPN tunnel entirely, exposing the user's real IP address and potentially revealing sensitive information. Many VPNs disable IPv6 by default or block it with firewall rules, but systems that don't do this are vulnerable to this subtle but serious privacy risk.

Even when a VPN application is secure and doesn't

suffer from such vulnerabilites out of the box, misconfiguration by the user can still lead to serious privacy or security issues. For instance, users may disable critical features such as DNS leak protection or fail to configure their operating system or browser to ensure all traffic routes through the VPN. Operating systems may have overrides that turned to be turned off that the user is unaware of. For example, a user may neglect to enable or forget to disable the 'kill switch' setting, a feature that automatically blocks internet traffic if the VPN connection drops for any reason. Without the kill switch, the system may revert to using the default (unprotected) network, exposing the user's real IP address and traffic. Similarly, users might allow IPv6 traffic despite their VPN not supporting it, leading to additional leaks. These examples underscore the importance of secure default settings and tools that can detect when a VPN is not protecting traffic as expected.

2 Literature Survey

Khan et al. [7] provided one of the most comprehensive empirical analyses of the commercial VPN ecosystem in 2018. They evaluated 62 VPN providers using a diverse range of active measurements. Their findings revealed that while most VPNs do not engage in overt traffic manipulation, many exhibit concerning issues related to traffic leakage, misrepresented server locations, and insecure default configurations. In particular, the study highlights that 58% of tested VPNs leaked user traffic during tunnel failure, and many failed to prevent IPv6 and DNS leaks. Even well-known providers like NordVPN and ExpressVPN were shown to leak data when their kill-switch mechanisms were disabled or misconfigured. The authors emphasize that such leakages, though unintentional, pose a significant risk to users who rely on VPNs for privacy and censorship circumvention.

Khan and his co-authors also reported on the marketing and business practices of a wider set of 153 VPN providers. They found that 88 of those 153 companies used the marketing phrase "military grade encryption" to help them sell their product. This phrase usually just signals the use of standard AES-256 encryption but serves to leave an impression on non-technical customers who may think that they are offering a special standard of security. They also posed as advertisers interested in purchasing user data and sent cold call emails to those 153 companies. They received several concerning responses inquiring for more details and showing interest in setting up the transaction, but did not push further due to ethical considerations.

In 2019, Bui et al. [8] conducted an extensive investigation into the security of client-side configurations in commercial VPN applications across Windows, macOS, and Ubuntu systems. Their study highlights that even when using well-respected VPN protocols like OpenVPN, many providers deploy configurations that are dangerously insecure. For instance, several VPN services still rely on publicly known pre-shared keys for IPsec tunnels, leaving users vulnerable to man-in-the-middle (MitM) attacks. Similarly, some clients fail to verify server certificates or allow unencrypted communication, effectively defeating the VPN's core purpose of securing user traffic. These issues are not limited to obscure providers, with many mainstream services being found to be vulnerable. In fact, not one of the 30 services examined was completely vulnerability-free, albeit some vulnerabilities were much less critical than others.

Back in 2015, Perta et al. [9] conducted a detailed experimental analysis of 14 popular commercial VPN services, focusing on their real-world security, particularly regarding IPv6 traffic leakage and DNS hijacking vulnerabilities. Their study reveals that despite bold marketing claims, many VPNs fail to protect user privacy under adversarial conditions. A major issue identified is that most VPN clients configure only IPv4 traffic for secure tunneling, leaving IPv6 traffic to leak directly through the native network interface. This vulnerability arises from the system's preference for IPv6 and the VPN software's failure to update the IPv6 routing table. The result is that sensitive user data like browsing activity can bypass the VPN tunnel entirely, even when the user believes they are fully protected.

More recently in 2024, Peter Membrey with ExpressVPN has introduced a new type of DNS leak [10]. Membrey expands the traditional one-mode classification by distinguishing between Type 1 and Type 2 leaks. Type 1 refers to the classic case where DNS requests bypass the VPN entirely and are exposed along with the user's real IP address. In contrast, Type 2 leaks are subtler. In this new type DNS requests are routed through the VPN tunnel but still end up being resolved by untrusted or ISP-assigned DNS servers, which means user activity is still visible to third parties despite the VPN being active. Membrey emphasizes that such leaks can arise without the user's knowledge, often due to default DHCP configurations in places like coffee shops or hotels. These environments can inject DNS settings silently, enabling tracking, censorship, or even surveillance, despite the user believing they are protected.

Another new contribution of the paper is the introduction of Stealth DNS servers. We could find no mention of such 'stealth servers' in literature about VPNs prior to the Membrey paper. These are DNS servers that do not appear in traditional leak tests but still receive and respond to DNS queries. This makes them hard to detect and potentially dangerous, especially if they are configured with-

out user consent. The paper discusses how standard DNS leak detection tools often fail to identify these servers, leaving users with a false sense of security. To mitigate these risks, Membrey proposes server-side solutions such as DNS traffic blocking, transparent DNS proxies, and selective DNS whitelisting. These methods aim to ensure that DNS requests are reliably intercepted or redirected to trusted servers.

3 Design of Our Tool

Our Tool is designed to gather information about DNS leaks or other privacy violating DNS behavior, as well as IPv6 leaks. The tool attempts to make connections and requests outside of the VPN tunnel, in order to verify if the VPN blocks non-VPN traffic. If it does not, requests, like DNS requests, can be made by non-VPN aware applications on the user's system, allowing for privacy-violating behavior. Then, the tool attempts to learn about the DNS servers used to perform DNS requests by the client's system. The server component of our tool acts as the nameserver for a domain, meaning that it will handle requests for its domain and every subdomain. The server tells the client which subdomains to query, so that the server can assosciate incoming DNS requests with a specific client. The client then uses the default web browser on the user's machine to perform the DNS queries, to ensure that leak detection is performed under what is most likely to be the exact configuration that the user will perform most of their DNS queries: while surfing the web. The server will recieve the requests from the resolvers that the client is configured to use, and will log the requests and send the IP addresses of the resolvers back to the client. The client then uses the IP addresses to obtain the name of the organization operating the DNS servers through reverse-DNS lookup, and WHOIS lookup.

Once the client has obtained the above information from before and after connecting to the VPN, it can determine whether privacy-violating behavior has occurred. If a resolver appears in both the before and after set, it is likely a direct VPN leak has occurred, and DNS requests are being routed outside the VPN. If the organizations are the same for the resolvers, but not the IPs, then it is likely that the VPN does not push its own DNS servers, and the user is informed that they may be at risk of deanonymization attacks, especially if they use the DNS servers from their Internet Service Provider.

To check for IPv6 leaks, the tool attempts to establish an ipv6 connection to the server from the client. In the case where IPv6 connection is impossible, the client knows that either their Internet Service Provider blocks IPv6, or their VPN does. Either way, in this case, they are safe from IPv6 leaks specifically. If an IPv6 connection can be established, the server notifies the client of the IPv6 address used to connect to it. The client can then compare the before and after IPv6 addresses, and if they are the same, then an IPv6 leak is likely.

4 Results from Our Tool

Though the tool is theoretically capable (the functionality has been implemented) of detecting IPv6 leaks, the Internet Service Provider for the server that we used to host the server component of the tool dropped incoming IPv6 packets, and as such, we were unable to gain information from that part of the tool.

The setup used to test our tool is as follows: First, the server is running on a Linux server, with root privilege, and the required ports forwarded. The user pc that runs the client is also a Linux PC, using OpenVPN as a VPN service. To test the tool, we used several public OpenVPN configurations, acquired from the VPN Gate service, a service operated by a resarch team at the Graduate School of University of Tsukuba, Japan. This service provides sample OpenVPN (or other VPN service) configurations, with both censorship circumvention, and anonymity being two of its stated goals. We tested a collection of these OpenVPN configurations, including one configuration that we modified to intentionally introduce DNS-related privacy violating behavior.

When testing these VPN configurations, we found that none of the configurations included mechanisms for blocking out-of-VPN traffic, meaning that misconfigured or non-VPN aware software could (knowingly or unknowingly) establish connections outside the VPN, leading to a failure for the VPN to perform the stated goal of anonymity protection. In order to gain protection from this leak, we had to modify the VPN configurations to include the OpenVPN "redirect-gateway" to redirect all traffic through the VPN.

We also found that a subset of the VPN configurations (including the one we modified, as well as some unmodified) did not push new DNS servers to the client. This means that for some configurations, the DNS requests, although routed through the VPN, still went to the same organization responsible for handling DNS requests from before the client connected to the VPN using the configuration. This means that the user could be subjected to deanonymization, including via traffic analysis attacks by the operator of the DNS servers, or by correlating the VPN IP to a user account logged into a site owned by the DNS operators, such as logging into a Google account from the VPN while using Google DNS servers.

Overall, we found the tested OpenVPN configurations to possess critical flaws requiring immediate user attention.

5 Conclusion

Our investigation into the literature and through testing popular VPN software has led us to make several recommendations to users and VPN providers. First, even if the software in use is airtight out of the box, it is easy for non-technical users to break security guarantees via the mutation of just a few settings. It is a good idea for users to not only run VPN leak tests on a freshly installed VPN, but also every time they make a change to their settings/configuration. Users can then attempt to rectify these changes on their own, for instance, through setting iptables rules on Linux, or switch VPNs. Second, it would be a useful feature of VPNs to run an internal leak test when it detects a configuration change, that way non-technical users could be automatically alerted to mistakes they may have made. Finally, we also note that the VPN industry may be ripe for increased government oversight. The results of investigations like those mentioned in the literature review, especially in Khan et al. [8] make clear that there are actors in the VPN ecosystem who will violate ethical limits on marketing and consumer data protection, and even perhaps the law.

References

[1] Gabe Vigderman, Aliza; Turner. 2025 VPN usage statistics. *Security.org*, 2025.

[2] Juliana De Groot. The history of data breaches. *digitalguardian*, 2018.

[3] Mitre. VPN CVEs. `https://cve.mitre.org/cgi-bin/cvekey.cgi?keyword=vpn`, 2025. Accessed: 2025-05-06.

[4] WIRED. This VPN lets anyone use your internet connection. `https://www.wired.com/story/residential-proxy-network-cybercrime-vpn/`, 2024. Accessed: 2025-05-07.

[5] PCMag. 7 vpn services found recording user logs, despite 'no-log' pledge. `https://www.pcmag.com/news/7-vpn-services-found-recording-user-logs-despite-no-`, 2020. Accessed: 2025-05-07.

[6] Mozilla. WebRTC API. `https://developer.mozilla.org/en-US/docs/Web/API/WebRTC_API`, 2025. Accessed: 2025-05-08.

[7] Siddharth Prakash Rao Thanh Bui, Markku Antikainen and Tuomas Aura. Client-side vulnerabilities in commercial vpns. 2019.

[8] Geoffrey M. Voelker Alex C. Snoeren Chris Kanich Mohammad Taha Khan*, Joe DeBlasio* and Narseo Vallina-Rodriguez. An empirical analysis of the commercial vpn ecosystem. 2018.

[9] Gareth Tyson Hamed Haddadi Vasile C. Perta, Marco V. Barbera and Alessandro Mei. A glance through the vpn looking glass: Ipv6 leakage and dns hijacking in commercial vpn clients. *Proceedings on Privacy Enhancing Technologies*, 2015.

[10] Peter Membrey. Shedding light on hidden dangers: A new perspective on dns leaks. 2024.

A Work Distribution

We both worked equally on all parts of the project and didn't have meaningful differences in output.