Poster: Rough Edges for IPv6 in VPNs

Yejin Cho
University of Southern California
Los Angeles, USA
yejincho@usc.edu

John Heidemann
University of Southern California
Los Angeles, USA
johnh@isi.edu

Abstract

How do VPNs interact with IPv6? Our poster shows that *VPNs often leak IPv6 traffic*, failing to provide the promised privacy, and *VPNs often prefer IPv4*, even though IPv6 is available and working. These results use new data from a website for IP identification, coupled with experiments on specific VPN software. We identify the fraction of v6 traffic leaked, and find the root-cause of IPv6 de-preferencing in interactions between address selection in OS and VPN.

ACM Reference Format:
Yejin Cho and John Heidemann. 2025. Poster: Rough Edges for IPv6 in VPNs. In *Proceedings of the 2025 ACM Internet Measurement Conference (IMC '25), October 28–31, 2025, Madison, WI, USA.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3730567.3768612

1 Introduction

Virtual private networks (VPNs) and IPv6 are important parts of today's Internet. VPNs provide privacy and censorship avoidance, and IPv6 offers more addresses and end-to-end connectivity. VPN providers increasingly advertise IPv6 support as a key feature.

Our poster explores **how do VPNs interact with IPv6?** We show two "rough edges" in this interaction: *VPNs often leak IPv6 traffic*, failing to provide the promised privacy, and *VPNs often prefer IPv4* even though IPv6 is available and working.

Our first contribution is to *measure IPv6 leaks in dual-stack VPN traffic*. Our findings show that VPNs should fully support IPv6, rather than blocking or ignoring it, to fully protect user's privacy.

Our second contribution is to *identify IPv6 de-preferencing and its root causes* in dual-stack VPNs. When dual-stack VPNs communicate with a dual-stack destination, the Happy Eyeballs algorithm should prefer IPv6 [8]. Instead, we find that 5 out of 6 VPNs de-preference IPv6 and use IPv4. We show that the root cause of this de-preference is how address selection rules in the operating system interact with VPNs' client-side address. The implementation of most VPNs systematically de-preference IPv6.

Our results use a new data source: logs from WhatIsMyIPAddress.com, a website that helps global users identify their IPv4 and IPv6 addresses. We confirm our findings with direct experimental evaluation of 6 VPN implementations on an Android testbed.

2 Prior Work

VPNs leak IPv6: Leaks occur when IPv6 traffic is not tunneled through the VPN, and instead uses local, non-VPN'ed address. Prior work evaluated client installations on specific platforms [5, 6]. We instead use data from a website to gain a global view of practice. Our data allows us to observe leaks in the wild, and to examine the behavior of all platforms. Since VPNs today span nine platforms including mobile, PC, and browser, this broad view is essential.

Prior work classified VPNs as either "safe" or "leaking" [5, 6]. We go further to measuring what *fraction* of connections leak. We provide a more nuanced picture of real-world VPN use, accounting for factors such as OSes and users' network settings.

VPNs de-preferencing IPv6: Sattler et al. show that browser misimplementations of Happy Eyeballs can cause IPv6 de-preferencing [7]. We are the first to identify VPNs as another factor.

3 How Often Do VPNs Leak IPv6 traffic?

VPN conceals user's traffic and IP addresses from observers. We define an *IPv6 leak* on a dual-stack computer as when a user's IPv4 traffic uses the VPN, but their IPv6 does not. Without the VPN for IPv6, the user does not get the privacy they expected: their non-VPN'ed IPv6 address is externally visible, and their traffic is exposed without the VPN's encryption.

We *observe* IPv6 leaks from paired IP logs in our WhatIsMyIPAddress.com data: We detect a VPN when the IPv4 address is associated with a VPN service, and we identify a *VPN leak* when the IPv6 address (i) does not belong to a VPN service, and (ii) originates from a different organization, and (iii) is classified as ISP AS.

Data from WhatIsMyIPAddress.com lists the IPv4 and IPv6 addresses associated with each user. We augment these addresses by identifying VPN classification from IPinfo [2], organizations from CAIDA's AS2Org [4], and AS classifications from ASdb [10].

We detect 123 VPNs in our data. Figure 1 shows the the 36 VPNs with 100 or more connections per day. The top solid red line gives the percentage of dual-stack users (those where we know both v4 and v6 addresses) running VPNs leak IPv6 addresses, grouped by VPN operator. We sort VPN operators by leak rate, and labeled dual-stack (names are blue with stars), v4-only (red with triangles), and unknown (gray with an X). We identify dual-stack support based on the webpage for each VPN service.

Dual-Stack Traffic: We first consider the percentage of leaks relative to all dual stack users in the solid red line. We observe that VPNs with IPv6 support (blue stars) show lower leak rates than those that are IPv4-only (red triangles). We are surprised that some IPv4-only providers sometimes do protect IPv6 traffic, showing leak rates below 100% (ranging from about 95% down to 5%).

Two special cases are relevant to our classification: First, we sometimes observe partial IPv6 deployment in some VPNs clients. These cases do not satisfy our leak criteria (i to iii), because the IPv6 address does belong to the VPN's AS, so they appear as unexpected protection, not a leak. Second, some traffic classified as safe, dual-stack sessions is due by Chrome prefetching. The Chrome browser speculatively loads resources in advance to improve page load

IMC '25, Madison, WI, USA

ACM ISBN 979-8-4007-1860-1/2025/10
https://doi.org/10.1145/3730567.3768612

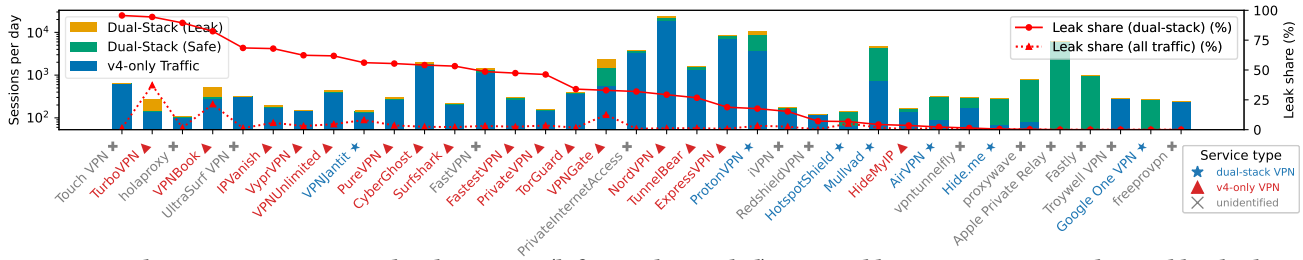

Figure 1: Bars show VPN sessions per day, by service (left axis, log-scaled), grouped by VPN operator and sorted by dual-stack leak rate. Lines show percentage of leaks relative to dual-stack sessions (solid red), and all sessions (dashed red).

performance. When it does so, the initial fetch of main page comes from Chrome's AS, but subsequent requests follow through the user's VPN. Thus the prefetched IPv6 connection appears outside the VPN and would be misclassified as a leak. To correct this case we treat traffic from Google's IPv6 addresses [1] as safe.

All Traffic: The above leak analysis considers only dual-stack visitors, where WhatIsMyIPAddress.com observes both IPv4 and IPv6 address. However, many visitors have only an IPv4 address. The dotted red line shows the fraction of IPv6 leaks across *all* users (dual-stack and IPv4-only), showing much lower and ranges from about 0% to 30% for all VPN packages. IPv4-only VPNs leak more frequently (mean 6.5%, median 2.8%) than dual-stack VPNs (mean 2.9%, median 1.8%) or unidentified services (mean 0.9%, median 0.4%) Looking at all sessions shows that most IPv4-only VPN packages protect their users because they *do* usually disable IPv6, but a few percent of the time they don't.

4 Why do VPNs De-preference IPv6?

We next examine, for users of dual-stack VPNs, if they actually use v4 or v6 in practice. We find that most commercial VPNs' configurations lead to v6 de-preference, even though the Happy Eyeballs algorithm (HE) prefers IPv6 [8]. We observe protocol preference by analyzing data from WhatIsMyIPAddress.com which uses DNS and embedded objects to identify v4 and v6 address and show their preference order. We find that 98% of VPN users de-preferred IPv6.

Protocol Preferences and a Potential Root Cause: HE [8] races A and AAAA queries, favoring IPv6 while keeping latency low. In normal Internet use, clients usually have both *public* IPv4 and IPv6 addresses. RFC-6724's prioritization ranks public IPv4 and IPv6 equally, so both address families are equally ranked options for making a connection, and HE would favor v6, but v4 is possible if v6 fails. In contrast, VPNs often assign *private* IPv4 and IPv6 addresses to VPN's tunnel, using RFC-1918 space for IPv4 and a ULA address for IPv6. Here, with private IPv4 and ULA IPv6 addresses, RFC-6724 prioritization produces a different outcome.

Prioritization rules rank public above private above link-local addresses [9]. De-prioritization occurs because public and private IPv4 addresses are consider equivalent priority, but v6 addresses have a ULA source address from the VPN and a GUA destination, causing them to be de-prioritized than v4 address. Thus, the use of private addresses by VPN, coupled with the different prioritization rules for v4 and v6 *systematically de-prioritize v6 use for VPN users.*

We see two potential solutions to avoid this de-prioritization. First, VPN operators could assign GUA address instead of ULA. We expect that VPN operators assign ULAs for v6 to directly mirror the use of private address in v4. However, given the wealth of v6 addresses, they could easily assign GUAs. (GUA pose no particular privacy risk as it is only used for VPN tunnel.) Alternatively, the IETF could revisit prioritization or assignment. Potentially v6 prioritization could rank ULA addresses higher, or IETF could create a new class of VPN-specific ULAs that are prioritized with GUAs.

Either way, most current VPNs will continue to de-prioritize v6 traffic if they continue to use ULA addresses. Some VPN vendors acknowledged this issue but indicate that GUA support is planned "eventually" and not provided today [3].

Validating Root Causes with Real-World VPNs: Finally, to confirm our understanding of VPN de-preferencing of IPv6, we tested 6 commercial VPNs that advertise support of native v6. We installed each vendor's VPN client on an Android device, activated the VPN, and inspected the assigned tunnel interface using the Android Debug Bridge.

We find that one out of the six VPN services (ProtonVPN) assigned GUA on the tunnel interface. The other five providers (Mullvad, AirVPN, hidemeVPN, Perfect Privacy, and Anonine) assigned ULAs. These results suggest our analysis of how HE and prioritization interact with IPv6 above is a plausible explanation for the IPv6 de-prioritization we observe in our data.

Acknowledgments

We thank Chris Parker, CEO of WhatIsMyIpAddress.com and IPInfo's Academic Dataset support for providing the relevant dataset for research. This work is partially supported by the U.S. NSF through Internet Map (CNS-2212480) and BRIPOD (OAC-2530698).

References

[1] Chrome prefetch proxy IP list. URL: https://www.gstatic.com/chrome/prefetchproxy/prefetch_proxy_geofeed.
[2] IPInfo proxy/VPN API. URL: https://ipinfo.io/products/proxy-vpn-detection-api.
[3] mullvadvpn-app Linux IPv6 support. URL: https://github.com/mullvad/mullvadvpn-app/issues/3608.
[4] The CAIDA UCSD AS Organizations dataset – [2025-08-01]. https://catalog.caida.org/dataset/as-organizations. CAIDA, UC San Diego.
[5] M. T. Khan, J. DeBlasio, G. M. Voelker, A. C. Snoeren, C. Kanich, and N. Vallina-Rodriguez. An empirical analysis of the commercial VPN ecosystem. In *Internet Meas. Conf.*, pages 443–456. ACM, 2018. doi:10.1145/3278532.3278570.
[6] R. Ramesh, L. Evdokimov, D. Xue, and R. Ensafi. VPNalyzer: Systematic investigation of the VPN ecosystem. In *Proceedings 2022 Network and Distributed System Security Symposium*. Internet Society. doi:10.14722/ndss.2022.24285.
[7] P. Sattler, M. Kirstein, L. Wüstrich, J. Zirngibl, and G. Carle. Lazy eye inspection: Capturing the state of happy eyeballs implementations. arXiv:2412.00263[cs], doi:10.1145/3730567.3732925.
[8] D. Schinazi and T. Pauly. Happy eyeballs version 2: Better connectivity using concurrency. URL: https://www.rfc-editor.org/rfc/rfc8305, doi:10.17487/RFC8305.
[9] D. Thaler, R. P. Draves, A. Matsumoto, and T. Chown. Default address selection for internet protocol version 6 (IPv6). URL: https://datatracker.ietf.org/doc/rfc6724, doi:10.17487/RFC6724.
[10] M. Ziv, L. Izhikevich, K. Ruth, K. Izhikevich, and Z. Durumeric. ASdb: a system for classifying owners of autonomous systems. In *Proc. Internet Measurement Conference*, pages 703–719. ACM, 2021. doi:10.1145/3487552.3487853.