Extended Abstract: Traffic Splitting for Pluggable Transports

Anna Harbluk Lorimer
University of Chicago
Chicago, IL, USA

Rob Jansen
U.S. Naval Research Laboratory
Washington, DC, USA

Nick Feamster
University of Chicago
Chicago, IL, USA

ABSTRACT

As censors continue to develop more advanced technologies to police access to the Internet, new techniques are also needed to promote Internet freedom. Traffic splitting is one technique that has been shown to defend against various privacy attacks and to improve network performance, but it has not been evaluated in the context of censorship-resistant systems. In this extended abstract we outline a design for traffic splitting for pluggable transports that uses what we call a "shim" pluggable transport and modified Tor Bridges as proxies. We describe our hypotheses and how we plan to evaluate our implementation to determine the extent to which pluggable transports that employ traffic splitting can improve performance, protect against website fingerprinting attacks, and improve resistance to detection and blocking by a censor.

KEYWORDS

anti-censorship, traffic splitting, pluggable transports, tor

1 INTRODUCTION & BACKGROUND

Since its deployment in 2002, Tor [3] has provided a way for users to browse the Internet while being protected from privacy attacks and censorship. Tor is a particularly important tool for anti-censorship as it provides a way for users using the Internet under censorship to access censored content safely without the threat of persecution. Gaining access to Tor can be difficult for users as censors have a vested interest in preventing users from accessing it. Thus, some Tor guard nodes known as *Tor Bridges* are kept secret so that it is more difficult for a censor to block access. In 2021, there were approximately 2400 bridges in operation [5] and there are a variety of out of band methods for obtaining the list of bridge addresses [13].

Pluggable transports (PTs) are cross-application tools that wrap application traffic to provide a censorship-resistant layer through obfuscation, noise, mimicry, collateral damage, or a combination of techniques such that a censor cannot determine if the traffic should be blocked. At the time of writing, there are five PTs that are widely deployed in the Tor ecosystem, however, the pluggable nature of these anti-censorship tools allow developers and users to write and "plug in" arbitrary anti-censorship channels. Prior work has shown that the degree of protection, as well as latency and bandwidth, varies widely between PTs [20] and that performance of a given PT has been linked to its underlying technology (i.e., protocol) [19].

Although Tor and PTs enable users around the world to fight back against Internet censorship and anti-access policies, neither are invulnerable to attack. Censors have targeted Tor bridges [11] and Website fingerprinting attacks are capable of linking a user to the destinations they visit and could thus be employed for selective blocking [9], while recent work shows how deep learning can be used by censors to develop precise protocol classifiers [21].

In this extended abstract, we explore *traffic splitting*, where network traffic is divided across two or more paths, as a means of defending against censors and improving PT performance. The key insight behind why traffic splitting might be a successful defence is that an attacker sees only a fraction of the information they need to correctly identify traffic they deem undesirable. Typically an attacker's classifier is trained on a complete network trace for a target website or protocol that is then used to compare against a user's traffic traces. If a user splits their traffic so that an attacker sees only a partial trace when they attempt to classify it, the accuracy of the classifier would be greatly reduced [6, 12, 22].

Traffic splitting has been implemented and evaluated for Tor network traffic where it has shown promise as a defence against privacy attacks [1, 2, 6, 12, 22] and in some cases can improve performance [14], but it has not been studied in the context of PTs. Our extended abstract proposes a plan for how we might fill this research gap. We describe the censorship model that we will consider in § 2, we explain in § 3 our PT traffic splitting design that enables us to simultaneously provide traffic splitting support to all PTs without modifying the code of each one, and we present our research hypotheses and evaluation plans in § 4 and Appendix A respectively.

2 CENSORSHIP MODEL

In our model, the censorship resistor is using a PT with traffic splitting support to evade Internet censorship.

This work considers a censor that is capable of passively observing and classifying traffic traces with a high degree of accuracy when traffic is not being split [10, 18, 23]. This censor is also monitoring Tor bridges and is actively engaged in blocking bridges either via bridge enumeration or blocking based usage [9]. Finally, we assume that while a censor can observe all N paths across which traffic is being split, they lack the infrastructure and data processing capability to correlate all paths with each other but is nonetheless still attempting to classify traffic [6, 22].

3 PT TRAFFIC-SPLITTING DESIGN

We plan to add traffic splitting support to existing, supported PTs using a *shim* pluggable transport to split the traffic across multiple paths between a Tor Browser and a gateway Tor bridge, while using existing PT bridges as proxies.

In Figure 1, we show how we can deploy existing pluggable transports to support traffic splitting using lyrebird (previously known as obfs4) as an example. We note that our shim design enables existing and future PTs to automatically benefit from traffic splitting without each of them needing to individually implement

Free and Open Communications on the Internet (1), 29–31

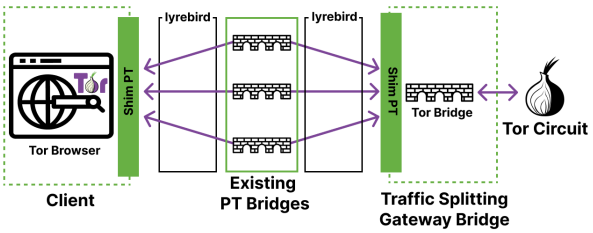

Figure 1: Our proposed design for adding traffic splitting support for PTs using Tor bridges and a shim PT with lyrebird as an example. At the client, outgoing traffic is split across three PT connections (in this case lyrebird) by sending the traffic to three existing PT bridges acting as proxies. At the gateway bridge, traffic is correlated and sent on to a typical Tor circuit. When traffic is sent from the gateway bridge to the client, the gateway bridge acts as the splitter and the client acts as the correlator. The opaque green boxes indicate the placement of the shim PT and the dashed green boxes indicate where code changes in existing Tor infrastructure are necessary to support traffic splitting. The existing PT bridges are in a solid green box to indicate that only configuration, not code, changes need to be made.

a traffic splitting design into their client-side or server-side. Shown in the green regions are the components necessary to implement traffic splitting: the shim PT deployed at the client and the gateway bridge split and reassemble traffic across an arbitrary number of connections using any supported PT, and the existing PT bridges act as intermediary proxies. At the client, a traffic-splitting PT acts as the go-between between the Tor browser and the PT that the user chooses to transport their traffic. The shim contains logic and scheduler for splitting the client traffic across N destinations and implements the Turbo Tunnel design pattern to provide a reliability layer [4]. The user picks which PT they want to use and the shim client then opens the necessary SOCKS proxy connections to the N PT client processes, which will establish connections to remote PT server processes and locally proxy their shares of the split traffic across those connections.

The existing PT bridges, running unmodified PT server code, accept incoming connections. Rather than unwrapping these connections and locally proxying the application data to a Tor process, they instead send all traffic to the server side of our shim PT running on the gateway bridge. This gateway bridge acts as a traditional Tor bridge: it correlates connections from a single client (with TurboTunnel style reliability) and forwards it to a Tor process and into the Tor network.

There are different algorithms that can be used to determine how exactly the traffic is split at a packet level. We plan to implement CoMPS-style consistent splitting (round robin, uniform random, and weighted random) [22] as part of our performance evaluation described in § 4.

4 HYPOTHESES

In this section we present our hypotheses about the impact of traffic splitting on the performance, website fingerprinting defences, and anti-censorship properties on PTs.

4.1 Performance

We expect that the performance of PTs (bandwidth, latency, throughput), under certain conditions, can be improved using traffic splitting [14]. Any performance improvements will be the result of the particular details and behaviour of a PT, the scheduling algorithm used to split the traffic, and the available bandwidth across the different paths. We suspect that traffic splitting will positively impact performance when PTs like Snowflake are experiencing performance degradations and can switch to more performant proxies using traffic splitting.

4.2 Website Fingerprinting

We expect that traffic splitting will improve protection against website fingerprinting attacks in the PT context as prior work has shown that it does outside of the PT context [6, 12, 22].

4.3 Resistance to Censorship

Censors use methods other than website fingerprinting to interfere with traffic, and we expect that resistance to host-based blocking or bridge blocking can be improved using traffic splitting. One potential implementation of traffic splitting is to switch connections after a variable time limit to prevent attacks based on identifiers such as DTLS connetion duration [21]. Further, our proposal for deploying traffic splitting with existing PTs may pave the way towards more ephemeral circumvention hosts, thwarting host-based classification techniques. In contexts other than Snowflake, the ability to split traffic across multiple bridges, or perhaps even across multiple PTs, will make it much more difficult for a censor to identify anti-censorship traffic or interfere with high-usage bridges.

There is of course the possibility that splitting PT traffic makes the traffic more identifiable to a censor. Since traffic splitting is not ubiquitous network behaviour, split PT traffic may appear anomalous and therefore a censor can distinguish between normal use of a PT and split use. In Appendix A, we describe our evaluation plan to determine what effect splitting PT traffic has on detectibility.

ACKNOWLEDGMENTS

The authors would like to thank Cecylia Bocovich for her contributions to the design of the shim PT and for her contributions to the discussion of traffic splitting for Snowflake.

REFERENCES

[1] Oscar Arana, Hector Benítez-Pérez, Javier Gomez, and Miguel Lopez-Guerrero. 2021. Never Query Alone: A Distributed Strategy to Protect Internet Users from DNS Fingerprinting Attacks. *Comput. Netw.* 199, C (nov 2021), 12 pages. https://doi.org/10.1016/j.comnet.2021.108445

[2] Wladimir De la Cadena, Asya Mitseva, Jens Hiller, Jan Pennekamp, Sebastian Reuter, Julian Filter, Thomas Engel, Klaus Wehrle, and Andriy Panchenko. 2020. TrafficSliver: Fighting Website Fingerprinting Attacks with Traffic Splitting. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security* (Virtual Event, USA) *(CCS '20)*. Association for Computing Machinery, New York, NY, USA, 1971–1985. https://doi.org/10.1145/3372297.3423351

[3] Roger Dingledine, Nick Mathewson, and Paul F. Syverson. 2004. Tor: The Second-Generation Onion Router. In *The 13th USENIX Security Symposium.*

[4] David Fifield. 2020. Turbo Tunnel, a good way to design censorship circumvention protocols. In *Free and Open Communications on the Internet*. USENIX. https://www.usenix.org/system/files/foci20-paper-fifield.pdf
[5] ggus. 2022. (2022). https://blog.torproject.org/run-a-bridge-campaign/
[6] Sébastien Henri, Gines Garcia-Aviles, Pablo Serrano, Albert Banchs, and Patrick Thiran. 2020. Protecting against Website Fingerprinting with Multihoming. *Proceedings on Privacy Enhancing Technologies* 2020 (2020), 89 – 110. https://api.semanticscholar.org/CorpusID:211224137
[7] Rob Jansen, Jim Newsome, and Ryan Wails. 2022. Co-opting Linux Processes for High-Performance Network Simulation. In *2022 USENIX Annual Technical Conference (USENIX ATC 22)*. USENIX Association, Carlsbad, CA, 327–350. https://www.usenix.org/conference/atc22/presentation/jansen
[8] Rob Jansen and Ryan Wails. 2023. Data-Explainable Website Fingerprinting with Network Simulation. *Proceedings on Privacy Enhancing Technologies* 2023, 4 (2023). https://doi.org/10.56553/popets-2023-0125
[9] Sheharbano Khattak, Tariq Elahi, Laurent Simon, Colleen M. Swanson, Steven J. Murdoch, and Ian Goldberg. 2016. SoK: Making Sense of Censorship Resistance Systems. *Privacy Enhancing Technologies* 2016, 4 (2016), 37–61. https://murdoch.is/papers/popets16makingsense.pdf
[10] Sheharbano Khattak, Mobin Javed, Philip D. Anderson, and Vern Paxson. 2013. Towards Illuminating a Censorship Monitor's Model to Facilitate Evasion. In *Free and Open Communications on the Internet*. USENIX. https://censorbib.nymity.ch/pdf/Khattak2013a.pdf
[11] Zhen Ling, Junzhou Luo, Wei Yu, Ming Yang, and Xinwen Fu. 2012. Extensive analysis and large-scale empirical evaluation of tor bridge discovery. In *2012 Proceedings IEEE INFOCOM*. 2381–2389. https://doi.org/10.1109/INFCOM.2012.6195627
[12] Jan Pennekamp, Jens Hiller, Sebastian Reuter, Wladimir De la Cadena, Asya Mitseva, Martin Henze, Thomas Engel, Klaus Wehrle, and Andriy Panchenko. 2019. Multipathing Traffic to Reduce Entry Node Exposure in Onion Routing. In *2019 IEEE 27th International Conference on Network Protocols (ICNP)*. 1–2. https://doi.org/10.1109/ICNP.2019.8888029
[13] The Tor Project. 2023. *BRIDGES*. Tor. https://tb-manual.torproject.org/bridges/
[14] K. Sangeetha and K. Ravikumar. 2015. A Novel Traffic Dividing and Scheduling Mechanism for Enhancing Security and Performance in the Tor Network. *Indian Journal of Science and Technology* 8 (03 2015), 689. https://doi.org/10.17485/ijst/2015/v8i7/62882
[15] Tor Anti-Censorship Team. 2023. *Technical Overview*. Tor. https://gitlab.torproject.org/tpo/anti-censorship/pluggable-transports/snowflake/-/wikis/Technical%20Overview
[16] Tor Anti-Censorship Team. 2023. *WebTunnel*. Tor. https://gitlab.torproject.org/tpo/anti-censorship/pluggable-transports/webtunnel
[17] Tor 2023. *lyrebird*. Tor. https://gitlab.torproject.org/tpo/anti-censorship/pluggable-transports/lyrebird
[18] Michael Carl Tschantz, Sadia Afroz, Anonymous, and Vern Paxson. 2016. SoK: Towards Grounding Censorship Circumvention in Empiricism. In *Symposium on Security & Privacy*. IEEE. https://www.eecs.berkeley.edu/~sa499/papers/oakland2016.pdf
[19] Zeya Umaya. 2023. *PTPerf: On the performance evaluation of Tor Pluggable Transports*. https://zenodo.org/record/8352814
[20] Zeya Umayya, Dhruv Malik, Devashish Gosain, and Piyush Kumar Sharma. 2023. PTPerf: On the performance evaluation of Tor Pluggable Transports. *Internet Measurement Conference*. https://conferences.sigcomm.org/imc/2023/program/#50
[21] Ryan Wails, George Arnold Sullivan, Micah Sherr, and Rob Jansen. 2024. On Precisely Detecting Censorship Circumvention in Real-World Networks. In *Network and Distributed System Security Symposium*. The Internet Society. https://www.robgjansen.com/publications/precisedetect-ndss2024.html
[22] Mona Wang, Anunay Kulshrestha, Liang Wang, and Prateek Mittal. 2022. Leveraging strategic connection migration-powered traffic splitting for privacy. *ArXiv* abs/2205.03326 (2022). https://api.semanticscholar.org/CorpusID:248562892
[23] Zhongjie Wang, Yue Cao, Zhiyun Qian, Chengyu Song, and Srikanth V. Krishnamurthy. 2017. Your State is Not Mine: A Closer Look at Evading Stateful Internet Censorship. In *Internet Measurement Conference*. ACM. http://www.cs.ucr.edu/~krish/imc17.pdf

A EVALUATION PLAN

We plan to evaluate the performance of our implementation of Traffic Splitting support using Shadow [7] and as previously mentioned, we will implement three different traffic splitting algorithms and assess their different effects on performance. For each PT that we use for evaluation will investigate and evaluate its own unique considerations for getting the most performant splitting setup. Additionally we will use canonical Website Fingerprinting attacks and network classification techniques to evaluate the defences [8, 21].

When choosing which PTs to use to evaluate the shim PT, we need to ensure that we are using existing PTs that are supported by Tor, popular to use, and cover a variety of protocols. To that end, we will consider three PTs composed with the shim traffic-splitting PT described in Section 3:

Snowflake. Snowflake is a pluggable transport where clients make WebRTC connections to *ephemeral proxies* operated by volunteers [15]. The NAT traversal features and browser support of WebRTC allow for these proxies to be run easily on home networks and through browser addons.

Snowflake is an interesting case study for traffic splitting due its use of ephemeral proxies, with a wide range of network stability and bandwidth characteristics. The Turbo Tunnel design pattern [4] implemented in Snowflake makes splitting traffic across multiple Snowflake proxies easy; the client and the server sides already have end-to-end reliability and session state that takes care of re-ordering multi-path packets. However, initial attempts to split traffic naively among multiple Snowflake proxies has had either no improvement on performance or a negative effect[1]. This work will consider more advanced techniques for splitting traffic in Snowflake.

WebTunnel. WebTunnel is a Pluggable Transport that runs on HTTP and imitates web browsing activities based on Frolov and Wustrow's HTTPT [16].

lyrebird (obfs4). Previously known as obfs4, lyrebird is a pluggable transport that provides a layer of obfuscation for TCP. Traffic that uses lyrebird appears completely random so that an observer is unable to identify the traffic [17].

[1] https://bugs.torproject.org/tpo/anti-censorship/pluggable-transports/snowflake/25723