Is Custom Congestion Control a Bad Idea for Circumvention Tools?

Anonymous Author(s)

Abstract

Circumvention proxies often have to operate under adverse network conditions, especially over cross-border links with high packet loss. These scenarios motivated the development of proxies that implement custom congestion control algorithms (CCAs) that aggressively sustain high sending rates by disregarding standard congestion responses. In this paper, we argue that such custom CCAs are fundamentally at odds with the core principles of censorship circumvention. Using Hysteria and TCP-Brutal as case studies, we demonstrate how these custom CCAs produce traffic patterns that significantly deviate from standard TCP and QUIC behaviors, and further develop simple, threshold-based classifiers to show how a censor can distinguish such proxy traffic by its lack of response to congestion signals. We emphasize that any performance optimizations must be grounded in standard protocol behaviors to maintain the indistinguishability required for effective censorship circumvention.

1 Introduction

The global escalation in censorship efforts has heightened the demand for circumvention tools. In a typical scenario, a user in a censored region connects to a proxy server located outside the jurisdiction to access application servers that are otherwise blocked. These proxy connections face several unique challenges inherent to the adversarial nature of censorship circumvention, with one of the most extensively studied being the need for proxies to obfuscate their traffic to avoid detection and blocking [17]. Over the past decades, proxy developers have continuously adopted various obfuscation strategies, whereas censors deployed increasingly sophisticated fingerprinting techniques to detect and disrupt proxy activities [1, 4, 19].

A less examined yet critical issue affecting the usability of proxy tools is their performance under unreliable and lossy network conditions. Proxy connections often need to traverse geographically distant and topologically disparate networks, in particular cross-border or intercontinental links that are characterized by frequent congestion and high packet loss. For example, prior studies have measured that inbound traffic crossing the Chinese border can experience packet loss rates ranging from 10% to 50% [24]. To optimize performance under such network conditions, some proxy tools have adopted custom, non-standard Congestion Control Algorithms (CCAs) designed to aggressively transmit packets in spite of packet loss. Notable examples include Hysteria [7] and TCP-Brutal [9], which maintain higher sending rates compared to conventional CCAs, allowing them to outcompete co-existing flows and secure an (unfairly) disproportionate share of throughput on congested links.

We argue in this paper that such custom CCA implementations fundamentally contradict the design principles of censorship circumvention tools. The essence of effective censorship circumvention lies in obfuscating proxy traffic such that it is indistinguishable from the broader category of traffic valued by censors, thereby leveraging "collateral damage" to achieve some degree of unblockability. Over the years, practitioners in the field have recognized that to do this well, circumvention tools must adopt standard, ubiquitous protocols like TLS and strive to align their behaviors with those expected from mainstream implementations (e.g., browsers). Custom congestion control, on the other hand, is a clear deviation from this principle: by reacting to packet loss in a manner inconsistent with standard TCP/QUIC behaviors, these implementations not only contribute to network congestion but also generate traffic patterns that diverge from the norm – fundamentally at odds with the goal of blending in.

We characterize the custom CCAs implemented by Hysteria and TCP-Brutal by quantifying their aggressive sending behaviors. Using a controlled testbed, we emulate various network conditions and demonstrate that these tools could be reliably differentiated from standard TCP and QUIC implementations across multiple mainstream CCAs, including TCP Cubic, YeAH, Vegas, HTCP and BBR. Furthermore, these distinctions generalize across varying round-trip-times (RTTs) and packet loss rates. To illustrate the feasibility of exploiting these behaviors for proxy detection, we develop a proof-of-concept two-stage classifier and evaluate it on a dataset of synthetic flows generated for a thousand different network conditions. Even with a simple threshold-based approach, the classifier is able to differentiate the custom CCA from standard TCP/QUIC flows with a rare few false positives and negatives.

We do not discount the importance of performance for circumvention tools; rather, we argue that such solutions must be sought within the framework of standard protocols and behaviors so that performance optimization does not come at the expense of fingerprintability. In our experiments, we found that BBR performs reasonably well under most network conditions, coming close to the performance levels achieved by the custom CCA, particularly when packet loss remains below 20%. As such, we see no compelling justification for adopting custom CCAs in circumvention tools – after all, any marginal performance gains are futile if the tools get blocked from fingerprintable traffic patterns.

2 Background

News, anecdotes, and measurement studies collectively suggest that network interferences like censorship have been on the rise on a global scale [12, 15, 16], motivating users to resort to circumvention tools like proxies [17]. The use of proxies in censored regions necessitates obfuscation mechanisms – otherwise, the channel itself could be fingerprinted and blocked. Over the years, the circumvention community has adopted various obfuscation approaches, with the shared goal of blending the circumvention traffic into other traffic categories that censors are reluctant to block [3, 11, 18, 20]. Still, the cycle of adaptation continues as censors deploy increasingly sophisticated fingerprinting methods to differentiate circumvention traffic, such as protocol discrepancies or traffic patterns [4, 5, 21, 22].

Proceedings on Privacy Enhancing Technologies YYYY(X), 1–5

https://doi.org/XXXXXXX.XXXXXXX

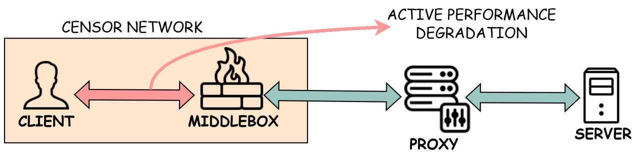

Figure 1: Testbed Setup. The middlebox can actively introduce loss or constrain bandwidth of traffic. It can also take advantage of the natural performance degradation on cross-border links. It can observe traffic before and after the middlebox device.

Real-world throttling events emerged as as an increasingly common methods of Internet censorship in various countries [13, 14]. In Iran, throttling has been deployed to limit access by deliberately reducing bandwidth, effectively slowing down connections to discourage the use of certain services, especially during periods of political unrest [2]. Similarly, Russia used throttling as a censorship technique by intentionally slowing down Twitter's traffic, impacting users' ability to access the platform [23]. In China, network congestion at the borders – often referred as the "Great Bottleneck of China" – significantly reduces network performance for transnational connections [24]. Persistent packet loss from congestion and deliberate throttling affects the performance of circumvention proxies in these regions, motivating the development of circumvention tools that use custom congestion control algorithms like Hysteria [7] and TCP-Brutal [9].

3 Threat Model

We envision an adversary aiming to detect proxy tools that aggressively try to obtain better performance in constrained network environments. The adversary capabilities are based on recent and past documented instances of adversaries intentionally restricting end-user performance [13, 14, 23]. Specifically, we assume an adversary along the network path, akin to an ISP, with the ability to impose restrictions on packet flows by either introducing artificial packet losses or constraining bandwidth. Further, the functioning of the proxy tools is assumed to be public knowledge, and the adversary can based on it to develop heuristics to detect their usage.

Testbed Setup We build our testbed following the threat model. We use four physical machines, all of which are configured in-lab to run Debian instances. We follow the topology as shown in Figure 1 and assume that a client machine residing inside a censored networks is connected to a proxy host, traversing through a hop where the censor deploys a middlebox. The client and middlebox are assumed to be within the censor's network boundary with other entities outside the censor's purview. We configure and run appropriate proxy tools under test (and popular CCAs) on the client and proxy machine. The link between the client and middlebox can be configured with arbitrary network conditions such as loss, RTT or bandwidth constraints using the `netem` and `tbf` modules in `tc` [6]. The analysis for detecting the presence of studied proxy tools is conducted on the middlebox machine, which also captures raw PCAPs for reference.

4 Behavior Characterization

The focus of our investigation is a class of transports used by obfuscated proxy protocols that are designed to enhance the performance

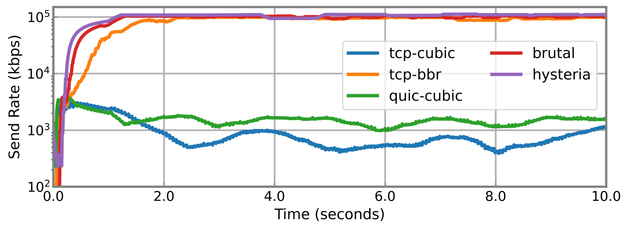

Figure 2: Comparing Aggressive Proxy Tools with Traditional CCAs. The link has a bandwidth of 100 Mbps, an RTT of 60ms, and an uniform packet loss rate of 5%.

over unreliable and lossy networks. Notable examples of these protocols include Hysteria [7] (QUIC-based) and TCP-Brutal [9]. These tools are promoted for use in specific scenarios such as tunneling cross-border connections, where network congestion leads to packet loss rates typically in the range of 5% to 15%, with peak loss rates as high as 50% [24]. To optimize the proxy's performance under such situations, Hysteria and TCP-Brutal use custom, non-standard Congestion Control Algorithms (CCAs) that aggressively transmit packets at a higher rate and outcompete standard CCAs on congested links.

Before we proceed with quantifying these deviations, we first provide the high-level intuition that guided our study: traditional CCAs – whether loss-based, delay-based, or rate-based – implement a "back off" mechanism in response to congestion signals. This mechanism aims to avoid excessive bandwidth usage and to ensure fairness when network conditions change, such as increased RTTs or reduced bandwidth capacity. On the other hand, the custom CCAs used by Hysteria and TCP-Brutal are designed to sustain high performance even under congestion. Instead of cooperating with competing flows by adapting to congestion, these protocols disregard congestion signals and even attempt to *compensate* for packet loss by further *increasing* their sending rate, therefore maintaining the throughput configured by the user-defined "download bandwidth". Such behavior represents a fundamental violation of the congestion control principles defined in both TCP and QUIC. Apart from disrupting the intended fairness of network resource allocation and inducing potential congestion collapse, such behaviors also generate distinctive traffic patterns that fundamentally undermine the objectives of censorship circumvention.

In this section, we empirically demonstrate the non-compliant behaviors of Hysteria and Brutal through a series of experiments. In § 4.1, we illustrate how the behaviors of Hysteria and Brutal diverge from those of typical TCP Cubic or QUIC Cubic under a specific set of network conditions. § 4.2 extends this analysis by showing consistent deviations across various Hysteria and Brutal configurations, in comparison to several loss-based TCP CCAs, such as Cubic, YeAH, HTCP, and Vegas. In § 4.3, we generalize these observations to a broad range of network conditions with varying loss rates and RTTs. Finally, in § 4.4, we further distinguish Hysteria and Brutal against rate-based CCAs like BBR.

4.1 Aggressive Sending Rates

Figure 2 shows the sending rates of five connections: Hysteria, TCP Brutal, QUIC-Cubic, TCP-Cubic, and TCP-BBR, over a link with a

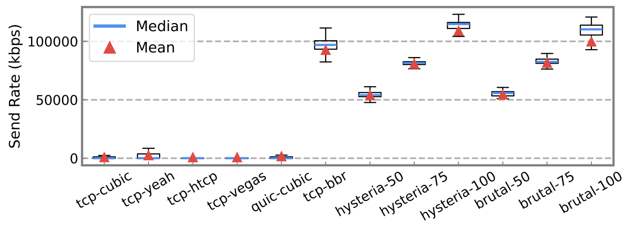

Figure 3: Differentiating Hysteria and Brutal from Loss-based CCAs based on Sending Rate. Sending rates over time is captured by box-plots. Hysteria and Brutal are configured at 50Mbps, 75Mbps, and 100Mbps on a link with 100Mbps bandwidth, 60ms RTT, and 5% uniform loss rate.

bandwidth of 100 Mbps, an RTT of 60 ms, and a random packet loss rate of 5% (uniform loss). For Hysteria and Brutal, the download bandwidth was configured to match the link capacity, as recommended by their protocol specifications [8].

As expected, Cubic, in both TCP and QUIC, being a loss-based CCA, suffers significant performance degradation at a 5% loss rate, leading it to transmit at rates several orders of magnitude lower than those achieved by Hysteria and Brutal. The custom CCA adopt a fixed-sending-rate model, effectively utilizing the available bandwidth regardless of packet loss. This difference lends us the intuition that in networks with non-negligible packet loss, such as cross-border or transcontinental links, Hysteria and Brutal can be easily distinguished from CCAs that interpret packet loss as a congestion signal and consequently back off from sending.

It is worth noting, however, that TCP-BBR [1], which adopts a model/rate-based CCA that does not directly respond to packet loss, exhibits sending behavior similar to that of Hysteria and Brutal, which necessitates further differentiation.

4.2 Differentiation from loss-reactive CCAs

Figure 3 compares the sending rates of Brutal and Hysteria with those of other standard CCAs, including TCP-YeAH, HTCP, and Vegas, in addition to Cubic and BBR. Additionally, Brutal and Hysteria are configured with different settings, including when the “Download Bandwidth” is set below the actual link capacity. In these configurations, they effectively acts as a self-imposed rate limiter. Despite this, both custom tools, which rely on a fixed, pre-configured sending rate, remain reliably distinguishable from loss-based CCAs.

BBR, however, is an exception among the standard CCAs. As a rate-based algorithm, BBR aims to maximize available network capacity rather than react to packet loss as a congestion signal. Surprisingly, BBR outperforms Hysteria and Brutal when these tools are configured with a lower-than-actual bandwidth. It’s worth noting that Hysteria has built-in support for BBR as an alternative to its custom CCA, presumably due to BBR’s ability in sustaining stable throughput even in moderately lossy environments.

4.3 Generalizing across Network Conditions

We demonstrate that the observed differentiating behaviors between the custom and standard CCAs generalize across varying network conditions – specifically network reliability (loss rate) and latency (RTT). We conducted experiments using our testbed to empirically measure the performance of different CCAs under varying loss-RTT combinations, using the Mathis Equation [10]:

$$Throughput = \frac{MSS}{RTT\sqrt{P_{loss}}} \qquad (1)$$

as a shared baseline for comparison [2]. Figure 4 shows the results as a heatmap, with the value in each entry corresponding to the ratio of the specific CCA’s performance relative to Mathis estimation.

We found that Mathis Equation provides an accurate estimation for Cubic under most of the tested network conditions, whereas Brutal and Hysteria consistently send at multiples of the estimated rates, especially under high loss-rate and/or high RTT scenarios. This is consistent with our understanding that the absence of slowing down under packet loss clearly distinguishes the two custom tools from CCAs that react to packet loss. However, BBR also shows significantly higher sending rates than those estimated when packet loss and RTT increase. As shown in Figure 7 in the Appendix, under most network conditions, Brutal and Hysteria have a sending rate similar to that of BBR (except when the loss rate reaches above 20%). Thus, a different approach is needed to differentiate CCAs like BBR that do not directly react to packet loss.

4.4 Active Throttling

To differentiate Brutal & Hysteria from TCP BBR, a potential censor could actively impose throttling mid-connection to induce BBR’s response to changes in bandwidth and contrast it with the behavior of the custom tools, which lack a similar adaptive mechanism. Specifically, in our experiments, we reduced the bandwidth by half (e.g., 200 Mbps → 100 Mbps, 50 Mbps → 25 Mbps) in the middle of each connection using a `token bucket filter` to simulate throttling. As illustrated in Figure 5, BBR is designed to respond to changing network conditions by adjusting its sending rate. Upon detecting reduced bandwidth, BBR begins probing for available capacity, eventually converging to the lowered rate after a few probing cycles. In contrast, Brutal and Hysteria interpret the reduced bandwidth as an increase in packet loss and thus attempt to compensate by increasing their sending rate even further.

5 Evaluation

We present a simple, threshold-based classifier to demonstrate the feasibility of differentiating the custom CCA among standard TCP and QUIC flows. The proof-of-concept classifier is then evaluated using a dataset of 10,080 traffic flows generated in controlled lab environments across varied network conditions.

5.1 Threshold-based Classifiers

The classification consists of two stages that are designed to distinguish the custom CCA from loss-based and rate-based CCAs respectively, as shown in Fig.6. In the first stage, we apply a simple threshold to determine if a connection exhibits a back-off response when encountering packet loss. Specifically, a connection is labeled as non-loss-based if its observed average sending rate surpasses the estimated performance by a scaling factor of $s = 5$. The expected performance is determined as $min(btl\_bw, s * mathis\_eqn)$, where btl_bw

[1]We use the BBR version included in Linux, which is version 1.

[2]We use a standard $MSS = 1460 bytes$

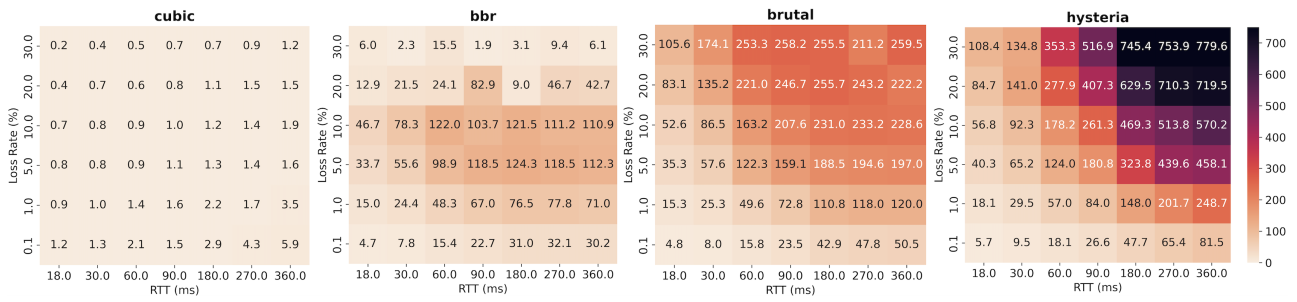

Figure 4: Differentiating Hysteria and Brutal across Varying Network Conditions. We quantify the average sending rate ratio of TCP-Cubic, TCP-BBR, Brutal, and Hysteria, respectively, with the estimated sendign rate using the Mathis equation.

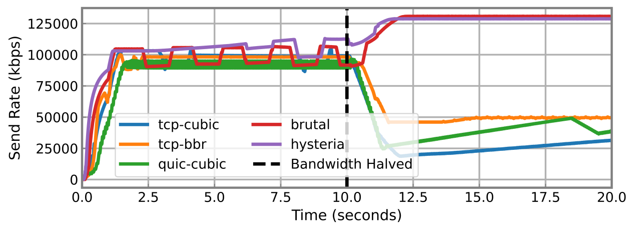

Figure 5: Differentiating Hysteria and Brutal with Active Throttling. The link has a bandwidth of 100Mbps, which is reduced to 50Mbps mid-connection, an RTT of 60ms, and no base loss rate.

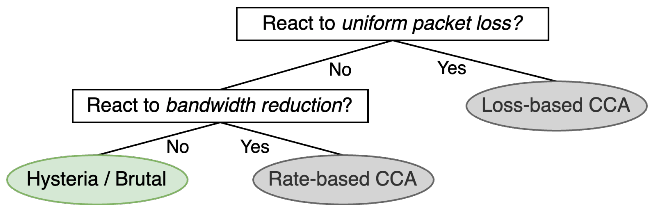
React to *uniform packet loss?*
No
Yes
Loss-based CCA
React to *bandwidth reduction*?
No
Yes
Hysteria / Brutal
Rate-based CCA

Figure 6: Classification Decision Tree. The two-stage classifier distinguishes Brutal from loss-based and rate-based CCAs based on reaction to packet loss and bandwidth reduction mid-connection

is the bottleneck bandwidth and *mathis_eqn* is derived from (1). Since the Mathis equation considers packet loss rate as the sole limiting factor, when a bottleneck bandwidth exists, loss-based CCAs are expected to operate below the minimum of these two constraints. By identifying flows that exceed this threshold, we isolate non-loss-based CCAs (including Hysteria and TCP-Brutal) from those CCAs that respond to packet loss using only passive measurement data.

The second stage further examines flows classified as non-loss-based to distinguish Brutal from rate-based CCAs like BBR that don't react to packet loss directly through back-off but rather conform to the new bandwidth in case of a bandwidth reduction. Here, we calculate the sending rate in 10ms windows to capture all peak transmissions and determine whether a flow continues to probe above the original bandwidth after the 7s mark, 2 seconds after a 50% bandwidth reduction is introduced to allow BBR to adjust its sending rate within a few bandwidth probing cycles, as described in § 4.4. Flows that persist in over-sending are flagged, while those conforming to the new reduced bandwidth are identified as rate-based CCAs.

Label/CCA	Brutal	Hysteria	BBR	CUBIC	YeAH	HTCP	Vegas	QUIC
loss-based	3	3	97	1260	1260	1260	1260	1260
non-loss	1257	1257	1163	0	0	0	0	0
rate-based	0	0	1147	0	0	0	0	0
Other(custom)	1257	1257	16	0	0	0	0	0

Table 1: Evaluation Results. Each tool was measured under 1260 different network condition combinations. The first stage labels flows as either loss-based or non-loss-based. The second stage then classifies non-loss-based flows further into rate-based or custom (e.g., Hysteria and TCP Brutal).

5.2 Results

The classifier was evaluated using traffic flows generated with our controlled testbed involving eight different CCA implementations. We configured various network conditions 20 different persistent RTTs, distributed evenly from 15ms to 300ms, and 21 different packet loss rates, ranging from 0.1% to 20%, under 3 different bandwidth capacity (200 Mbps, 100 Mbps, and 50 Mbps), totaling 10080 experiments. Each experiment lasts 10s, with a 50% reduction of available bandwidth at 5s mark. For Hysteria and Brutal, we configure them according to the protocol specifications to use the link bandwidth limit as the download bandwidth configuration [8].

Table 1 presents the classification results. The first stage achieved 100% accuracy in differentiating loss-based CCA flows from non-loss-based ones. Although 97 BBR flows were initially categorized as loss-based due to performance degradation under high packet loss – similar to loss-based CCAs – these did not contribute to false positives. In the second stage, the classifier correctly identified all traffic flows exhibiting custom congestion behaviors while producing only 16 false positives from BBR flows, mostly clustered in high-RTT, low-bandwidth scenarios.

6 Conclusion

Our study emphasizes that the custom congestion control implemented in circumvention proxies fundamentally conflicts with need for blending into mainstream traffic patterns to remain undetected. Through lab-based case studies on Hysteria and TCP-Brutal, we demonstrated how their aggressive sending behaviors can render such proxies identifiable and thus vulnerable to detection by censors. We argue that any performance optimizations for circumvention tools must align with standard protocol behaviors to maintain the efficacy of circumvention efforts.

References

[1] Alice, Bob, Carol, Jan Beznazwy, and Amir Houmansadr. 2020. How China Detects and Blocks Shadowsocks. In *Internet Measurement Conference*. ACM. https://censorbib.nymity.ch/pdf/Alice2020a.pdf

[2] Collin Anderson. 2013. Dimming the Internet: Detecting Throttling as a Mechanism of Censorship in Iran. arXiv:1306.4361 [cs.NI] https://arxiv.org/abs/1306.4361

[3] Kevin P. Dyer, Scott E. Coull, Thomas Ristenpart, and Thomas Shrimpton. 2013. Protocol Misidentification Made Easy with Format-Transforming Encryption. In *Computer and Communications Security*. ACM. https://eprint.iacr.org/2012/494.pdf

[4] Sergey Frolov and Eric Wustrow. 2019. The use of TLS in Censorship Circumvention. In *Network and Distributed System Security*. The Internet Society. https://tlsfingerprint.io/static/frolov2019.pdf

[5] Michelina Hanlon, Gerry Wan, Anna Ascheman, and Zakir Durumeric. 2024. Detecting VPN Traffic through Encapsulated TCP Behavior. In *Free and Open Communications on the Internet*. https://www.petsymposium.org/foci/2024/foci-2024-0016.pdf

[6] Bert Hubert. 2001. tc - show / manipulate traffic control settings. https://man7.org/linux/man-pages/man8/tc.8.html Accessed: 2024-11-04.

[7] Aperture Internet Laboratory. 2024. Hysteria: A High-Performance Proxy and Relay Tool. https://hysteria.network/ Accessed: 2024-11-04.

[8] Aperture Internet Laboratory. 2024. Hysteria: A High-Performance Proxy and Relay Tool. https://v2.hysteria.network/docs/advanced/Full-Server-Config/#congestion-control-details Accessed: 2024-11-04.

[9] Aperture Internet Laboratory. 2024. TCP Brutal: A High-Performance Congestion Control Algorithm. https://github.com/apernet/tcp-brutal Accessed: 2024-11-04.

[10] Matthew Mathis, Jeffrey Semke, Jamshid Mahdavi, and Teunis Ott. 1997. The macroscopic behavior of the TCP congestion avoidance algorithm. *SIGCOMM Comput. Commun. Rev.* 27, 3 (July 1997), 67–82. https://doi.org/10.1145/263932.264023

[11] Hooman Mohajeri Moghaddam, Baiyu Li, Mohammad Derakhshani, and Ian Goldberg. 2012. SkypeMorph: Protocol Obfuscation for Tor Bridges. In *Computer and Communications Security*. ACM. https://www.cypherpunks.ca/~iang/pubs/skypemorph-ccs.pdf

[12] Ram Sundara Raman, Prerana Shenoy, Katharina Kohls, and Roya Ensafi. 2020. Censored Planet: An Internet-wide, Longitudinal Censorship Observatory. In *Computer and Communications Security*. ACM. https://www.ramakrishnansr.com/assets/censoredplanet.pdf

[13] OONI Reports. 2023. Throttling of news media amid Kazakhstan's 2022 presidential election. https://ooni.org/post/2023-throttling-kz-elections/ Accessed: 2024-11-04.

[14] OONI Reports. 2024. Turkey: Throttling and DNS blocking of Twitter following deadly earthquake. https://ooni.org/post/2023-turkey-throttling-blocking-twitter/ Accessed: 2024-11-04.

[15] Z. Rosson, F. Anthonio, S. Cheng, C. Tackett, and A. Skok. 2022. Internet Shutdowns in 2022: The KeepItOn Report. https://www.accessnow.org/internet-shutdowns-2022/. Accessed: 2023-05-04.

[16] Zach Rosson, Felicia Anthonio, Sage Cheng, Carolyn Tackett, and Alexia Skok. 2022. Internet shutdowns in 2022: the KeepItOn Report — accessnow.org. https://www.accessnow.org/internet-shutdowns-2022/. [Accessed 04-May-2023].

[17] M. C. Tschantz, S. Afroz, Anonymous, and V. Paxson. 2016. SoK: Towards Grounding Censorship Circumvention in Empiricism. In *2016 IEEE Symposium on Security and Privacy (SP)*. https://doi.org/10.1109/SP.2016.59

[18] Zachary Weinberg, Jeffrey Wang, Vinod Yegneswaran, Linda Briesemeister, Steven Cheung, Frank Wang, and Dan Boneh. 2012. StegoTorus: A Camouflage Proxy for the Tor Anonymity System. In *Computer and Communications Security*. ACM. https://www.frankwang.org/files/papers/ccs2012.pdf

[19] 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 *USENIX Security Symposium*. USENIX. https://www.usenix.org/system/files/sec23fall-prepub-234-wu-mingshi.pdf

[20] Diwen Xue and Roya Ensafi. 2023. The Use of Push Notification in Censorship Circumvention. In *Free and Open Communications on the Internet*. Proceedings on Privacy Enhancing Technologies, Lausanne, Switzerland.

[21] 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)*. USENIX Association, PHILADELPHIA, PA.

[22] 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 *USENIX Security Symposium*. USENIX. https://www.usenix.org/system/files/sec22-xue-diwen.pdf

[23] Diwen Xue, Reethika Ramesh, Valdik S S, Leonid Evdokimov, Andrey Viktorov, Arham Jain, Eric Wustrow, Simone Basso, and Roya Ensafi. 2021. Throttling Twitter: an emerging censorship technique in Russia. In *Proceedings of the 21st ACM Internet Measurement Conference* (Virtual Event) *(IMC '21)*. Association for Computing Machinery, New York, NY, USA, 435–443. https://doi.org/10.1145/3487552.3487858

[24] Pengxiong Zhu, Keyu Man, Zhongjie Wang, Zhiyun Qian, Roya Ensafi, J. Alex Halderman, and Haixin Duan. 2020. Characterizing Transnational Internet Performance and the Great Bottleneck of China. In *Abstracts of the 2020 SIGMETRICS/Performance Joint International Conference on Measurement and Modeling of Computer Systems* (Boston, MA, USA) *(SIGMETRICS '20)*. Association for Computing Machinery, New York, NY, USA, 69–70. https://doi.org/10.1145/3393691.3394180

7 Appendix

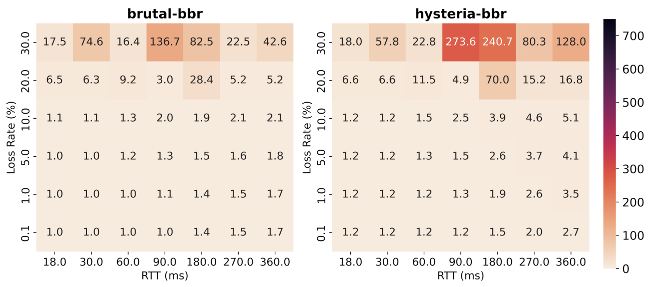

Figure 7: Average sending ratio comparison of Brutal and Hysteria with BBR. Both Hysteria and Brutal are similar to BBR for lower loss rates, whereas they significantly differ at higher loss rates ($> 20\%$).