HTTK(S): Characterizing and Evading Application-Layer Censorship by TransTeleKom

Aaron Ortwein
aortwein@umd.edu
University of Maryland, College Park

Abstract

Decentralized censorship systems prove challenging to study and defeat due to the diversity of censorship mechanisms. However, for countries with lesser government control over national network infrastructure, decentralized censorship is becoming more common. The most aggressive pursuer of decentralized control is Russia, whose surveillance and censorship regime has rapidly expanded within the last decade, and even more so in the midst of conflict with Ukraine. In this paper, we investigate the censorship system implemented by TransTeleCom (also TransTeleKom or TTK), one of Russia's largest ISPs. In particular, we focus on application-layer censorship of HTTP and HTTPS requests. We present insight on the characteristics of TransTeleCom's censors and show that it is possible to bypass them with a variety of packet-manipulation-based evasion strategies. Our investigation is a small step towards defeating Russian censorship at a nationwide scale.

1 Introduction

Many governments around the world censor Internet traffic in an effort to control the exchange of information online, restrict access to activities deemed unethical, or quash political or religious opposition and activism, amongst other reasons [12, 13]. Nation-state censorship models generally take one of two forms. In centralized censorship systems, the government exerts direct control over national networking infrastructure and the implementation of new censorship mechanisms. The alternative is a decentralized censorship model, in which the government legislates Internet restrictions to be implemented by individual ISPs however they see fit. As a result, censorship implementations are highly variable, and users are likely to experience different forms and degrees of blocking [13].

Decentralized censorship is becoming increasingly common, especially in Russia. The Russian surveillance and censorship regime has developed significantly since the 2012 inception of a national blocklist [13], and its activity has continued to surge in the midst of conflict with Ukraine [8], with over 8,000 additional websites blocked since the invasion on February 24, 2022 [10]. In addition to increased blocking by ISPs, the Russian government has recently made efforts to prevent censorship circumvention [4, 7, 9, 16] and move towards a closed Internet system (RuNet) entirely under Russian control [16].

The censorship situation in Russia poses multiple problems to users and censorship researchers. First and foremost, the Russian Constitution guarantees freedom of opinion and the rights to privacy and to freely search, receive, transmit, produce, and disseminate information. Authorities cite their motivations as wanting to protect the privacy of citizens, the Russian Internet, and national security [16], but surveillance and censorship violate citizens' guaranteed rights, as well as Internet security. Second, decentralized censorship systems are difficult to study and evade due to the diversity of censorship implementations by individual ISPs; understandings and evasion strategies may only apply locally as opposed to nationally.

In this work, we take a small step towards defeating Russian censorship. We conducted a series of experiments from a single vantage point primarily subject to censorship by Russian ISP TransTeleCom. We investigate the criteria for triggering censorship responses, localize middleboxes along routing paths to various servers, and test existing and attempt to derive new packet-manipulation-based evasion strategies. We find that TransTeleCom censors HTTP and HTTPS requests destined for a blacklisted IP address upon reading a forbidden domain in the Host header or SNI field. However, the rate at which censorship occurs depends on the requested domain, as requests to certain domains are not always routed to a middlebox capable of censoring them. Finally, we show that many client-side evasion strategies evolved by Geneva [3] – particularly TCB teardown strategies – are highly successful against the censors, and we present 3 new evasion strategies.

2 Background

2.1 Stateful Nation-State Censorship

While in-path ("man-in-the-middle") censorship is a powerful and effective analysis model because censors are hops in the routing path and can directly view, manipulate, or drop traffic, it is usually too computationally expensive to be a primary censorship method on a nationwide scale. As a result, nation-states tend to largely employ on-path ("man-on-the-side") censorship, in which censors monitor copies of packets for forbidden keywords or domains and inject packets with forged responses or to terminate connections [3]. Inspection of application-layer packet data requires that censors implement TCP to reconstruct data flow and maintain a

Transmission Control Block (TCB) containing information about every connection [2, 3, 15].

Censors have inherent limitations which make evasion possible. Because they do not implement TCP in the same manner as end hosts, censors will always be incapable of reconstructing the TCP stream as the hosts see it [15]. Furthermore, operating stateful censorship at scale requires deciding when to stop tracking a connection – commonly when a censor detects an RST or FIN packet and believes that the connection has ended – to avoid state exhaustion [2, 3].

2.2 Application-Layer Censorship

Application-layer filters which operate over HTTP or HTTPS traffic are becoming increasingly common for multiple reasons. First, DNS poisoning – in which a DNS resolver intentionally returns an incorrect answer, such as an IP address hosting a blockpage or a non-existent domain response (NXDOMAIN) – can be easily evaded by querying a trusted resolver; although on-path censors can still race resolvers to provide a response upon observing unencrypted DNS queries [5]. Second, the prevalence of CDNs, which host many websites behind a few IP addresses, means that IP blocking can cause significant collateral damage. To prevent overblocking, censors require a more granular blocking policy, thus necessitating that they are capable of inspecting request headers and potentially payloads [14].

The Host header is a mandatory HTTP header created to signal the domain to which the client wishes to connect to web servers that host multiple websites. Because HTTP is unencrypted, reading and censoring based upon packet contents is trivial. Despite the encryption of HTTPS requests such that the Host header cannot be read, the Server Name Indication (SNI) extension to TLS, which serves the same purpose as the HTTP Host header, is sent in plaintext during the Client Hello [5].

2.3 Russian Censorship Model

Russia's decentralized censorship regime has grown quickly in the last decade. Since 2012, in implementation of federal law 139-FZ, the primary entity in charge of Russian Internet censorship policy – the Federal Service for Supervision of Communications, Information Technology and Mass Media, abbreviated as Roskomnadzor – has maintained a Registry of Banned Sites consisting of domains, IP addresses, and subnets that are required to be blocked by law. However, the Russian government does not maintain the same level of control over national network infrastructure as centralized regimes such as China or Iran, so the implementation of blocking falls to individual ISPs. Consequently, censorship techniques vary between autonomous systems and even their constituent networks, as do the accessibility of web resources and the success rates of circumvention tools [13]. The highly-fractured state of the Russian Internet may cause even geographically close users to have vastly different experiences online, and it further complicates censorship research and evasion because understandings of and evasion strategies for censorship systems do not uniformly apply across the country.

In recent years, the Russian government has sought to further extend its control over and isolation from the global Internet. While not directly responsible for carrying out censorship, it developed deep packet inspection (DPI) technology called System of Operative Search Measures (SORM) which ISPs run for both surveillance and censorship purposes [13]. The 2019 "sovereign Internet" law requires ISPs to install DPI technology that would grant Roskomnadzor the capabilities to block banned sites and reroute Internet traffic themselves. The same law required the creation and use of a national DNS effective January 1, 2021 [16]. The full realization of this law would allow for centralized control of Russian networks. In addition to ISPs, Russian authorities have attempted to compel other Internet services to cooperate with their censorship efforts. A 2018 law introduces fines for search engines that provide users instructions for circumventing censorship or access to proxy services to successfully do so. In 2019, Roskomnadzor mandated that VPNs, anonymizers, and search engines block access to sites included on the Registry of Banned Sites [9, 16].

In 2020, Russia moved to take more measures against censorship circumvention efforts, proposing an amendment to ban secure protocols such as TLS 1.3, DNS over HTTPS (DoH), DNS over TLS (DoT), and ESNI that hide the server name indication inside HTTPS traffic [4]. Because these protocols have not yet seen widespread adoption, outright blocking of these protocols would result in only minor collateral damage.

The methods by which Russia has censored forbidden content has been the subject of previous study by academic and activist research. A collaborative report by OONI, OutRight Action International, and CitizenLab [11] found that using HTTP transparent proxies and DNS hijacking to serve blockpages were the most common censorship responses to requests for LGBTIQ-related websites. In their case study of Russia [13], Censored Planet finds that residential probes mostly encountered blockpages, while VPSes in data centers were primarily subject to IP blocking. In December 2021, Russian ISPs began to block access to the Tor network via IP blocking and to the Tor Project website by means of serving a blockpage or interfering with the TLS handshake [7]. Following Russia's invasion of Ukraine on February 24, 2022, OONI reported a variety of censorship methods for implementing new blocks, including DNS manipulation, blockpages, and interference with the TLS handshake, and found that the most common censorship mechanism was an injected RST packet following the TLS Client Hello [8]. Interestingly, Russia has shown evidence of centralized, targeted throttling of Twitter for censorship purposes on multiple occasions, further indicating their growing censorship capabilities [8, 17].

2.4 Client-Side Packet-Manipulation-Based Evasion

Client-side packet-manipulation-based censorship evasion strategies require sending specially-crafted packets from the client that manipulate censor state or otherwise render the censor unable to locate forbidden domains or keywords [2, 6]. One such example is sending insertion packets – packets that induce processing by the censor but not the server [6] – such as TTL-limited RST packets which force a TCB teardown at the censor and expire before reaching the server.

Historically, the task of censorship evasion has required an understanding of how a target censor works before manually developing evasion strategies. Geneva [3] is a genetic algorithm that can run from the client and randomly manipulate outbound packets – biased towards favorable behaviors – in order to independently derive packet manipulation strategies. Geneva has multiple advantages compared to the traditional process of manually devising new censorship evasion strategies. Not only does Geneva exploit the inherent limitations of nation-state censors, but it can also find unintuitive strategies and bugs in censor implementations. Additionally, Geneva significantly reduces the amount of time needed to devise new censorship evasion strategies and shifts the issue of understanding a target censor to after it has been evaded.

3 Methodology

In order to investigate Russian censorship mechanisms, we acquired a single vantage point physically located in Moscow and part of the address space of AS50867. We then conducted a series of experiments with the objective of answering two questions: (1) How does the censorship system to which our vantage point is subject work? and (2) Can we bypass this censorship system? Given Russia's decentralized censorship model, it is important to note that we cannot necessarily generalize our findings to other networks or autonomous systems.

3.1 Censorship Type Testing

Detecting censorship requires testing websites likely to elicit responses from censors. Top10VPN has published a list of domains that have been added to the Registry of Banned Sites by Roskomnadzor since Russia's invasion of Ukraine on February 24th, 2022 and are related to the conflict [10]. At the time of download, this list contained 931 domains. We used curl to send an HTTP GET request to each of these domains and observed the response headers and body. We repeated this process with HTTPS by prepending "https://" to each domain. For each domain which elicited an unsuccessful or anomalous response, we compared the result of a GET request from a known uncensored vantage point in Japan.

To detect DNS censorship, we used OONI Probe, which performs several web connectivity tests on websites from Citizen Lab test lists. These web connectivity tests include whether DNS lookups from within a control network and the client's network yield different responses, whether a TCP session can be established, and whether the results of HTTP(S) GET requests from the control and client's network differ. The accuracy of the test results was verified by first resolving each domain potentially subject to DNS manipulation from the Russian vantage point, and if the domain successfully resolved, issuing a GET request to the returned IP address (with the Host header or SNI field set appropriately) from our uncensored vantage point and checking for a successful or redirection response. If DNS resolution from the Russian client resulted in NXDOMAIN, we confirmed that this was also the case from our uncensored vantage.

3.2 Censorship Reliability Testing

We observe that censorship is inconsistent, as successive runs of our scripts to detect HTTP and HTTPS censorship yielded non-identical responses for the same domains; some requests would receive responses from the censor and others would receive responses from the server. Censor unreliability raises two additional questions: (1) What is the baseline rate at which forbidden resources are censored? and (2) Does the consistency of censorship responses depend on the requested domain? Knowledge of censor reliability is especially important for determining whether a potential evasion strategy is legitimately successful or if the censor simply failed to block access to the requested resource. We therefore take the same list of domains used for detecting HTTP and HTTPS censorship and issue 1,000 HTTP requests and 1,000 HTTPS requests to each, recording the count of censorship responses.

3.3 Censorship Triggers

Host Header and SNI Inspection To test for filtering based on the Host header or SNI field, we send web requests satisfying each of the following conditions:

- The Host header or SNI field is set to an inconsistently-censored domain (ex. twitter.com), and the destination host is innocuous (ex. google.com).
- The Host header or SNI field is set to a consistently-censored domain (ex. ukrmilitary.com), and the destination host is innocuous.
- The Host header or SNI field is set to an inconsistently-censored domain, and the destination host is consistently censored.
- The Host header or SNI field is set to a consistently-censored domain, and the destination host is inconsistently censored.

Bidirectionality Another important question is whether censorship is bidirectional – that is, whether a censorship response can be triggered by both inbound and outbound traffic to and from the target country. In particular, we sought to determine whether censors inspect the Host Header – a mandatory header indicative of the domain to which the client wishes to connect – of all passing traffic in order to make blocking decisions. We ran HTTP servers on both the Russian vantage point and the uncensored vantage point in Japan, and from each machine, we issued GET requests to the other, with the Host header set to a reliably censored domain (ukrmilitary.com).

3.4 Client-Side Evasion Strategy Testing

We use Geneva to test existing and develop new packet-manipulation-based evasion strategies. We first ran Geneva from the Russian vantage point, manipulating outbound HTTP requests according to client-side evasion strategies reproduced or found by Bock et al. in the original Geneva work [3]. For each evasion strategy, we sent 1,000 HTTP requests to a domain measured to have a 100% censorship rate and recorded the number of times we received a response from the real server. Additionally, we allowed Geneva to train on the censor to derive new evasion strategies. We configured Geneva to evolve with a population pool of 100 individuals over the course of 30 generations (or until population convergence occurred). After finding successful strategies with Geneva, we manually analyzed each strategy to better understand censor functionality.

3.5 Localizing Censors

In order to obtain a sense of where along the routing paths censors are located, we utilize the TraceVis middlebox-discovery tool, which allows for sending arbitrary TTL-limited packets with increasing TTL values until they reach either the server or a censor. In sending these packets, we consider three cases of censorship:

- The requested domain has a 100% censorship rate and is always censored at the same hop.
- The requested domain has a 100% censorship rate but the hop at which it is censored varies across requests.
- The requested domain has between a 0% and 100% (both exclusive) censorship rate.

If a website is always censored at the *n*th hop, every possible routing path must lead to a censor at the *n*th hop, and so we assume that all possible *n*th hops from the Russian vantage point along the path to the domain are in-path or are near on-path censors. For a website that is always censored in the range of the *m*th to *n*th hops, every possible routing path must still lead to a censor, but not every possible *m*th through *n*th hop will necessarily be or have a censor. We begin with the assumption that every machine between *m* and *n* hops from the Russian vantage point is or is adjacent to a censor, and later eliminate machines for which this assumption does

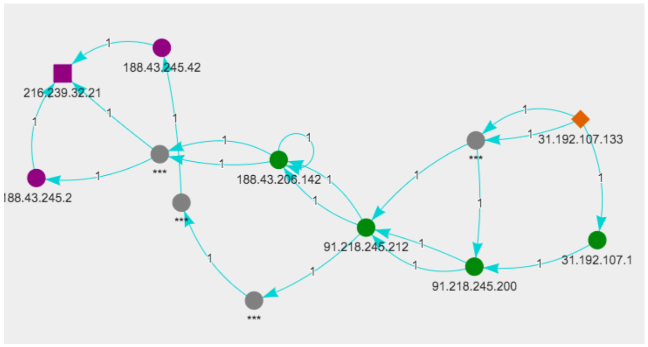


Figure 1. Routing paths of censored requests to ukrmilitary.com (represented by the purple square) from our Russian vantage point (represented by the orange diamond). The hop indicated as being the server is actually the responding censor.

not make sense (such as those belonging to non-Russian ISPs). For domains that inconsistently trigger censorship, we can only localize the closest censors, as the TTL is not guaranteed to expire at a censor. It is possible that when the TTL exceeds the distance of a censor, one packet misses the censor, while the next iteration takes a path through the censor and triggers a response. In this case, the response appears to come from the hop at which the TTL expires, when in reality, it was produced by an earlier hop. However, we can ascertain that if a packet travels through an IP address at the earliest censor hop without being censored, then that machine is neither actually or adjacent to a censor.

We then directed TTL-limited HTTP or HTTPS GET requests to domains fulfilling each of the aforementioned cases of censorship, recording the hop number(s) at which requests were censored. We subsequently sent TTL-limited requests with packets manipulated according to successful evasion strategies we discovered. Figures 1 and 2 show the difference between the routing path of censored requests and that of requests which reached the server. With the knowledge of how many hops away censors are located, we could identify the specific IP addresses along the path to the server that are close to or are the actual censor. In addition to localizing censors, we sought to determine whether inconsistency of censorship responses is the result of unreliable censors or a lack of censors along all possible routing paths.

4 Results

4.1 Types of Censorship

HTTP Censorship We observe that the typical censorship response to forbidden HTTP requests is redirection to a blockpage via the Location header of a 301 Moved Permanently HTTP response. For any censored website, the

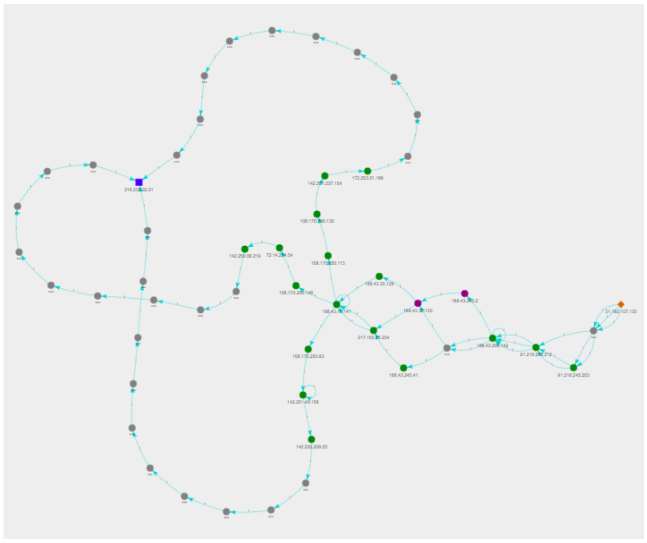
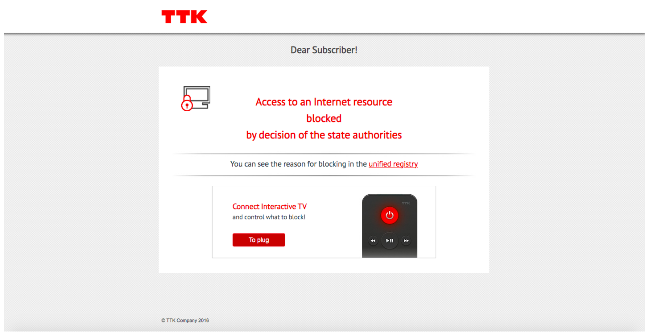


Figure 2. True routing paths of requests to the real ukrmilitary.com (represented by the blue square) from our Russian vantage point (represented by the orange diamond).



TTK

Dear Subscriber!

Access to an Internet resource
blocked
by decision of the state authorities

You can see the reason for blocking in the unified registry

Connect Interactive TV
and control what to block!

To plug

© TTK Company 2016

Figure 3. Blockpage served upon sending a forbidden HTTP request.

blockpage is identical and always hosted at fz139.ttk.ru. Figure 3 shows the blockpage. However, the URL contains dynamic query parameters, including ones indicative of the date of the forbidden request and the law banning the requested resource. Additionally, censorship response packets are returned with FIN+PSH+ACK flags set, whereas server responses only set the PSH+ACK flags.

Of the 931 domains from the Top10VPN list, we recorded at least one blockpage response for only 113 of them, despite all of them being entered into the Registry of Banned Sites. We found only one domain – ukrmilitary.com – which was censored on every request. With the exception of viptrade.global, which triggered a blockpage in 99.7% of measurements (at time of writing, viptrade.global no longer triggers censorship), no domain triggered censorship with greater than 36% frequency. The average and median rates of censorship

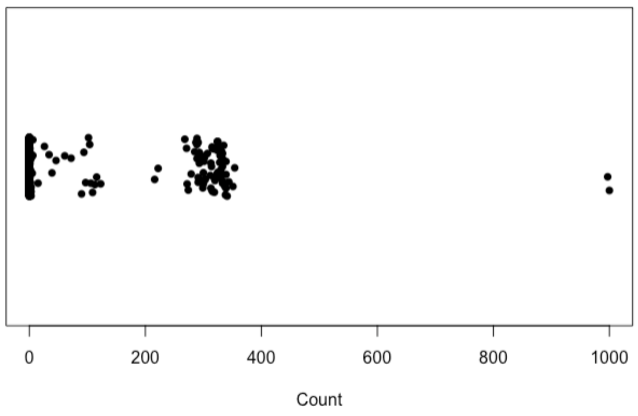


Figure 4. Distribution of the rates (out of 1000 requests) at which HTTP requests to domains on the Top10VPN list are censored.

across these domains were 23.5% and 30%, respectively. Figure 4 depicts the distribution of censorship rates amongst the tested HTTP websites. Overall, the HTTP censorship experienced by our vantage point is rather ineffective, and no evasion strategy beyond simply re-requesting a website seems to work with at least moderate success (> 60%) for the vast majority of domains.

HTTPS Censorship The typical observed censorship response to forbidden HTTPS requests is an injected RST+ACK or FIN+ACK packet returned to the client just after the Client Hello message is sent. Compared to HTTP censorship, HTTPS censorship is much more effective, though still inconsistent. Of the 931 tested domains, 187 exhibited evidence of HTTPS censorship at least once. While the median rate of censorship across only censored domains was only 33.4%, the average rate was 52.1%, skewed by 77 domains that were blocked at least 90% of the time, of which 46 were blocked 100% of the time. Figure 5 depicts the distribution of censorship rates amongst the tested HTTPS websites. More domains are more consistently blocked over HTTPS, which poses a greater threat to the availability of Internet content as websites increasingly automatically redirect users to HTTPS for security.

DNS Censorship We do not observe any evidence of DNS manipulation. Every website that OONI Probe designated as potentially being subject to DNS-based censorship either legitimately resulted in an NXDOMAIN message or resolved to a correct IP address. Differences in IP addresses provided by the DNS resolvers used by the control and Russian machines can likely be attributed to the resolver returning geographically close IP addresses to improve connection speed.

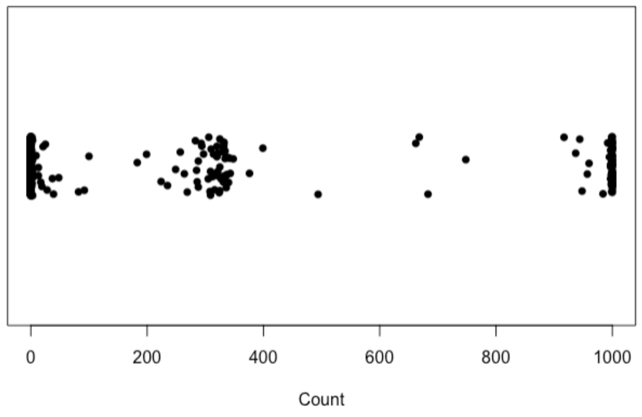


Figure 5. Distribution of the rates (out of 1000 requests) at which HTTPS requests to domains on the Top10VPN list are censored.

4.2 Censorship Triggers

TransTeleCom appears to censor HTTP and HTTPS requests based on the Host header and SNI field, respectively, conditional on the destination host also being blacklisted. When requests are directed to or have the Host header or SNI field set to an innocuous website such as google.com, censorship is not (generally) triggered. We did observe an unusual case that specifically when the destination host of an HTTPS request is ukrmilitary.com, an RST+ACK response is always injected, regardless of the hostname specified by the SNI field. When both the destination host and the Host header or SNI field are forbidden, we find that the censorship response is triggered by the Host header or SNI field (based on the blockpage query parameters or SSL error message), but requests are censored at the rate of the destination host. These results indicate that the censorship system is limited to select IP ranges, though we did not attempt to find the specific IP ranges which trigger censorship.

Our experiment to test whether censorship is bidirectional was performed with two uncensored destination IP addresses. Given the above findings, we are unable to confirm whether censors operate on both inbound and outbound traffic. Unfortunately, we do not control any censored IP addresses, nor did we have access to tools that would allow us to see return traffic had we spoofed a censored IP address. It is possible that censors do work bidirectionally when both the destination IP address and Host header are forbidden.

4.3 Client-Side Evasion Strategies

Testing Existing Strategies We tested all strategies in [3], finding that every TCB Teardown strategy – except for the second Corrupt Ack variant of the With RST+ACK subspecies, which always resulted in an RST from the server – worked with guaranteed success, as did the Hybrid strategy. TCB Desynchronization strategies yielded mixed results. Of the Inc. Dataofs subspecies, only the Corrupt Checksum, Small TTL, and Corrupt Ack variants circumvent censorship and result in valid server responses with 100% success. The Invalid Flags and Corrupt WScale variants both avoid filtering, but respectively result in an RST packet or a 400 Bad Request error from the server. All variants of the Invalid Payload and Simple subspecies evade censorship with approximately a 75% success rate. The Stutter Request subspecies is completely ineffective. Similarly, none of the Segmentation strategies worked even once, nor did TCB Turnaround and Invalid Options. The standout success of TCB Teardown strategies indicates that censors seem most susceptible to TCB manipulation with insertion packets. Indeed, manual analysis of TCB Teardown strategies revealed that TCB Teardown can be induced by sending an insertion packet with any arbitrary combination of TCP flags including RST or FIN.

Deriving New Strategies We now present 3 strategies, all of which have a 100% success rate for both HTTP and HTTPS, derived from training Geneva against TransTeleCom's censors.

Strategy 1
[TCP:flags:S]-tamper{TCP:dataofs:replace:5}-|

Strategy 1 triggers only on SYN packets, modifying the Data Offset field of the TCP header to 5. Importantly, the TCP header has variable length when it includes options, but its minimum length is 5 bytes otherwise. Setting the TCP header to its minimum length (or any longer length that is still smaller than the real header size) when it does have options shifts the excess data into the packet payload. SYN packets with a payload are permitted by RFC 793 [1], as they are required to implement TCP Fast Open; however, TCP Fast Open is uncommon in practice, so censors likely ignore these packets along with the rest of the connection.

Strategy 2
[TCP:flags:PA]-fragment{tcp:-1:False:15}-|

Strategy 2 segments PSH+ACK packets into two pieces with some amount of overlap before sending them out of order. The strategy found by Geneva splits the packet in half and creates 15 bytes of overlap between the two segments, but post-hoc manual analysis revealed that segmenting the packet at any non-zero index (segmenting at index 0 would effectively perform no segmenting at all) with any non-zero overlap maintains the success of this strategy. Returning packets out of order is necessary, as this strategy is rendered completely ineffective otherwise. The success of this strategy indicates that TransTeleCom censors are incapable of reassembling reversed and overlapping packet segments.

Strategy 3
[TCP:flags:A]-duplicate(,tamper{TCP:flags:replace:S})-|

Strategy 3 duplicates ACK packets, replacing the flags of the copy with SYN. We find that setting any arbitrary combination of flags including SYN but excluding RST is still successful. We hypothesize that this strategy may desynchronize the censor from the connection. The first packet will complete the TCP three-way handshake with the server. After the initial SYN packet, no further messages sent by the client in a typical TCP connection are expected to contain a SYN flag. As a result, the presence of a SYN flag in subsequent packets may cause the censor to believe that a new TCP connection is being initiated and create a new TCB, thus ignoring the rest of the current connection. To test this hypothesis, we use Geneva to drop outbound ACK packets such that the client only sends a SYN packet before a forbidden request. Despite not completing the three-way handshake prior to the request, we still observe a censorship response, suggesting that censors instantiate a TCB upon partial observation of the three-way handshake.

4.4 Localizing Censors

The IP addresses we deemed to likely be closest to those or directly responsible for the censorship of various domains are noted in Table 1. WHOIS lookups of these IP addresses indicated that all of them are located in AS20485 Joint Stock Company TransTeleCom, which is an upstream autonomous system of AS50867. Lookups of further hops reveal that TransTeleCom's censors are located on the edge of Russia's network borders.

Inconsistent censorship responses seem to be a result of load balancing and the distribution of censorship responsibilities amongst censors. For the domains in Table 1 that were not censored 100% of the time, there are non-censor IP addresses at the same hop that censorship occurs – TTL-limited packets traversed them prior to being censored or reaching the server –, indicating that there exist routing paths that simply miss the censor. Given that the TransTeleCom IP addresses seemingly associated with censorship differ across domains, it is possible that various censors have different blocklists (with some overlap). Interestingly, for the HTTP version of kherson-news.info, packets often travel through two machines indicated as being associated with censorship of the HTTPS version without being censored. It may be the case that these machines are not configured to monitor HTTP traffic.

5 Conclusion

In this paper, we present an understanding of and evasion strategies for TransTeleCom's censorship system in AS20485. We find that TransTeleCom primarily censors HTTP and HTTPS requests by inspecting the Host header or SNI field of requests destined for a blacklisted IP address, which should be explored more thoroughly in future work. Censorship of requests over both protocols are largely inconsistent – a result of load balancing and the distribution of censorship responsibilities amongst middleboxes – and can often be evaded with no strategy besides re-requesting resources. However, we do find several highly-successful evasion strategies whose success is independent of the requested resource. These strategies primarily exploit censors' inability to handle SYN packet payloads, limited packet reassembly capabilities, or vulnerability to TCB manipulation by connection-starting or -ending flags in insertion packets.

Importantly, these censors are not located in the same autonomous system as our vantage point. Therefore, it is possible that these results extend to autonomous systems with AS20485 as an upstream autonomous system. For instance, we encounter the same blockpage as AS21127 [7]. However, even so, the decentralized nature of Russia's censorship infrastructure means that our results do not generalize to the entire country. Defeating censorship in Russia will require a greater understanding of the censorship systems used by all Russian ISPs. There remains much future work in acquiring a diverse range of vantage points throughout Russia, characterizing and measuring the censorship systems to which they are subject, testing existing and deriving new evasion strategies, and determining which of these strategies provide the largest coverage of censorship systems.

References

[1] 1981. Transmission Control Protocol. RFC 793. https://doi.org/10.17487/RFC0793

[2] Kevin Bock, George Hughey, Louis-Henri Merino, Tania Arya, Daniel Liscinsky, Regina Pogosian, and Dave Levin. 2020. Come as You Are: Helping Unmodified Clients Bypass Censorship with Server-Side Evasion. In *Proceedings of the Annual Conference of the ACM Special Interest Group on Data Communication on the Applications, Technologies, Architectures, and Protocols for Computer Communication* (Virtual Event, USA) *(SIGCOMM '20)*. Association for Computing Machinery, New York, NY, USA, 586–598. https://doi.org/10.1145/3387514.3405889

[3] Kevin Bock, George Hughey, Xiao Qiang, and Dave Levin. 2019. Geneva: Evolving Censorship Evasion Strategies. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security* (London, United Kingdom) *(CCS '19)*. Association for Computing Machinery, New York, NY, USA, 2199–2214. https://doi.org/10.1145/3319535.3363189

[4] Catalin Cimpanu. 2020. *Russia wants to ban the use of secure protocols such as TLS 1.3, DoH, DoT, ESNI.* https://www.zdnet.com/article/russia-wants-to-ban-the-use-of-secure-protocols-such-as-tls-1-3-doh-dot-esni/#:~:text=Log%20Out-,Russia%20wants%20to%20ban%

20the%20use%20of%20secure%20protocols%20such,fully%20hide%20the%20traffic's%20destination.&text=Catalin%20Cimpanu%20was%20a%20security,Sep%202018%20and%20Feb%202021.

[5] Lin Jin, Shuai Hao, Haining Wang, and Chase Cotton. 2021. Understanding the impact of encrypted DNS on internet censorship. In *Proceedings of the Web Conference 2021* (Ljubljana Slovenia). ACM, New York, NY, USA.

[6] Sheharbano Khattak, Mobin Javed, Philip D. Anderson, and Vern Paxson. 2013. Towards Illuminating a Censorship Monitor's Model to Facilitate Evasion. In *3rd USENIX Workshop on Free and Open Communications on the Internet (FOCI 13)*. USENIX Association, Washington, D.C. https://www.usenix.org/conference/foci13/workshop-program/presentation/khattak

[7] Arturo Filastò Maria Xynou. 2021. *Russia started blocking Tor.* https://ooni.org/post/2021-russia-blocks-tor

[8] Arturo Filastò Maria Xynou. 2022. *New blocks emerge in Russia amid war in Ukraine: An OONI network measurement analysis.* https://ooni.org/post/2022-russia-blocks-amid-ru-ua-conflict/

[9] David Meyer. 2019. *VPN providers pull Russian servers as Putin's ban threatens to bite.* https://www.zdnet.com/article/vpn-providers-pull-russian-servers-as-putins-ban-threatens-to-bite/

[10] Simon Migliano. 2022. *Websites Blocked in Russia Since Ukraine Invasion.* https://www.top10vpn.com/research/websites-blocked-in-russia/

[11] The Citizen Lab Open Observatory of Network Interference (OONI), OutRight Action International. 2021. *No Access: LGBTIQ Website Censorship in Six Countries.* https://ooni.org/post/2021-no-access-lgbtiq-website-censorship-six-countries/

[12] Ram Raman, Adrian Stoll, Jakub Dalek, Reethika Ramesh, Will Scott, and Roya Ensafi. 2020. Measuring the Deployment of Network Censorship Filters at Global Scale. https://doi.org/10.14722/ndss.2020.23099

[13] Reethika Ramesh, Ram Sundara Raman, Matthew Bernhard, Victor Ongkowijaya, Leonid Evdokimov, Anne Edmundson, Steven Sprecher, Muhammad Ikram, and Roya Ensafi. 2020. Decentralized control: A case study of Russia. In *Proceedings 2020 Network and Distributed System Security Symposium* (San Diego, CA). Internet Society, Reston, VA.

[14] Benjamin VanderSloot, Allison McDonald, Will Scott, J. Alex Halderman, and Roya Ensafi. 2018. Quack: Scalable Remote Measurement of Application-Layer Censorship. In *27th USENIX Security Symposium (USENIX Security 18)*. USENIX Association, Baltimore, MD, 187–202. https://www.usenix.org/conference/usenixsecurity18/presentation/vandersloot

[15] 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 *Proceedings of the 2017 Internet Measurement Conference* (London, United Kingdom) *(IMC '17)*. Association for Computing Machinery, New York, NY, USA, 114–127. https://doi.org/10.1145/3131365.3131374

[16] Human Rights Watch. 2020. *Russia: Growing Internet Isolation, Control, Censorship.* https://www.hrw.org/news/2020/06/18/russia-growing-internet-isolation-control-censorship

[17] 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

Table 1. TTK and Non-Censor IP Addresses at Same Hop as Censorship

Website	Censorship Rate	Censored Hop	TTK IP Addresses	Non-Censor Nth Hop IP Addresses
ukrmilitary.com	100%	6, 7	188.43.30.129 188.43.245.41 217.150.55.234	N/A
https://kherson-news.info	100%	6	188.43.30.129 188.43.245.41 172.31.3.12 172.31.3.16	N/A
kherson-news.info	33.3%	6	188.43.30.129	172.31.3.12 172.31.3.16
imag24.net	33.7%	6	188.43.30.129	62.115.139.173