TSPU: Russia's Decentralized Censorship System

Diwen Xue
University of Michigan

Benjamin Mixon-Baca
ASU/Breakpointing Bad

ValdikSS
Independent

Anna Ablove
University of Michigan

Beau Kujath
ASU/Breakpointing Bad

Jedidiah R. Crandall
ASU/Breakpointing Bad

Roya Ensafi
University of Michigan

ABSTRACT

Russia's Sovereign RuNet was designed to build a Russian national firewall. Previous anecdotes and isolated events in the past two years reflected centrally coordinated censorship behaviors across multiple ISPs, suggesting the deployment of "special equipment" in networks, colloquially known as "TSPU". Despite the TSPU comprising a critical part of the technical stack of RuNet, very little is known about its design, its capabilities, or the extent of its deployment.

In this paper, we develop novel techniques and run in-country and remote measurements to discover the *how, what,* and *where* of TSPU's interference with users' Internet traffic. We identify different types of blocking mechanisms triggered by SNI, IP, and QUIC, and we find the TSPU to be in-path and stateful, and possesses unique state-management characteristics. Using fragmentation behaviors as fingerprints, we identify over one million endpoints in Russia from 650 ASes that are behind TSPU devices and find that 70% of them are at most two hops away from the end IP. Considering that TSPU devices progressed from ideation to deployment in three years, we fear that the emerging TSPU architecture may become a blueprint for other countries with similar network topology.

CCS CONCEPTS

• **General and reference** → **Measurement**; • **Security and privacy** → **Firewalls**; **Web protocol security**; • **Social and professional topics** → **Censorship**.

KEYWORDS

Censorship, Interception, Measurement, Russia

ACM Reference Format:
Diwen Xue, Benjamin Mixon-Baca, ValdikSS, Anna Ablove, Beau Kujath, Jedidiah R. Crandall, and Roya Ensafi. 2022. TSPU: Russia's Decentralized Censorship System. In *ACM Internet Measurement Conference (IMC '22), October 25–27, 2022, Nice, France.* ACM, New York, NY, USA, 16 pages. https://doi.org/10.1145/3517745.3561461

IMC '22, October 25–27, 2022, Nice, France

ACM ISBN 978-1-4503-9259-4/22/10.
https://doi.org/10.1145/3517745.3561461

1 INTRODUCTION

Since 2012, the Russian government has been developing both legal and technical frameworks to construct its censorship apparatus [17]. In May 2019, the "Sovereign RuNet" law was signed, requiring telecom operators to install a home-grown DPI system, colloquially known as "TSPU", on their networks free of charge [20]. This provides the government with an extraordinary ability to centrally and unilaterally control the traffic passing through thousands of privately-owned, distributed ISPs. This centralized control was established to isolate Russia's internal Internet ecosystem from the rest of the world to "protect" Russia in the face of foreign threats [27].

Previous studies independently point to the deployment of the TSPU. In March 2021, Russia pressured Twitter to comply with its content removal requests with targeted throttling and threats of outright blocking [29]. Xue *et al.* showed that throttling behaviors demonstrated a high degree of uniformity and coordination across a range of ISPs [98]. Subsequently, Roskomnadzor, Russia's communication agency, publicly confirmed that the TSPU, which comprises the technical stack of RuNet, was used for throttling [28]. In March 2022, censorship observatory OONI reported that many news and social media sites promoting narratives critical of the Russian war effort were suddenly blocked [72]. In particular, their collected censorship data showed temporal uniformity across ISPs in Russia in "some sort of centralized way", as opposed to relying on each ISP to implement censorship independently as previous work observed [81].

TSPU (технические средства противодействия угрозам) "Technical Measures to Combat Threats", refers to the homegrown DPIs that are developed and distributed directly by Roskomnadzor, who administers censorship policies, governing what resources to block and how and when to block them. Though the TSPU has been in operation for years, very little is known about its design, capability, or the extent of its deployment.

This is the first study dedicated to analysis of the TSPU including discovering *How* the TSPU blocks a connection, *What* resources it blocks, and *Where* it is installed with respect to Russian users. Answering such questions is challenging: we need local Russian vantage points in networks where TSPU devices are deployed. More, distinguishing TSPU behaviors from other DPI is non-trivial, as ISPs also deploy their own commodity DPIs that censor many of the same resources. Finally, due to the asymmetricity of TSPU blocking, existing remote measurement platforms, *e.g.*, CensoredPlanet [84], are not suitable for observing or tracking TSPU behaviors.

To answer *how* the TSPU blocks a connection and *what* resources it blocks, first we devise a technique to isolate and identify TSPU behaviors based on its uniformity across ISPs. From our vantage points in Russia, we generate ad-hoc measurements to our control servers with the goal of triggering the TSPU. We find three types of triggers—SNI-based, IP-based, and QUIC—that result in six different blocking behaviors we attribute to the TSPU. We find that TSPU devices are in-path, block only connections that originate from inside Russia, and exhibit a degree of statefulness. For the IP layer, it maintains state to buffer and forward IP fragments in a unique way. For the TCP layer, we observe that different sequences of TCP flags may put a connection into different states, some of which exempt the connection from being blocked even with a trigger. Based on these observations, we devise several circumvention strategies, some of which can be deployed solely on the server side without modification to a client's network stack. However, we note that some of these strategies may be patchable assuming TSPU devices are well-provisioned with computational resources.

To answer *where* the TSPU devices are installed with respect to Russian users, we first use a TTL-based technique from our in-country vantage points and find that they are close to end users. We also find evidence suggesting multiple TSPU installations on the same network path, some of which have only partial (upstream-only) visibility into user's traffic. To scale up measurements beyond vantage points that we own, we design novel remote measurement techniques based on how the TSPU assigns "client" and "server" roles and how the TSPU handles IP fragmentation. In particular, we use TSPU's fragmentation queue limit as its fingerprint to probe TSPU devices from outside the country in a scalable, ethical way. Out of four million endpoints in Russia, we identified over a million of them from 650 ASes as behind at least one TSPU device. For endpoints behind a TSPU device, we measure the exact network hop of the device's location, accompanied by traceroutes, to identify the network links that host TSPU devices. Our results suggest an architecture where the DPI devices are deployed closer to network leaves (possibly end users) than to border or backbone networks.

Our study reveals pervasive deployment of TSPU devices close to end users that empowers the Russian government to achieve fine-grained control over thousands of privately-owned, distributed ISPs. Using this architecture, the Russian government can easily and effectively escalate its control over the free flow of information. For example, in the midst of its conflict with Ukraine, by limiting access to facts about the war, Russia has created a propaganda bubble for its citizens [31, 59, 61, 86]. In contrast to the Great Firewall of China (GFW) that took decades to build and deploy at choke points in the nation's Internet topology, in less than three years Russia achieved building a nation-scale censorship architecture deployed in decentralized networks. In addition, by being in-path and close to end users, the TSPU is much better suited to perform potential targeted surveillance and machine-in-the-middle attacks. As Russia has been exporting its censorship techniques to other countries [91], we warn that this emerging TSPU architecture may become a blueprint for other countries seeking to exercise greater information control over their entire Internet. We hope our findings serve as a wake-up call to our community and encourage future work on Russia's new censorship model.

2 BACKGROUND

Anecdotes and news reporting collectively suggest that governments around the world are increasingly seeking information control that interferes with the free flow of information [12]. In response, censorship researchers have been studying government practices and, in particular, the technical implementation that enforces censorship policies. Inspired by seminal work in mid-2000s [41, 42, 96, 100], numerous studies have studied the Great Firewall of China (GFW) and how it detects disallowed traffic [32, 47, 88, 95], how it tracks and blocks a connection [37, 90], and its architecture and geographic distribution [34, 48, 58, 97, 99]. Other studies have focused on country-specific censorship events, especially during times of political or social upheaval: increased Internet shutdown in Iran and India during protests or elections [11, 33, 34], HTTPS interception attacks launched by the government of Kazakhstan in 2019 [80], and blocking of social media and websites in Myanmar following the military coup in 2021 [73].

Previous research also revealed common censorship techniques being used, *e.g.*, keyword blocking that searches for forbidden keywords in unencrypted packet streams [24, 55, 92]; HTTP filtering that looks for forbidden content in outgoing URL requests as well as incoming HTTP responses [42, 43, 74]; TCP/IP blocking that terminates all connections to and from disallowed network hosts [75]; and DNS censorship where a censor returns fake DNS responses (*e.g.* IPs of blockpages) when the user attempts to resolve a blocked domain [52, 53, 76]. With the adoption of encrypted protocols such as TLS, SNI-based blocking is on the rise, where a censor filters encrypted traffic based on the Server Name Indication (SNI) field that leaks domain names in plaintext [8, 40, 85].

Countries that seek to perform nation-scale information control typically look for, or even foster, choke points in the nation's Internet topology to deploy their censorship apparatus, such as backbone networks or Internet Exchange Points [33, 48, 99]. However, for countries like Russia with thousands of privately-owned, distributed ISPs, leveraging a centralized architecture is challenging without an overhaul of the country's network topology. As a result, censorship in Russia historically follows what previous work termed a "decentralized model" [81]. Following the passage of law 139-FZ [17], Russia's federal institution for communication and media, Roskomnadzor, began maintaining a singular blocklist, officially called the Registry of Banned Sites [22] (*Blocking Registry* in this paper). However, Roskomnadzor does not prescribe the technical mechanism to enforce censorship. ISPs are required by law to ensure that the access to resources listed in the registry is blocked, but they can decide on the specific method of blocking. Previous work has found ISPs in Russia implemented different blocking mechanisms with varying efficacy, such as keyword filtering or DNS censorship [81].

This decentralized model has changed since the signing of the law "On Sustainable RuNet" on May 1, 2019 [20]. The law appoints Roskomnadzor with responsibility for implementing its provisions to counter threats to the "stability, security, and integrity" of Russia's public communication network. In particular, this law provides the legal basis for requiring ISPs to install in their networks the state-supplied DPI devices, subsequently known as TSPU devices, that are capable of "restricting access to resources of prohibited

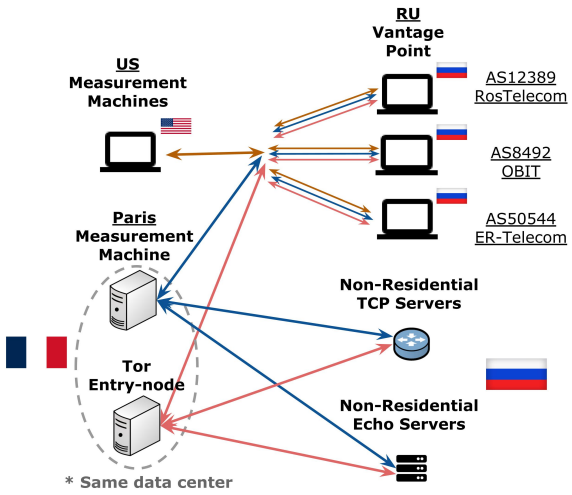

Figure 1: Measurement Setup—Non-Residential servers are selected following the procedure described in § 4.

information" [20]. These devices are made by RDP.RU [1] and distributed by Roskomnadzor, who also provides instruction to ISPs on where in the network topology the devices should be placed [83]. In March 2021, the throttling of Twitter became the first incident that captured this system being used [29]. The throttling was implemented at a national scale, but the throttling behaviors demonstrated a high degree of uniformity across ISPs, suggesting that the censorship was centrally coordinated [98]. Furthermore, the resources being censored (*i.e.*, Twitter domains) were not added to the blocking registry at the time. This event marked a departure from Russia's previous decentralized model and suggested that the new TSPU architecture has provided Roskomnadzor the means to instate censorship uniformly across the country in real-time, without relying on ISPs' technical capabilities or blocking registry. One year later, in March 2022, Russia again started throttling (and later blocking) social media such as Twitter and Facebook, claiming that by restricting access to Russian news outlets, the platforms violated the "key principles of the free flow of information" [23]. Days later, the blocking was extended to western and independent Russian news agencies, *e.g.* BBC, Meduza, Deutsche Welle, as they disseminated "false information about the actions of the Russian Army" or called the "special operation" "an attack, invasion, or a declaration of war" [7]. These recent censorship events also showed a high degree of temporal and geographical uniformity across the entirety of Russia [72].

The emerging TSPU system exemplifies a decentralized architecture that enforces uniform censorship policies on a national scale, marking a departure from other centrally deployed national firewalls such as the GFW. In the following sections, we describe our understanding of the TSPU based on measurements of its design, capability, scale of deployment, and circumvention strategy.

3 MEASUREMENT SETUP

To study the details of how the TSPU functions, we built our measurement infrastructure as shown in Figure 1. First, we need vantage points inside Russia that experience censorship, not only from ISP DPIs but specifically also from TSPU devices. This is challenging because anecdotal reports suggest that only residential networks are targeted. Indeed, all data center VPSes we rent show little to no signs of censorship, while all three residential vantage points we own experience heavy censorship. Our three in-country vantage points are located inside three different residential ISPs (in St. Petersburg, Moscow, and Krasnoyarsk, respectively), and we use them to observe censorship akin to what an user in a residential network may experience.

To study SNI-based blocking, from our Russian vantage points, we connect to two dedicated measurement machines in the US, located in the same network, and send different types of traffic—often with triggers—while capturing traffic from both ends for analysis. In addition, to measure IP-based blocking on the TSPU, we use a Tor entry node we own located in Paris, France. The IP of the Tor node, while not present in the blocking registry, has been blocked by the TSPU ("out-registry" block, as shown in § 5.2) since December 2021. While the Tor node is no longer in operation as of March 2022, residual censorship remains in place up until the date of submission of this paper.

All measurements, unless stated otherwise, were repeated multiple times (>5) to account for the TSPU failure or transient routing changes. For measurements that involve sequential tests (*e.g.* § 5.3.2), we randomized their order and ensured that each test used a fresh source port on Russian vantage points to prevent residual censorship affecting results of subsequent tests. In addition, we establish "control" experiments in order to identify blocking triggers. For example, we obtained control endpoints in Paris ("Paris Measurement Machine") located in the same data center as the Tor entry node to rule out potential routing and load-balancing effects in our IP-based blocking measurements. This pair of machines is also used in measurements where we want to minimize the effect of path differences when correlating results from SNI-based and IP-based censorship (§ 7.2). For SNI-based blocking, we conduct a control experiment by making connections between the same pairs of endpoints but with a non-triggering SNI in ClientHello. For QUIC control experiments, we remove the fingerprint that the TSPU uses to detect the protocol.

In addition to vantage points and measurement machines, we also select a group of echo servers and general-purpose TCP servers for our measurements in § 7. Since these measurements involve sending sensitive traffic, namely, either SYN packets from the Tor entry node IP or ClientHellos with triggering SNIs, we select only non-residential targets [1] by following the procedure described in [89]. In total, we select 1,136 of 1,404 echo servers and 13,309 of 200,000 TCP servers. For all remote measurements that involve sending any sensitive traffic, we restrict ourselves to testing non-residential targets only. Please refer to § 4 for more details regarding ethical considerations of our remote measurements.

[1] By non-residential, we refers to targets that are more likely to be embedded network infrastructure such as routers or switches, compared to end-user devices. But it is possible that these targets reside in residential networks.

4 ETHICS

Before performing any measurements, we carefully review the details of the measurement and we adopt best practices described in the Menlo report [44] as well as community norms from similar past studies [39, 80, 81, 89, 98]. We evaluate potential risks that our measurements may incur through discussions with prominent activists within Russia and with experienced colleagues from the censorship and measurement fields. We ourselves have performed this kind of research in prior work and are aware of the risks.

For our in-Russia measurements, we only use vantage points owned by the team. For one of our remote measurement techniques, we send TCP SYN packets (no other traffic) from one of our VPS machines in a data center that used to be a Tor entry node. The IP of the node, while not present in the blocking registry, has been blocked by the TSPU. We note that the Tor instance has not been in operation since March 8, over a month before our remote measurement started, but residual IP blocking remained in place. We also note that the network traffic resulting from such SYN scanning differs from the traffic pattern of a typical Tor user in both traffic direction and duration. For our other remote measurements, from our VPS in data center, we use the peer-reviewed technique, Quack, that sends TLS ClientHellos containing triggering domain names inside the SNI field to echo servers.

For all measurements involving remote hosts not owned by us, we took a complementary set of approaches to reduce potential risks. First, for measurements that require sending censorship triggers, we limit the scale of measurements by targeting only IPs that are less likely to be end-user devices following the procedure outlined in [89]. Specifically, for each IP candidate, we performed OS detection using Nmap, and select only those IPs that have full device labels containing the words "router" or "switch". We performed all follow-up measurements within 24 hours of target discovery to minimize potential IP churns. The goal is to ensure that in the unlikely event of authorities tracking down violating traffic, it would be obvious that the traffic was not generated by an actual user on browser. In addition, for our censorship triggers, we used only domains that are general-purpose services popular among Russian and non-Russian users (*e.g.*, *facebook.com*) as opposed to sites with more targeted user pools (*e.g.*, sites with political leanings). Furthermore, our dedicated measurement machines are provisioned with web pages that explain the nature of our research and provide contact information. Finally, for our large-scale measurement, we sent only innocuous traffic (*e.g.*, fragmented SYN packets) without any censorship trigger. While we acknowledge that some techniques can be imperfect, *e.g.*, inaccuracy of Nmap's OS fingerprinting, we believe that a combination of these complementary approaches allows us to responsibly balance the benefits and potential risks of our remote measurements.

5 HOW DOES THE TSPU BLOCK?

We start our investigation by asking: *how does the TSPU block a connection, and what triggers it?* To answer these, we first need to be able to attribute observed blocking behaviors to the TSPU, which is not trivial as there are no previous studies to use as a reference point. Therefore, we start by looking at anecdotal reports from Russian users about popular services and websites being unavailable since the end of February, including social media [62, 64, 66, 67, 86], news agencies [59, 61, 72], applications and circumvention tools [65, 69–71], and QUIC/HTTP3 [54, 60, 68]. We test the connectivity to these resources from our in-country vantage points. In § 6, we expand our test list with top domains and resources found in Roskomnadzor's blocking registry, for completeness.

5.1 Identifying TSPU Blocking

We attribute an observed blocking instance, including the trigger and the corresponding blocking behavior, to the TSPU based on the following rationale. First and foremost, TSPU blocking should show a high degree of uniformity in blocking behaviors across ISPs. This is because TSPU devices are ordered, distributed, and controlled by Roskomnadzor. This is in contrast to blocking performed by individual ISPs, who are responsible for maintaining their own blocklists and blocking equipment. Previous work found that the lack of a prescription on how to implement the censorship mechanism has led to each ISP implementing different methods [81].

In addition, uniformity is also expected for the blocking targets as well. While individual ISPs query Roskomnadzor's blocking registry, but each maintains a separate blocklist, the centrally-controlled nature demands that the blocklists used by TSPU devices scattered in different ISPs be uniform at a given time. Also, while ISP blocklists are mostly subsets of the blocking registry, the TSPU is capable of blocking resources that are not present in the registry (out-registry blocking).

Finally, we expect that different blocking mechanisms that are attributed to the TSPU are co-located. This could mean that the blocking happens at the same network location (hop), based on TTL-limiting measurements as described in § 7. And, in some cases, two mechanisms conducting blocking in a collaborative way offers additional evidence of co-existence (*e.g.* § 5.3.2).

5.2 Triggers and Blocking Behaviors

We use *Trigger* to refer to the offending signature of a packet that causes a TSPU device to initiate blocking, and *Behavior* to refer to how connections are blocked. We identify three types of triggers: SNI-based, IP-based, and QUIC blocking, leading to six different behaviors. All observed combinations satisfy every consistency and co-location assumption listed above. Figure 2 shows the diagrams of these blocking mechanisms.

SNI-based Blocking. The TSPU primarily relies on matching the SNI field in a ClientHello and can affect domains both within and outside the blocking registry. A ClientHello with a targeted SNI destined for port 443 sent from the local (RU) side to the remote (non-RU) side will trigger censorship behaviors from a TSPU device. To identify which parts of the ClientHello are inspected by the TSPU, we fuzz the triggering ClientHello with different alteration strategies, such as padding the SNI, changing TLS versions, adding ClientCert or different Ciphersuites, or corrupting protocol parsing by masking the "length" fields. Figure 13 in Appendix shows which parts of a ClientHello packet are inspected by the TSPU. Specifically, we find that by altering values in positions that represent "type" or "length" would lead to different censorship behaviors. This suggests that the TSPU parses a ClientHello in order to locate the

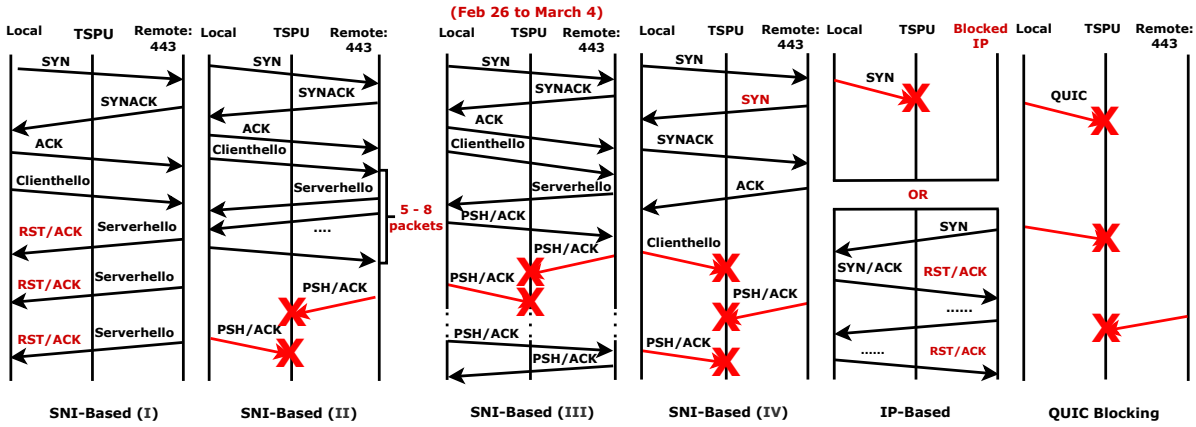

Figure 2: Different blocking Behaviors—SNI-based trigger leads to four different behaviors depending on the domain. Throttling (SNI-III) was observed between Feb 26 and March 4, 2022, but has since been replaced by RST-based blocking (SNI-I).

SNI field, rather than matching the targeted domain strings over entire packets.

Interestingly, the domain within the SNI affects the blocking behavior. We identified four different behaviors that all trigger on SNI. **SNI-I:** The majority of domains targeted experience response modification wherein the TSPU modifies Remote-to-Local packets by truncating the payload and changing the TCP flags to RST/ACK after the ClientHello is seen. Other packet metadata, such as TTL, sequence and acknowledgement numbers, are not altered. **SNI-II:** A different behavior is induced for a select list of "out-registry" domains, e.g., *news/play.google.com* and *nordvpn.com*. Specifically, once a triggering ClientHello is seen, an additional five to eight packets can be delivered from either side, after which symmetric packet drops occur. **SNI-III:** For some domains (*e.g. twitter.com*, *fbcdn.net*), during the period between February 26 to March 4, 2022, hard-throttling was in place. Compared with last year's throttling event [98], the same traffic policing mechanism was used, which drops packets that exceed the rate limit. However, this time, the rate limit was much lower, at around 600-700 bytes per second (compared to 130kbps from last year). On March 4, such throttling behaviors stopped and the sites affected have been seeing the SNI-I: behavior thereafter, *i.e.* RST/ACK. **SNI-IV:** This method targets a selective list of domains associated with Facebook, Twitter and Instagram, which are also targeted by SNI-I. In most cases, these domains are blocked with RST/ACK. However, with certain TCP sequences, another blocking mechanism is also applied that immediately drops all packets from both sides, including the initial Clienthello, presumably as a backup filtering mechanism to method SNI-I. We detail how the two mechanisms interact in § 5.3.2.

QUIC Blocking. On March 4, Russian users reported that QUIC connections could not be established [54, 60, 68]. It has been identified that the TSPU uses a fingerprint based on plaintext byte patterns signaling QUIC version (0x00, 0x00, 0x00, 0x01 for version 1), and it applies the filter for any UDP packet destined to port 443 that has at least 1001 bytes in its payload [68]. Note that the fingerprint only targets QUIC versions 1, which is the most commonly used version. Other versions are not targeted yet, and quicping and QUIC draft-29 can evade blocking, which use (0xba, 0xba, 0xba, 0xba) and (0xff, 0xff, 0x00, 0x1d) in their version fields, respectively [54]. Similar to SNI-based blocking, the TSPU is only triggered if such a QUIC packet is sent from the local (RU) side, after which all following packets from both sides will be dropped immediately. Figure 14 in Appendix shows a minimum "fingerprint" that the TSPU uses to detect QUIC. Note that once such a packet is detected, all following packets from the same flow will be dropped, regardless of their length or the presence of the QUIC fingerprint.

IP-based Blocking. Finally, we found that the TSPU blocks connections to/from certain IP addresses. Specifically, if a Russian client behind a TSPU device tries to contact the blocked IP, the outgoing packets would be dropped at the TSPU. If the blocked IP initiates a connection to a server behind the TSPU, the request (*BlockedIP -> RussiaServer*) can pass through, while the response will have its payload stripped off and TCP flags changed to RST/ACK. The censorship is applied regardless of packet payload or TCP ports. ICMP Pings to/from blocked IPs are also dropped. Besides the Tor entry node IP, we also found six additional IPs that are being blocked with the same set of behaviors, including IPs from VPN providers and Google services. None of these IPs were present in the blocking registry at the time of testing.

We note that each of the six blocking behaviors requires the TSPU to modify or drop packets in order to sever a violating connection. Such capability suggests that TSPU devices have *in-path* components. We highlight that this means the TSPU have more means to interfere with users' traffic than known *on-path* censorship systems, such as the GFW [41, 42, 58].

5.2.1 Trigger Reliability. Previous studies suggest middleboxes do not consistently apply censorship policies [39, 42, 48]. To test the reliability of the three TSPU blocking mechanisms, we send 20,000 requests from our RU vantage points to measurement machines, with different triggers corresponding to each blocking type. For *SNI-based* and *QUIC* blocking, we use measurement machines located in the US and look for expected blocking behaviors. Unfortunately, we could not measure throttling (SNI-based III) before it was changed

	SNI-I	SNI-II	SNI-IV	QUIC	IP-Based
Rostelecom	0.084%	0.0025%	0.27%	0.02%	0.00%
ERTelecom	N/A	1.76%	2.19%	0.93%	0.045%
OBIT	0.14%	0.005%	0.04%	0.00%	0.02%

Table 1: Percentage of TSPU failures—Note that while we observed throttling (SNI-III) during the period of Feb 26 to March 4, the behavior was replaced by outright blocking before a reliability experience could be performed.

to RST/ACK (SNI-based I) on March 4, 2022. For *IP-Based* blocking, we send TCP SYNs from our Tor entry node located in France and respond with SYN/ACKs from the vantage points, and then we check whether the returned packets are changed to RST/ACKs.

We conducted these experiments multiple times, during different times of the day and report on the average results. We also tried different levels of concurrency but found no observable differences from sequential testing results. Table 1 shows the average percentage of un-blocked connections broken down by vantage points and blocking types. While we found that TSPU devices are generally effective in enforcing censorship policies, we also noticed the ER-Telecom vantage point has seen significantly more censorship failures for some blocking types than the other two vantage points. Further investigation revealed that for our vantage points in Rostelecom and Obit, there are more than one TSPU devices on the path to the US/France testing servers. As a result, requests from these two vantage points require both devices to fail in order to avoid censorship. We explain the identification and localization methodologies in § 7.1.1.

5.3 TSPU State Management

Previous work has found that the GFW operated in a stateless manner before 2007 and had become stateful thereafter [41, 42, 88, 100]. During our investigation of the TSPU, we found multiple cases where changes in the IP or TCP layer resulted in different censorship behaviors, implying a degree of statefulness when the TSPU makes access control decisions. For IP, we focus on how the TSPU handles IP fragmentation, a well-known source of stateness that violates the statelessness of IP [26]. For TCP, we explore sequences of TCP flags that create states at the TSPU's connection tracking component that bypass blocking, and from observed behaviors we estimate timeout values for different states.

5.3.1 IP Fragmentation Behaviors. We investigate how the TSPU handles fragmented IP packets in general, regardless of censorship being triggered or not. First, we found that TSPU devices buffer incomplete fragments, but does not de-fragment them before forwarding to the next hop. Specifically, fragments are buffered at the TSPU, and once the last fragment arrives, all fragments are forwarded individually, without reassembly. Figure 3 depicts this behavior. In addition, when fragments are forwarded to the next hop, the Time-to-live (TTL) field of the first fragment (identified by zero offset) is used for subsequent fragments. The fact that TSPU devices buffer fragments may suggest that it is capable of reassembling and inspecting fragmented IP packets.

In addition, we found that the TSPU enforces several restrictions on fragmented packets, and a "malformed" fragment will cause the current fragmentation state, identified by the source, destination

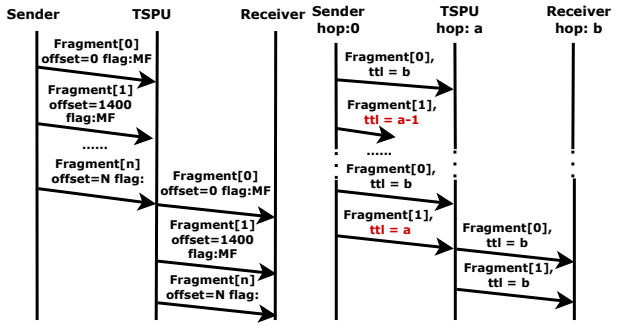

Figure 3: TSPU handling of IP fragmentation—Note that it buffers IP fragments and forwards the fragments individually after the last fragment arrives. In addition, when fragments are forwarded, the TTL fields of second to last fragments are rewritten to the TTL of the first fragment.

IP and IPID tuple, to be discarded. Specifically, if one or more fragments is duplicated or share an overlapping payload (*i.e.*, it has the same or overlapping offset as another fragment), then no fragment belonging to this IP packet will be forwarded. Furthermore, the fragment cache on TSPU devices seems to have a short timeout of around 5 seconds, and if by then there are still fragments missing, the entire queue is discarded. Finally, we found that TSPU devices enforce a limit of 45 as the maximum number of fragments permitted in a single packet. Specifically, TSPU accepts up to 45 fragments of a single packet before the entire fragment queue is discarded. These behaviors can be observed by sending fragments through a TSPU device in either direction, and in § 7 we leverage these behaviors to remotely measure the deployment of TSPU devices.

5.3.2 TCP Sequences. We explore different TCP sequences for the ones that establish a "state" at the TSPU for inspection and possible blocking. Our testing methodology consists of sending TCP packets between the local and remote endpoints, alternating source and destination, and modulating the TCP flags. We exhaustively searched all combinations of such sequences of length up to 3 packets. For each sequence, we determine if it is a valid prefix sequence for TSPU blocking by appending a triggering ClientHello and observing if the connection exhibits expected blocking behaviors.

As shown in Figure 4, we first notice that any sequence starting with a packet sent by the remote peer is NOT a valid prefix to trigger the TSPU. This means that the censorship is likely not symmetric with respect to inside and outside Russia, which is also noted in previous work [98]. Such behavior makes remote measurement extremely challenging. For example, for out-registry blocking (*e.g. play.google.com*) that is only blocked by the TSPU, remote measurement platforms such as Censored Planet are not able to detect this anomaly whereas the in-country observatory OONI reports over 70% of web connectivity tests as anomalies [19]. The fact that TSPU censorship depends on which machine, local or remote, sends the first packet suggests that TSPU devices maintain states about packet direction, which is challenging for stateful DPIs to handle unambiguously. Specifically, a DPI may use heuristics in the packet header to infer the roles of "client" and "server".

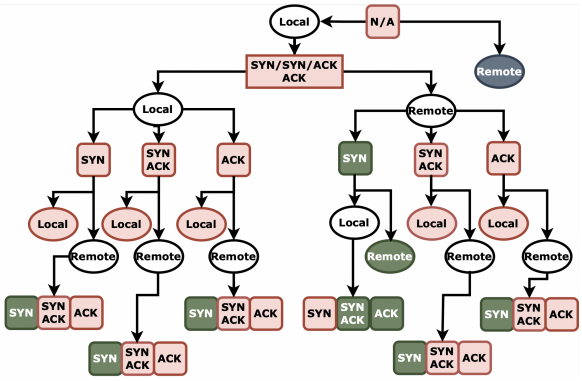

Figure 4: TSPU Triggering Sequences. Path to green nodes are sequences that evade SNI-I but not SNI-IV.

In the case of TCP, the first "SYN" packet usually indicates which side is the "client". Corner cases in TCP exist, such as Simultaneous Open and Split Handshake [87], where both client and server send a SYN packet during the initial handshake phase. Devices that rely on literal SYN or SYN/ACK packets to infer which machine is "client" and "server" can be tricked into reversing those roles. Indeed, we found that sequences that start by an outgoing (local to remote) packet but later contain an incoming SYN packet (colored green in Figure 4) show different behaviors from other sequences. For sequences with remote-sent SYNs, most sites targeted by SNI- I are no longer blocked. However, for sites also targeted by SNI- IV, such as sites corresponding to Twitter and Facebook domains, a different blocking behavior is triggered (SNI- IV), which drops all packets from both sides. We note that SNI- IV requires a longer prefix sequence to trigger, and it is only triggered when SNI- I fails to take action (otherwise RST/ACKs would be dropped as well). Such behavior is highly similar to the "backup" filtering mechanism described in previous work on the Great Firewall of China [39].

5.3.3 Timeouts for TCP States. Connection tracking systems trade-off between maintaining states for connections they have previously seen without exhausting system resources or storing stale connection information. This is achieved by removing entries for connections after a timeout expires. We adopt a methodology based on sending TCP packets with different sequences of flags to induce state transitions and to estimate different timeout values. Our timeout estimation methodology is based on the fact that we can differentiate states based on whether a trigger causes the TSPU to block a connection or not, as well as the duration for which the blocking or bypassing persists.

We focus on SYN-SENT, SYN-RECEIVED, and ESTABLISHED because these states exist at the beginning of a TCP connection while it is in active use. We do not include the SYN2 state of Simultaneous Open because we could not measure a distinct timeout value for it, and we found no evidence that the TSPU tracks the SYN2 state (and some evidence that it does not, in that Simultaneous Open evades some TSPU blockings). The SYN-SENT state is entered whenever a host initiates a TCP connection via the TCP three-way handshake. The SYN-RECEIVED state is entered when a host (typically the "server") receives a SYN packet in the three-way handshake. Specifically, we estimate the timeouts of a state

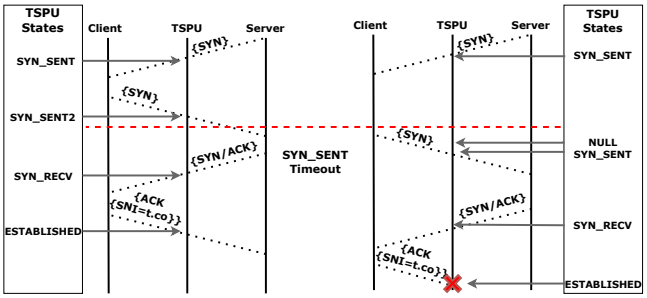

Figure 5: Timeout inference for TCP SYN-SENT.

Sequence	Timeout	State
Remote.SYN; SLEEP; Local.SYN; Remote.SA; Local.Trigger	60	SYN_SENT
Local.SYN; Remote.SYN; Local.A; SLEEP; Local.Trigger	105	SYN_RCVD
Local.SYN; Remote.SA; SLEEP; Remote.ACK; Local.Trigger	480	ESTABLISHED
Local.Trigger(SNI- I); SLEEP	75	SNI- I
Local.Trigger(SNI- II); SLEEP	420	SNI- II
Local.Trigger(SNI- IV); SLEEP	40	SNI- IV
Local.Trigger(QUIC); SLEEP	420	QUIC

Table 2: Sequences for state timeout measurements.

by sending sequences of packets, such as SYNs from the remote peer, then wait for some period of time T before sending triggers[2] and check whether blocking is induced. We repeat the experiment while iteratively adjusting T until we find a threshold that consistently leads to different behaviors (blocking or bypassing). Figure 5 shows an example of how we estimate the timeout for SYN-SENT. Note that we use SNIs not present in the blocking registry to avoid potentially inducing interference from ISPs' filtering devices on network paths.

Table 2 summarizes the timeout values we found corresponding to different TCP states and the timeouts for different blocking polices once they are triggered. Table **??** and Table 7 in Appendix show timeout values corresponding to other states as well as popular OSes and specifications. We note that the timeout values for the TSPU do not seem to conform to any other OSes with documentation. Specifically, the TSPU has much shorter timeouts for SYN-SENT and ESTABLISHED when compared to Linux and FreeBSD. In § 8, we discuss how such short timeouts may help with devising circumvention strategies.

6 WHAT DOES THE TSPU BLOCK?

After identifying blocking behaviors that can be attributed to the TSPU, we want to understand what resources or categories of resources are being targeted by expanding measurements to cover more domains.

6.1 Testing Input Lists

Tranco list. We first select the top 10k domains as ranked by Tranco [79], a research-oriented top sites ranking list (Alexa has been deprecated as of May 1, 2022). Following previous studies [63, 77], we complement this list with domains from the Citizen Lab Global Block List (CLBL) [57]. CLBL contains URLs specifically

[2]On vantage points with multiple TSPU devices, we TTL-limited the triggers to the first symmetric TSPU device to ensure server-sent packets are seen. See § 7.1.1.

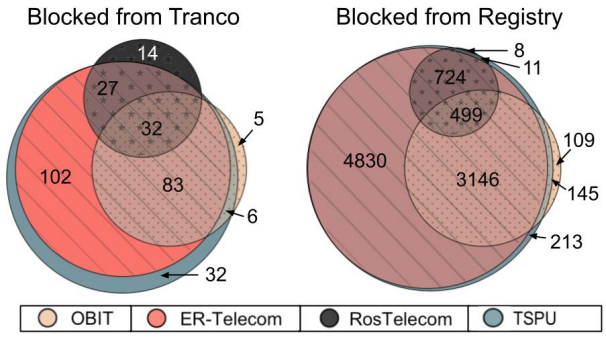

Figure 6: Sets of domains blocked by ISPs and the TSPU—TSPU blockings are more consistent across vantage points.

selected for censorship testing, including a range of popular websites with content that is "provocative or objectionable". In total, our *Tranco list* contains 11325 unique domains, most of which are in English. Both Tranco and CLBL lists provide category information for their domains.

Registry Sample. Roskomnadzor maintains a centralized blocking registry after the passage of federal law 139-FZ [17]. This blocking registry is public but protected with a CAPTCHA [22], where only singular queries of domain or IP can be made. Since bulk querying is not possible, we obtained a link to a "leaked" repository that contains a copy of the blocked domains that is distributed by Roskomnadzor to ISPs [21], regularly updated since 2012. Previous studies validated this registry by comparing it with samples signed by Roskomnadzor and found that the two registries are practically identical [81]. We therefore use this registry to create our *Registry Sample* by randomly sampling 10,000 domain names that have been added to the registry since January 1, 2022.

Categorizing our *Registry Sample* is not trivial. Roskomnadzor's blocking registry does not provide any category information and online classification services such as Fortiguard [50] do not work well for Russian sites. Following previous work, we used the topic modeling algorithm developed by Ramesh *et al.* [81] based on previous techniques [93]. From our measurement machine in the US, we visited and obtained the HTML responses for each domain from our *Registry Sample*. Then, we clustered the received webpages using Latent Dirichlet Allocation (LDA) clustering to identify common topics [35]. Refer to [81] for a more detailed description of the algorithm. We focused only on webpages whose primary language is Russian or English in order to reduce the manual effort of labelling topics. Finally, we manually merge the topics from the *Tranco list* and our *Registry Sample* into 11 categories.

6.2 Testing Methodology

To test for TSPU censorship, we crafted ClientHellos containing a test SNI to send from the three RU vantage points to the measurement machines in the US. We coordinated RU clients and measurement machines to send different packet sequences to test all types of SNI-based blocking, and we captured packets from both sides for reference. For example, to test for SNI-based IV, the measurement machines were configured to respond to a SYN with a SYN to start a split handshake.

Types	Domains
SNI- I	infox.sg, tor.eff.org, googlesyndication.com, theins.ru, twimg.com, t.co facebook.com, twitter.com, dw.com, instagram.com ...(9899)
SNI- II	nordaccount.com, play.google.com, news.google.com, nordvpn.com
SNI- IV	twimg.com, t.co messenger.com, cdninstagram.com, twitter.com, web.facebook.com, numbuster.ru

Table 3: Domain blocking types Out-registry domains. Domains added to the registry after 2022-02-24

In addition, we also want to test censorship by ISPs. Traditionally, censorship in Russia has been carried out in a "decentralized" form [81], where each ISP implements its own blocking devices and keeps its blocklist up-to-date. We want to understand the characteristics of TSPU blocking in comparison to ISP blocking. To observe censorship from ISPs, we first notice that all three RU vantage points are located inside residential ISPs that implement blockpages as their blocking methods. Specifically, ISPs' DNS resolvers would return IPs pointing to the ISP's blockpage, which is different from ISP to ISP, in response to any query with a domain name found inside the blocklist maintained by the ISP. We select three local resolvers inside the three RU ISPs, and send queries to them once from the RU vantage points and once from US measurement machines. We find no difference in responses between the two cases. We focus on only blockpage-based blocking for two reasons: previous work suggests that typically a single ISP-implemented blocking method dominates at each ISP; and, for residential ISPs the method is blockpage-based blocking, which is also consistent with Roskomnadzor's guidelines [18, 81].

6.3 Results

We first note that the list of domains that trigger TSPU blocking are consistent across vantage points, as expected. As shown in Figure 6, for both the Tranco list and our *Registry Sample*, blocking implemented by individual ISPs falls behind the TSPU in terms of coverage. Most of the Tranco domains that are blocked only by the TSPU are not present in the blocking registry ("out-registry" blocking), most of which are Google services, circumvention tools, news and pornography sites. For our *Registry Sample*, we note that resolvers in Rostelecom and OBIT do not enforce blocking effectively on domains recently added to the registry, returning blockpages for only 1,302 and 3,943 domains, respectively, while the TSPU blocks the same list of 9,655 domains in all three ISPs. This result illustrates the major change underway in Russian censorship practice: the previous decentralized model where each ISP enforces different blocklists is being superseded by a more centralized one, with the TSPU enforcing the blocking registry in a more uniform and effective way.

Figure 7 shows the categories for domains blocked by the TSPU. Consistent with previous work on Russia's censorship [81], the majority of blocked domains are for gambling, news, and streaming/media sites. In particular, the category "Informative Media" sees the largest number of domains being blocked, which include sites from news agencies, personal blogs, social media and multimedia platforms, suggesting active interference in the free flow of information. Table 3 shows the blocking types for domains censored by the TSPU. We note that the vast majority of blocking was implemented by SNI- I (RST/ACK). In particular, the list of domains blocked by SNI- IV is a select subset of targets of SNI- I. It is unclear to us why

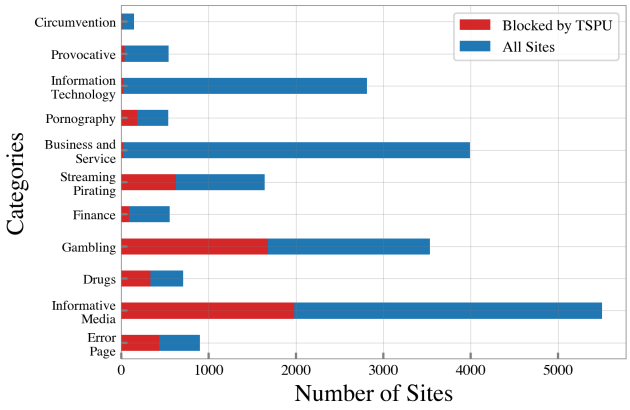

Figure 7: Domain Categories. Excluding (1398+2680) domains that failed TCP, or empty/unparseable HTML responses. *Error page* includes geoblocked or parking pages.

these domains are specially targeted other than that most of them are associated with Twitter and Facebook.

7 WHERE DOES THE TSPU BLOCK?

We investigate *where* TSPU devices are located. We start by measuring how far they are from our RU vantage points with TTL-limited measurements. In addition, we devise novel techniques based our knowledge about TSPU devices from previous sections to remotely identify and locate them.

7.1 Local-to-Remote Localization

To identify where in the network path TSPU blocking occurs, we employ a TTL-based technique similar to traceroute. Specifically, from each RU vantage point, we send two types of packets, trigger and control packets, to the measurement machines. Trigger packets have increasing TTL values, and have the same payload as described in § 5, *i.e.*, ClientHello with a blocked SNI or a UDP packet of more than 1001 bytes with the QUIC fingerprint. The control packets are basic non-triggering packets that are used to either establish a state (*e.g.* sequences ot TCP packets with flags) or to infer whether blocking has occurred as a result of trigger packets. To estimate the network location of TSPU devices, we repeat the experiment while assigning trigger packets increasing TTL values. If we identify some TTL value *N* where we do not observe blocking behavior but TTL *N+1* results in blocking, then we report that the TSPU device exists between hop *N* and *N+1*.

For all three vantage points, we identified that the corresponding TSPU device was located within the first three hops. This is consistent with the installation guideline sent by Roskomnadzor to ISPs, which recommends that TSPU devices be installed before carrier-grade NAT, close to end users [83].

7.1.1 Multiple TSPU devices on a network path. On Rostelecom and OBIT, we identified a second TSPU installation location further away from our vantage points. To explain how we identified them, we first note that asymmetric routing is common in Russia: on all three vantage points, our upstream and downstream traffic

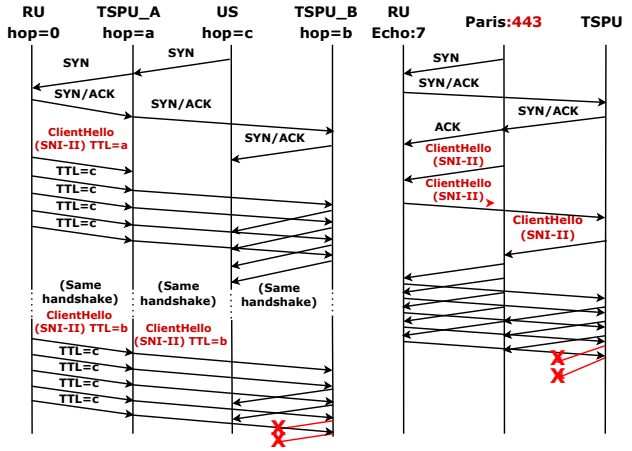

Figure 8: TSPU devices with partial visibility—Left: Experiment to identify upstream-only TSPU devices from RU vantage points; Right: Remote measurement using echo servers within Russia.

would traverse different hops, and in some cases different transit ISPs within Russia, depending on the destination. Based on this observation, we designed an experiment to identify TSPU devices that have only partial (upstream-only) visibility into users' traffic.

Figure 8 (left) shows a diagram of the experiment. On a high level, the experiment is based on the fact that the TSPU only applies its blocking rules to flows initiated remotely. We trick a TSPU device with only upstream visibility into reversing the roles of "client" and "server" for a remotely-originated flow and therefore applies blocking rules. Specifically, we assume there are two TSPU installation points, A and B, with A being close to the end user and B exists somewhere on the path between A and the remote destination (a<b<c). TSPU A, being close to end user, has symmetric visibility whereas TSPU B only observes upstream (RU to US) traffic. We start by having the US machine send a SYN packet to the RU vantage point, which is seen by TSPU A but not TSPU B. RU vantage point, upon receiving the SYN, completes the handshake by returning a SYN/ACK, which traverses through both TSPU devices. Next, we send a ClientHello with a reduced TTL and a SNI from the SNI-II group (because SNI-II affects both downstream and upstream packets, whereas SNI-I acts only on downstream packets). Note that even if this ClientHello passes TSPU A, censorship would not be triggered since the connection was initiated by the US peer. However, from the perspective of TSPU B, the connection was initiated with a RU-sent SYN/ACK (which is unusual but we show in Figure 4 that a single SYN/ACK is a valid prefix). Therefore, as soon as the ClientHello passes TSPU B, the session would see the SNI-II blocking behavior.

We repeat this experiment on all three vantage points with increasing TTLs and different destinations. On OBIT, we identified two upstream-only TSPU devices, both located at the first link of the transit RU ISP (Rostelecom and RasCom, depending on the destination). This seems to suggest that transit (upstream) ISPs can have their own TSPU devices in place to filter traffic on behalf of their client networks, which is consistent with the requirements listed in the executive order "Requirements for the Procedure for Passing

	Echo Servers	Nmap-filtered	TSPU-positive
IPs	1404	1136	417
ASes (Networks)	188 (344)	47 (141)	15 (69)

Table 4: Results from measurements with Echo servers—"Nmap-filtered" are targets that are deemed unlikely to be end-user devices following the procedure outlined in § 4.

	Echo (N)	Echo (B)	Hamming Distance
IP (N)	673	12	0.0493
IP (B)	44	405	
	Fragment (N)	**Fragment (B)**	**Hamming Distance**
IP (N)	828	85	0.0199
IP (B)	151	7567	

Table 5: Results of TSPU IP blocking, Echo and Fragmentation measurements. B=Blocked, N=Not blocked.

Traffic in Data Networks" [30] that exempts network operators from installing TSPU devices on their own links, if they can route their traffic to TSPU devices installed by their upstream providers [2]. On Rostelecom, we identified an upstream-only TSPU device one hop (still in the same AS) behind the TSPU device that has symmetric visibility. Our findings are supported with TCP traceroutes from both directions. We note that we are only able to identify the first symmetric and upstream-only TSPU devices on a network path, and there could be additional installation locations further down the network path, which we are unable to identify. Furthermore, since the blocking triggers (§ 5) need to be sent in the upstream direction, we are not able to identify TSPU devices that see only downstream traffic.

We note that TSPU devices with asymmetric visibility may result in *underblocking* (*e.g.* SNI-I is triggered by upstream ClientHello but acts only on downstream traffic) and *overblocking* (since the TSPU may not be able to determine whether the connection is initiated within Russia).

7.2 Remote Measurements

To answer questions such as "where TSPU devices are deployed" at scale, we are limited by the number of vantage points we have. We therefore attempt to identify TSPU devices by remote measurements. As shown in § 5, remotely identifying TSPU devices is extremely challenging, as TSPU blocking is not applied symmetrically. Furthermore, considering the volatile situation in the region, we do not want to rely on crowdsourced measurement, since it involves having volunteers sending disallowed traffic to trigger censorship. We devise two methods to identify and locate TSPU devices remotely and ethically, using knowledge we gained in previous sections. We validate the two methods on a select set of non-residential-user (*i.e.*, routers or switches) targets within Russia.

Echo Measurements We demonstrate in § 7.1.1 that TSPU devices can be installed potentially inside transit ISPs, and when they do they may have an asymmetric view of users' traffic. In particular, for upstream-only TSPU devices, the roles of "client" and "server" may be reversed since it does not see the initial SYN packet sent by the remote peer. However, the triggering ClientHello would still need to traverse the TSPU from within Russia in order to trigger censorship.

Based on this observation, we design an experiment using echo servers within Russia in order to extend measurements beyond vantage points we own. As shown in Figure 8 (right), we run Quack [89] from measurement machines to echo servers on TCP port 7. After the initial handshake, we send a ClientHello with a target SNI and wait for it to be echoed back. Next, we send 20 TCP packets with random payloads to the echo server, and count the number of packets received. If we receive all 20 packets with a non-offending SNI but less than that (we use a threshold of ≤ 5) with a domain from the SNI-II group, then we conclude there is a TSPU device on path that is triggered by the ClientHello.

We used ZMap [46] to scan all Russian IP prefixes on TCP port 7. For each open port discovered, we sent TCP packets with random payloads, and we selected servers that were still echoing our packets after we sent 20 packets. Next, following previous work [89], we used Nmap's OS detection module to scan the IPs according to the process described in § 4. For each remaining endpoint, we test it from our Paris measurement machine following the experiment described above. Table 4 shows the results. We were able to observe TSPU blocking from 417 echo servers located in 15 ASes, which is surprising as it shows upstream-only TSPU devices can be prevalent on Russia's network. Interestingly, to trigger blocking, the client (ephemeral) port on the Paris machine needs to be set to 443, which further confirms our hypothesis that the roles of "client" and "server" are reversed from the perspective of TSPU devices with partial visibility.

To correlate our echo results with other TSPU behaviors, we try to connect to these servers from the Tor entry node that is "out-registry" blocked by the TSPU. We note that the Tor node is located in the same data center as the measurement machine used in the echo experiment so the effect from routing differences should be minimized. Specifically, for each echo server, we send a SYN packet to TCP port 7 and we label the test as "IP Blocked" if the returning SYN/ACK is changed to RST/ACK, which is the expected behavior for IP-based blocking on the TSPU. Table 5 suggests a strong correlation between IP-based blocking by the TSPU and the observed SNI-II blocking on echo servers.

Fragmentation Measurements To further scale up our measurements of the TSPU, we employ a different technique based on how it handles IP fragmentation. § 5 finds that TSPU devices buffer fragmented IP packets, and it: 1) accepts up to 45 fragments of a single packet before discarding the entire queue; 2) drops the entire queue if duplicate or overlapping fragments are found; and 3) changes the TTL of each subsequent fragment to the TTL of the first fragment. We exploit the first two behaviors to identify whether there is a TSPU-like DPI on the path, and we use the third behavior to pinpoint the network location of the device.

We first note that a fragment queue limit of 45 is not a common behavior. Linux has this value set by default to 64, whereas on Cisco and Juniper network devices the limit is 24 or 250, respectively [6, 14, 15]. We randomly sampled 1 million US hosts with TCP port 7547 open (the port where we saw the most TSPU-like fragmentation behavior in our Russian scan), and we found only 0.708% endpoints to have a similar limit on the size of fragmentation queue, most of which are from a single AS (AS17306). In addition, while TSPU devices drop fragment queues with duplicates, RFC 5722 notes that

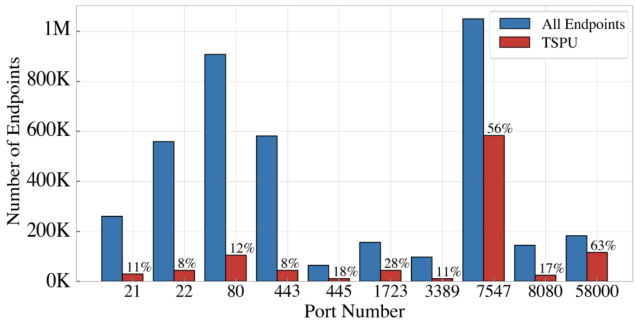

Figure 9: Endpoints with TSPU installations by port—High percentage of TSPU interference on endpoints with port 7547 could suggest that residential ISPs are more targeted.

duplicates should rather be "ignored" and the queue kept for later reassembly [51, 56].

We conduct a similar test to see the correlation between the fragmentation behaviors and IP-based blocking by the TSPU. We sampled 200,000 RU hosts on TCP port 7547 and filtered 13,309 non-residential endpoints. Prior to testing, we performed control measurements to remove all endpoints that do not respond to either any SYN packets, SYNs with payloads, or fragmented SYN packets. For each of the remaining 8,566 endpoints, we sent a SYN packet from the Tor entry node and two fragmented SYN packets, with fragment size of 45 and 46, from the Paris measurement machine. We labeled an endpoint *Fragment (B)* if it responded to fragments of size 45 but not 46, and *IP (B)* if we recorded at least one RST/ACK from the Tor node. Results shown in Table 5 suggest a strong correlation between the two behaviors. We suspect, but are unable to verify, that disagreements may be due to either downstream-only TSPU devices or other middleboxes on the path that reassemble fragments.

To identify where the TSPU device is located on a path, we exploit the the fact that when a TSPU device forwards IP fragments, it changes all fragments to have the TTL of the first fragment. As shown in Figure 3, we send a SYN packet broken into two fragments, with the first fragment having a full TTL and second fragment a reduced TTL. We repeat this step while assigning the second packet increasing TTL values. Since the first fragment would be buffered at the TSPU device, if the second fragment arrives at the TSPU device there before its TTL expires, both of them will be forwarded with full TTL and we will receive a response from the server.

On April 26, 2022, we queried Censys [45] for RU hosts with open TCP ports, and we selected IPs from the most popular ten ports. In total, this gives us 4,005,138 unique endpoints (IP:Port). We did not perform NMap scans as we will only send fragmented TCP packets with random payloads and "SYN" flag set from our measurement machines, and we believe the potential risk in doing so is minimal. For each endpoint, we conduct the fragmentation measurement, and for each target that shows TSPU-like fragmentation behavior, we locate the device by iteratively sending TTL-limited fragments, as described above. We accompany each test with a TCP SYN traceroute to record the IPs of the "TSPU link" (*i.e.*, between the last hop where we do not observe TSPU behaviors and the first hop that we do).

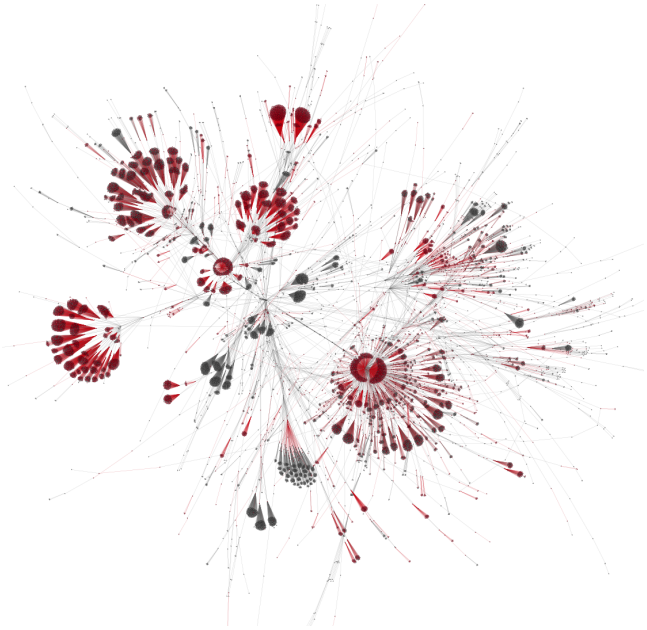

Figure 10: 25K Traceroutes visualization—TSPU links are colored red to show their position with respect to destination IP.

7.3 Results

Limitations: We note that the results and estimates we provide in this section are likely lower bounds. First, the fragments need to pass through a TSPU device before reaching the destination, which requires the TSPU device to have visibility into downstream traffic. For upstream-only TSPU devices (which we showed in § 7.1.1 are prevalent), the end users are still affected by its blocking, but we are unable to detect it with fragmentation measurements. Also, other DPIs or firewalls on the path may buffer or reassemble fragments before reaching the TSPU. Further, our technique does not measure TSPU devices installed behind NATs, which is a recommended installation location according to Roskomnadzor's letter to ISPs [83]. Finally, our measurement covers only the top 10 ports in Russia.

Figure 9 shows the result broken down by ports. In total, out of 4,005,138 endpoints from 4,986 ASes that we checked, 1,013,600 (25.31%) endpoints from 650 ASes showed TSPU-like IP fragmentation behavior, suggesting that this new TSPU system has already achieved significant deployment within Russia. We note that endpoints on port 7547 exhibit the highest number of TSPU-like behaviors, and compared to "server" ports like 80, 22, or 443, hosts with port 7547 open are over 300% more likely to have a TSPU device on path. Note that this does not mean some ports are specifically targeted by the censor; rather, we hope that different open ports can be correlated with the types of the networks where the hosts reside. TCP port 7547 is used by TR-069, a protocol that provides remote management and communication between Customer Premises Equipment (CPE) devices like home routers and ISPs' configuration servers. Since the protocol is mostly used only by residential ISPs, this corroborates existing evidence that the TSPU specifically target residential networks while data centers may be exempted from it. In addition, we found that while only 12.8% of

all ASes show TSPU-like behaviors, the majority of affected ASes have mid-to-large networks. For example, among the 85 ASes that we have at least 5,000 testing targets in, over 75% of them contain endpoints that are behind TSPU installations.

For each testing endpoint, we use traceroute and TTL-limited fragments to locate the the IPs of the hops before and after the TSPU device ("TSPU links"). We cluster these TSPU links based on IP address. For TSPU links that connect leaf nodes, we cluster them based only on the IP of the hop before. We did not perform alias resolution because we do not know if the TSPU device acts as a router, acts as an in-path device on specific interfaces of a router, or is configured in some other way. Moreover, we are interested in paths from our measurement machines, not a full topology. For these reasons, we base our results on the raw traceroute data. From over one million traceroutes that suggest TSPU installations on network paths, we identified 6,871 unique TSPU links. We believe this number is a gross underestimation on how many TSPU devices have been actually deployed, since we only identify the TSPU devices that are, against Roskomnadzor's recommendation [83], outside a NAT. Figure 10 shows a sample of 25,000 traceroutes with TSPU links colored red to show their position with respect to leaf nodes. Figure 12 in Appendix plots a histogram of the number of hops TSPU devices are from destination IPs, which shows for over 69% cases the TSPU link is within the first two hops from the destination IP. Both plots suggest that TSPU devices are located closer to network leaves (possibly end users) than to border or backbone networks. We highlight this is different from the deployment location of the GFW as noted in previous work [42, 97], which measured the GFW in large IXPs, regional gateways, or close to the border.

Finally, we have seen cases where traffic to endpoints from small ISPs goes through TSPU devices that are installed in their upstream providers. Figure 11 shows one such example, where traffic to AS207967 Anton Mamaev as well as three other small ISPs in Tyumen city pass through a TSPU link installed inside AS12389 Rostelecom. Barring the possibility that additional TSPU devices are installed further down the path, this likely shows that small ISPs sometimes rely on TSPUs installed by their upstream providers to be compliant with the requirement [2].

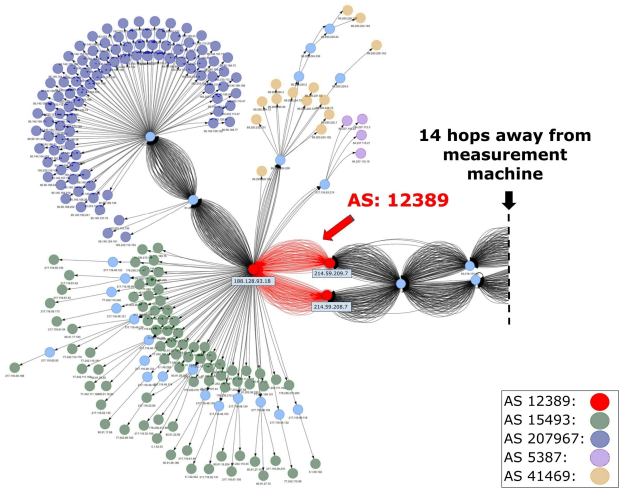

Figure 11: Traceroutes for three ISPs in Tyumen city—TSPU devices are installed inside Rostelecom (red links), which seems to provide "censorship-as-a-service" to downstream ISPs [2].

8 CIRCUMVENTION

Based on what we learned about the TSPU, we discovered several strategies at various layers of the protocol stack that provide means for censorship circumvention, such as fragmentation at the IP layer or Split Handshake at the TCP layer. We broadly classify them as either Server-side strategies that can be deployed solely by the server, or Client-side strategies that require modification to the application or operating system on the client side.

As with all research on censorship circumvention, there is a risk that our work detailing ways to bypass the TSPU will lead to censors patching their system against circumvention strategies described here. We offered potential server-side strategies to sites that were affected by TSPU blocking in March 2022 while placing our report under embargo. However, considering how widely deployed the TSPU is, its technical sophistication, and the existing difficulty that circumvention techniques have of getting deployed or disseminated without the censors learning about them, we follow the same policy of previous work in this area [36–39, 98] and report full details here about packet sequences that evade the TSPU.

Server-side Strategies We identified two server-side circumvention strategies that require no client-side modification. First, servers can announce a smaller TCP window size within the SYN/ACK packet of the initial handshake. By doing so, the client-side TCP stack would divide the ClientHello into multiple TCP segments. This strategy was previously used by brdgrd [4] to break the cipherlist of Tor's ClientHello to avoid detection by China's GFW. Choosing an initial window size may be tricky: we have seen some ISPs that filter out SYN packets with certain small window sizes, but less is known about whether window sizes in SYN/ACKs are filtered as well. In addition, servers can remove the ACK flag from the SYN/ACK packet of the initial handshake. This results in the so-called "Split Handshake" [87]. Specifically, an unmodified client sends a SYN to the server, and the server, instead of responding with the usual SYN/ACK, removes the ACK flag and just sends a SYN. The unmodified client, upon receiving the SYN, would send a SYN/ACK back to the server, and the server can ACK to finish the handshake. This is one of the "green" sequences from Figure 4, and this strategy works for sites targeted by SNI-I. Two strategies can even be combined: the server sends SYN in response to client's SYN, and announce a smaller window size in the ACK that precedes the ClientHello.

Previous work has explored server-side censorship circumvention in other countries [37]. While server-side strategies can help non tech-savvy users to bypass censorship with unmodified software, there are a few issues that can limit their usefulness. First, server-side strategies will not help clients bypass DNS-based censorship, such as the blockpages implemented by Russian ISPs. Second, even though these strategies look simple, as previously discussed, site operators may not be able to deploy them. Finally, as shown in § 7.1.1, server-side strategies may not be effective against middleboxes that have only partial visibility into users' traffic, as server-sent packets may not be routed through them. For example, sites

targeted by SNI-II can still be blocked even with the Split Handshake strategy, due to the existence of an upstream-only TSPU device on the path.

Client-side Strategies We found that some client-side strategies discovered in previous work [98] have been mitigated. Specifically, sending a TTL-limited random-looking packets no longer prevents the following ClientHello from triggering the TSPU. Apparently, the "inspection window" has been extended to cover packets that come later in the session. IP fragmentation or TCP segmentation still help bypass the TSPU when the ClientHello is broken across more than one packet, as well as modifications to the triggering ClientHello (*e.g.*, padding extension, or prepending the ClientHello with another TLS record).

There are a few other potential circumvention approaches. For example, based on the fact that TSPU devices have a short timeout for the SYN-SENT state, one strategy could be simply having the server wait until the entry that tracks the connection at the TSPU is evicted before responding (by then the connection would look like server-initiated). Figure 4 also suggests a few other TCP packet sequences that lead to circumvention. However, we note that these approaches either introduce large delay or require modification on both clients and servers. Thus, we do not discuss them further due to their limited usability/deployability.

The TSPU could easily "patch" these evasion strategies (server-side or client-side), assuming it is provisioned with enough computation and memory resources. Handling Simultaneous Open or Split Handshake simply requires reasoning about the roles of "Client" and "Server" in a more ad-hoc way. TCP flow reassembly is a standard feature for today's DPIs, though it comes with a significantly higher requirement for resources. The server-side reduced window size strategy could be countered with a simple restriction that filters servers' advertised flow control windows. While we do not know if TSPU devices can handle additional computation and storage requirements, the fact that they do not reassemble TCP streams—a capability that the Great Firewall has had since at least 2013 [37, 90, 94]—, and the substantial number of TSPU deployments both suggest that Russia's TSPU is potentially making a trade-off to use several low-cost, commodity hardware boxes and putting them close to users, at the expense of being less able to pool resources to resist evasion attempts.

9 DISCUSSION AND CONCLUSION

Our investigation suggests pervasive deployment of TSPU devices, which constitute a critical part of the technical stack of the RuNet's provision. While commodity filters and DPIs were originally adopted for purposes such as caching and security, they are increasingly being used to actively interfere with users' traffic. The TSPU progressed from ideation to deployment in three years, and we show that their deployment has delivered fine-grained information control to the hands of the Russian government. Our results reveal Russia has made a conscious decision to place them in residential networks and close to end users. Such pervasive deployment of in-path middleboxes may have far-reaching security implications beyond censorship: compared to on-path systems (*e.g.* the GFW [42]) or centrally-deployed in-path systems (*e.g.* the Great Cannon, Sandvine PacketLogic, or Kazakhstan's MITM system [3, 58, 80]), in-path middleboxes deployed in decentralized networks on the same side of CG-NAT as users, like the TSPU, are much better suited for targeted surveillance and machine-in-the-middle attacks. Additionally, while our work focuses primarily on Russia, we fear the TSPU may become a blueprint for other countries to follow, especially considering that Russia has a record of exporting censorship techniques. Researchers should be on alert for the possibility that TSPU devices may be used for more active network interference beyond censorship, as well as their deployment in other countries.

10 ACKNOWLEDGMENT

The authors are grateful to Georgios Smaragdakis for shepherding the paper, and to the anonymous reviewers for their constructive feedback. We also thank Renuka Kumar, Reethika Ramesh, David Fifield, and Andrey Viktorov for insightful discussions. This material is based upon work supported by the National Science Foundation under Grant No.2141547, 2042795, 2007741, 2141512, and DRL SLMAQM20GR2132.

REFERENCES

[1] Academics: Russia deployed new technology to throttle twitter's traffic. https://therecord.media/academics-russia-deployed-new-technology-to-throttle-twitters-traffic/.
[2] Approved requirements for the procedure for passing traffic in data networks and in the public telephone network. https://d-russia.ru/utverzhdeny-trebovanija-k-porjadku-propuska-trafika-v-setjah-peredachi-dannyh-i-v-telefonnoj-seti-svjazi-obshhego-polzovanija.html.
[3] Bad traffic: Sandvine's packetlogic devices used to deploy government spyware in turkey and redirect egyptian users to affiliate ads? https://citizenlab.ca/2018/03/bad-traffic-sandvines-packetlogic-devices-deploy-government-spyware-turkey-syria/.
[4] brdgrd (bridge guard). https://github.com/NullHypothesis/brdgrd.
[5] Cisco asa. https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/ipaddr_nat/configuration/15-mt/nat-15-mt-book.pdf.
[6] Cisco asa series syslog messages. https://www.cisco.com/c/en/us/td/docs/security/asa/syslog/b_syslog/syslog-messages-201002-to-219002.html.
[7] Established facts of dissemination of false information in the media. https://rkn.gov.ru/news/rsoc/news74112.htm.
[8] Evading sni filtering in india with geneva. https://geneva.cs.umd.edu/posts/india-sni-filtering/.
[9] Freebsd. https://github.com/freebsd/freebsd-src/tree/main/sys/netpfil.
[10] huawei. https://support.huawei.com/enterprise/en/doc/EDOC1100058384/5072245d/setting-the-aging-time-for-nat-session-entries.
[11] Internet shutdowns in 2021 report: India is the world's largest offender. https://www.accessnow.org/internet-shutdowns-india-keepiton-2021/.
[12] Internet shutdowns in 2021: the return of digital authoritarianism. https://www.accessnow.org/internet-shutdowns-2021/.
[13] Juniper. https://docs.128technology.com/docs/concepts_session_timer/.
[14] Juniper network - fragment-limit. https://www.juniper.net/documentation/us/en/software/junos/interfaces-adaptive-services/topics/ref/statement/fragment-limit-edit-interfaces.html.
[15] The linux kernel archives. https://www.kernel.org/doc/Documentation/networking/ip-sysctl.txt.
[16] Linux nat timers. https://www.kernel.org/doc/Documentation/networking/nf_conntrack-sysctl.txt.
[17] On amendments to federal law on protecting children from information harmful to their health and the development and certain legislative acts of the russian federation. https://web.archive.org/web/20140208154037/http://eng.kremlin.ru/acts/4246.
[18] On recommendations of roskomnadzor to telecom operators on blocking illegal information on the internet. https://archive.fo/LGszb;https://archive.fo/XAgJk.
[19] Ooni measurement aggregation toolkit. https://explorer.ooni.org/chart/mat?probe_cc=RU&test_name=web_connectivity&domain=play.google.com&since=2022-03-01&until=2022-05-13&axis_x=measurement_start_day.
[20] The president signed the law on sustainable runet. https://d-russia.ru/prezident-podpisal-zakon-ob-ustojchivom-runete.html.
[21] Register of internet addresses filtered in russian federation resources (github). https://github.com/zapret-info/z-i.
[22] Registry of banned sites. https://blocklist.rkn.gov.ru.

[23] Response measures taken to restrict access to russian media. https://rkn.gov.ru/news/rsoc/news74156.htm.

[24] We chat, they watch: How international users unwittingly build up wechat's chinese censorship apparatus. https://citizenlab.ca/2020/05/we-chat-they-watch/.

[25] Windows nat timers. https://docs.microsoft.com/en-us/azure/virtual-network/nat-gateway/nat-gateway-resource.

[26] Internet Protocol. RFC 791, Sept. 1981. https://www.rfc-editor.org/info/rfc791.

[27] Reassessing runet: Russian internet isolation and implications for russian cyber behavior, 2021. https://www.atlanticcouncil.org/in-depth-research-reports/issue-brief/reassessing-runet-russian-internet-isolation-and-implications-for-russian-cyber-behavior/.

[28] Russia tests way to disconnect from worldwide internet., 2021. https://learningenglish.voanews.com/a/russia-tests-way-to-disconnect-from-worldwide-internet/5976331.html.

[29] Russia throttled twitter to censor content — here's what happens next, 2021. https://www.accessnow.org/russia-throttled-twitter/.

[30] Order of the ministry of digital development, telecommunications and mass media of the russian federation dated january 26, 2022 no. 44 "on approval of the requirements for the procedure for passing traffic in data networks", 2022. http://publication.pravo.gov.ru/Document/View/0001202203010002.

[31] Voa condemns russian attempts to interfere in free flow of information, 2022. https://www.newsweek.com/voa-condemns-russian-attempts-interfere-free-flow-information-1684384.

[32] Alice, Bob, Carol, J. Beznazwy, and A. Houmansadr. How china detects and blocks shadowsocks. In *Proceedings of the ACM Internet Measurement Conference*, IMC '20, page 111–124, New York, NY, USA, 2020. Association for Computing Machinery.

[33] C. Anderson. Dimming the internet: Detecting throttling as a mechanism of censorship in iran, 2013. https://arxiv.org/abs/1306.4361.

[34] S. Aryan, H. Aryan, and J. A. Halderman. Internet censorship in iran: A first look. In *FOCI*, 2013.

[35] D. M. Blei, A. Y. Ng, M. I. Jordan, and J. Lafferty. Latent dirichlet allocation. *Journal of Machine Learning Research*, 2003.

[36] K. Bock, Y. Fax, K. Reese, J. Singh, and D. Levin. Detecting and evading censorship-in-depth: A case study of iran's protocol whitelister. In *FOCI @ USENIX Security Symposium*, 2020.

[37] K. Bock, G. Hughey, L.-H. Merino, T. Arya, D. Liscinsky, R. Pogosian, and D. Levin. Come as you are: Helping unmodified clients bypass censorship with server-side evasion. pages 586–598, 07 2020.

[38] K. Bock, G. Hughey, X. Qiang, and D. Levin. Geneva: Evolving censorship evasion strategies. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, CCS '19, page 2199–2214, New York, NY, USA, 2019. Association for Computing Machinery.

[39] K. Bock, G. Naval, K. Reese, and D. Levin. Even censors have a backup: Examining china's double https censorship middleboxes. In *Proceedings of the ACM SIGCOMM 2021 Workshop on Free and Open Communications on the Internet*, FOCI '21, page 1–7, New York, NY, USA, 2021. Association for Computing Machinery.

[40] Z. Chai, A. Ghafari, and A. Houmansadr. On the importance of encrypted-SNI (ESNI) to censorship circumvention. In *USENIX Workshop on Free and Open Communications on the Internet (FOCI)*, 2019.

[41] R. Clayton, S. J. Murdoch, and R. N. M. Watson. Ignoring the great firewall of china. In G. Danezis and P. Golle, editors, *Privacy Enhancing Technologies*, pages 20–35, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[42] J. Crandall, D. Zinn, M. Byrd, E. Barr, and R. East. Conceptdoppler: A weather tracker for internet censorship. pages 352–365, 01 2007.

[43] J. Dalek, B. Haselton, H. Noman, A. Senft, M. Crete-Nishihata, P. Gill, and R. J. Deibert. A method for identifying and confirming the use of url filtering products for censorship. In *Proceedings of the 2013 Conference on Internet Measurement Conference*, IMC '13, page 23–30, New York, NY, USA, 2013. Association for Computing Machinery.

[44] D. Dittrich, E. Kenneally, et al. The Menlo Report: Ethical principles guiding information and communication technology research. Technical report, U.S. Department of Homeland Security, 2012.

[45] Z. Durumeric, D. Adrian, A. Mirian, M. Bailey, and J. A. Halderman. Censys: A search engine backed by Internet-wide scanning. In *ACM Conference on Computer and Communications Security (CCS)*, 2015.

[46] Z. Durumeric, E. Wustrow, and J. A. Halderman. ZMap: Fast internet-wide scanning and its security applications. In *USENIX Security Symposium*, 2013.

[47] R. Ensafi, D. Fifield, P. Winter, N. Feamster, N. Weaver, and V. Paxson. Examining how the great firewall discovers hidden circumvention servers. IMC '15, page 445–458, New York, NY, USA, 2015. Association for Computing Machinery.

[48] R. Ensafi, P. Winter, A. Mueen, and J. Crandall. Analyzing the great firewall of china over space and time. *Proceedings on Privacy Enhancing Technologies*, 1, 04 2015.

[49] B. Ford, S. Guha, K. Biswas, S. Sivakumar, and P. Srisuresh. NAT Behavioral Requirements for TCP. RFC 5382, Oct. 2008. https://www.rfc-editor.org/info/rfc5382.

[50] FortiNet. Fortiguard labs web filter. https://fortiguard.com/webfilter.

[51] Google. inet fragments management. https://android.googlesource.com/kernel/msm/+/refs/heads/android-msm-redbull-4.19-android11-qpr3/net/ipv4/inet_fragment.c.

[52] N. P. Hoang, A. A. Niaki, J. Dalek, J. Knockel, P. Lin, B. Marczak, M. Crete-Nishihata, P. Gill, and M. Polychronakis. How great is the great firewall? measuring china's dns censorship. In *USENIX Security Symposium*, 2021.

[53] B. Jones, T.-W. Lee, N. Feamster, and P. Gill. Automated detection and fingerprinting of censorship block pages. In *ACM Internet Measurement Conference (IMC)*, 2014.

[54] kelmenhorst/quic censorship. Broad blocking of http/3 traffic in russia (as31213, as12389). https://github.com/kelmenhorst/quic-censorship/issues/4.

[55] J. Knockel and L. Ruan. Measuring qqmail's automated email censorship in china. In *Proceedings of the ACM SIGCOMM 2021 Workshop on Free and Open Communications on the Internet*, FOCI '21, page 8–15, New York, NY, USA, 2021. Association for Computing Machinery.

[56] S. Krishnan. Handling of Overlapping IPv6 Fragments. RFC 5722, Dec. 2009. https://www.rfc-editor.org/info/rfc5722.

[57] C. Lab and Others. Url testing lists intended for discovering website censorship, 2014. https://github.com/citizenlab/test-lists.

[58] B. Marczak, N. Weaver, J. Dalek, R. Ensafi, D. Fifield, S. McKune, A. Rey, J. Scott-Railton, R. Deibert, and V. Paxson. An analysis of China's "Great Cannon". In *5th USENIX Workshop on Free and Open Communications on the Internet (FOCI 15)*, Washington, D.C., Aug. 2015. USENIX Association.

[59] meduza. The day the news died here are all russia's independent media outlets banned, blocked, or shuttered in just the past few days. https://meduza.io/en/short/2022/03/05/the-day-the-news-died.

[60] net4people. Blocking of http/3 (quic) in russia. https://github.com/net4people/bbs/issues/108.

[61] net4people. Blocking of news websites in russia. https://github.com/net4people/bbs/issues/107.

[62] net4people. Partial blocking or throttling of social media in russia. https://github.com/net4people/bbs/issues/106.

[63] A. A. Niaki, S. Cho, Z. Weinberg, N. P. Hoang, A. Razaghpanah, N. Christin, and P. Gill. Iclab: A global, longitudinal internet censorship measurement platform, 2019. https://arxiv.org/abs/1907.04245.

[64] ntc.party. Blocking static content youtube yt3.ggpht.com on tspu. https://ntc.party/t/youtube-yt3-ggpht-com/1784/13.

[65] ntc.party. Blocking torguard. https://ntc.party/t/torguard/2154.

[66] ntc.party. Blocking/slowing down twitter. https://ntc.party/t/twitter-26-02-2022/1724/30.

[67] ntc.party. Facebook limit. https://ntc.party/t/facebook/1722.

[68] ntc.party. Http/3 (quic) restriction. https://ntc.party/t/http-3-quic/1823/10.

[69] ntc.party. news.google.com blocked and play.google.com unavailable. https://ntc.party/t/news-google-com-play-google-com/2071/7.

[70] ntc.party. Nordvpn is not available. https://ntc.party/t/nordvpn/1936.

[71] ntc.party. Ooni reports of tor blocking in certain isps since 2021-12-01. https://ntc.party/t/ooni-reports-of-tor-blocking-in-certain-isps-since-2021-12-01/1477/145.

[72] OONI. 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/.

[73] R. Padmanabhan, A. Filastò, M. Xynou, R. S. Raman, K. Middleton, M. Zhang, D. Madory, M. Roberts, and A. Dainotti. A multi-perspective view of internet censorship in myanmar. FOCI '21, page 27–36, New York, NY, USA, 2021. Association for Computing Machinery.

[74] J. C. Park and J. R. Crandall. Empirical study of a national-scale distributed intrusion detection system: Backbone-level filtering of HTML responses in china. In *International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2010.

[75] P. Pearce, R. Ensafi, F. Li, N. Feamster, and V. Paxson. Augur: Internet-wide detection of connectivity disruptions. In *IEEE Symposium on Security and Privacy (S&P)*, 2017.

[76] P. Pearce, B. Jones, F. Li, R. Ensafi, N. Feamster, N. Weaver, and V. Paxson. Global measurement of DNS censorship. In *USENIX Security Symposium*, 2017.

[77] P. Pearce, B. Jones, F. Li, R. Ensafi, N. Feamster, N. Weaver, and V. Paxson. Global measurement of DNS manipulation. In *26th USENIX Security Symposium (USENIX Security 17)*, pages 307–323, Vancouver, BC, Aug. 2017. USENIX Association.

[78] R. Penno, S. Perreault, M. Boucadair, S. Sivakumar, and K. Naito. Updates to Network Address Translation (NAT) Behavioral Requirements. RFC 7857, Apr. 2016. https://www.rfc-editor.org/info/rfc7857.

[79] V. L. Pochat, T. V. Goethem, S. Tajalizadehkhoob, M. Korczynski, and W. Joosen. Tranco: A research-oriented top sites ranking hardened against manipulation. In *Proceedings 2019 Network and Distributed System Security Symposium*. Internet Society, 2019.

[80] R. S. Raman, L. Evdokimov, E. Wurstrow, J. A. Halderman, and R. Ensafi. Investigating large scale https interception in kazakhstan. In *Proceedings of the ACM*

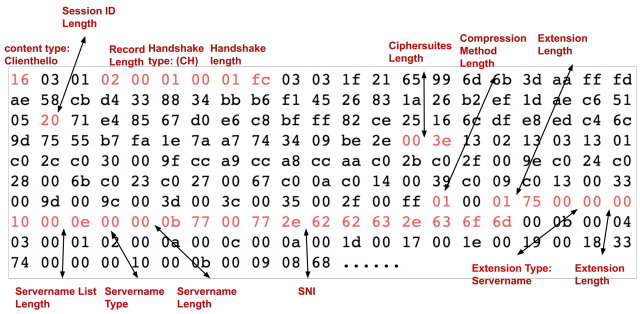

Figure 13: A Clienthello that triggers TSPU's SNI-based censorship.

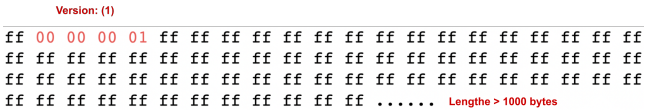

Figure 14: A UDP packet that triggers TSPU's QUIC censorship.

Internet Measurement Conference, IMC '20, page 125–132, New York, NY, USA, 2020. Association for Computing Machinery.

[81] R. Ramesh, R. Sundara Raman, M. Bernhard, V. Ongkowijaya, L. Evdokimov, A. Edmundson, S. Sprecher, M. Ikram, and R. Ensafi. Decentralized control: A case study of Russia. In *Network and Distributed Systems Security Symposium (NDSS)*, 2020. https://censoredplanet.org/assets/russia.pdf.

[82] RDP. Ecosge user guide, 2021. https://www.rdp.ru/wp-content/uploads/ENAT_UserGuide_EN.pdf.

[83] Report about roskomnadzor's letter to isps, 2019. https://habr.com/ru/post/459894/.

[84] R. Sundara Raman, P. Shenoy, K. Kohls, and R. Ensafi. Censored Planet: An Internet-wide, Longitudinal Censorship Observatory. In *ACM Conference on Computer and Communications Security (CCS)*, 2020.

[85] R. Sundara Raman, A. Stoll, J. Dalek, A. Sarabi, R. Ramesh, W. Scott, and R. Ensafi. Measuring the deployment of network censorship filters at global scale. In *Network and Distributed System Security Symposium (NDSS)*, 2020.

[86] techcrunch. Instagram is now blocked in russia. https://techcrunch.com/2022/03/14/instagram-is-now-blocked-in-russia/.

[87] B. Tod and Q. Jin. The tcp split handshake: Practical effects on modern network equipment. *Network Protocols and Algorithms*, 2, 05 2010.

[88] M. C. Tschantz, S. Afroz, Anonymous, and V. Paxson. SoK: Towards grounding censorship circumvention in empiricism. In *Symposium on Security & Privacy*. IEEE, 2016. https://internet-freedom-science.org/circumvention-survey/sp2016/.

[89] B. VanderSloot, A. McDonald, W. Scott, J. A. Halderman, and R. Ensafi. Quack: Scalable remote measurement of application-layer censorship. In *USENIX Security Symposium*, 2018.

[90] Z. Wang, Y. Cao, Z. Qian, C. Song, and S. V. Krishnamurthy. Your state is not mine: A closer look at evading stateful internet censorship. In *Proceedings of the 2017 Internet Measurement Conference*, IMC '17, page 114–127, New York, NY, USA, 2017. Association for Computing Machinery.

[91] V. Weber. The worldwide web of chinese and russian information controls. https://public.opentech.fund/documents/English_Weber_WWW_of_Information_Controls_Final.pdf.

[92] Z. Weinberg, D. Barradas, and N. Christin. Chinese wall or swiss cheese? In *Proceedings of the Web Conference 2021*, WWW '21, page 472–483, New York, NY, USA, 2021. Association for Computing Machinery.

[93] Z. Weinberg, M. Sharif, J. Szurdi, and N. Christin. Topics of controversy: An empirical analysis of web censorship lists. *PoPETs*, 2017(1):42–61, 2017.

[94] P. Winter. How the great firewall of china is blocking tor. https://www.cs.kau.se/philwint/gfw/.

[95] P. Winter and S. Lindskog. How china is blocking tor, 2012. https://arxiv.org/abs/1204.0447.

[96] S. Wolfgarten. Investigating large-scale internet content filtering. 2005.

[97] X. Xu, Z. M. Mao, and J. A. Halderman. Internet censorship in china: Where does the filtering occur? In N. Spring and G. F. Riley, editors, *Passive and Active Measurement*, pages 133–142, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[98] D. Xue, R. Ramesh, ValdikSS, L. Evdokimov, A. Viktorov, A. Jain, E. Wustrow, S. Basso, and R. Ensafi. Throttling twitter: An emerging censorship technique in russia. In *Proceedings of the 21st ACM Internet Measurement Conference*, IMC '21, page 435–443, New York, NY, USA, 2021. Association for Computing Machinery.

[99] P. Zhu, K. Man, Z. Wang, Z. Qian, R. Ensafi, J. A. Halderman, and H. Duan. Characterizing transnational internet performance and the great bottleneck of china. *Proc. ACM Meas. Anal. Comput. Syst.*, 4(1), may 2020.

[100] J. Zittrain and B. Edelman. Internet filtering in China. *IEEE Internet Computing*, 2003.

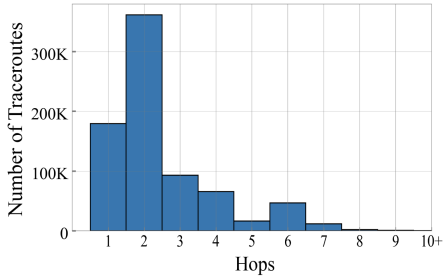

Figure 12: Number of hops that TSPU devices are away from destination IPs.

A SNI-BASED AND QUIC BLOCKING TRIGGERS

Figure 13 shows which parts of a ClientHello packet are inspected by the TSPU. Specifically, we find that the TSPU parses a ClientHello in order to locate the SNI field, rather than matching the targeted domain strings over entire packets. The TSPU ignores other TLS extensions.

Figure 14 shows a minimum "fingerprint" that the TSPU uses to detect QUIC packets. Specifically, the fingerprint is based on QUIC's plaintext "version" field (0x00, 0x00, 0x00, 0x01 starting from the second byte) within the protocol's header, and it applies the filter for any UDP packet destined to port 443 that has at least 1001 bytes in its payload [68].

B TCP STATES TIMEOUTS

Table **??** summarizes the timeout value estimates for our state enumeration experiments. We found a total of four unique timeout values, none of which match known connection tracking implementations. This is a lower bound as some states could share the same timeout value. Table 7 lists the state timeout values for common OSes and specifications.

OS/Spec	Connection State	Timeout
rdp	timeout_inactivity translation	86400
rdp	timeouts_inactivity tcp_handshake	4
rdp	timeouts_inactivity tcp_active	300
rdp	timeouts_inactivity tcp_final	240
rdp	timeouts_inactivity tcp_reset	4
rdp	timeouts_inactivity tcp_session_active	120
freebsd	tcp.first	120
freebsd	tcp.opening	30
freebsd	tcp.established	86400
freebsd	tcp.closing	900
freebsd	tcp.finwait	45
freebsd	tcp.closed	90
windows	TCP FIN	60
windows	TCP RST	10
windows	TCP half open	30
windows	TCP idle timeout	240
linux	syn_sent	120
linux	syn_recv	60
linux	established	432000
linux	time_wait	120
linux	unacknowledged	300
linux	last_ack	30
linux	fin_wait	120
linux	close	10
linux	close_wait	60
rfc 5382	half open	240
rfc 5382	established idle	7200
rfc 5382	TIME WAIT	240
rfc 7857	partial open idle timeout	240
Huawei	TCP session aging time	600
Cisco	Tcp-timeout	86400
Juniper	TCP session timeout	1800

Table 7: Timeout values for TCP states for open and closed source connection tracking systems. RDP [82], FreeBSD [9], Windows [25], Linux [16],rfc 5382 [49], rfc 7857 [78],Huewei [10], Cisco [5], Juniper [13]