Network Measurement Methods for Locating and Examining Censorship Devices

Ram Sundara Raman*
University of Michigan
ramaks@umich.edu

Mona Wang*
Princeton University
monaw@princeton.edu

Jakub Dalek
The Citizen Lab
jakub@citizenlab.ca

Jonathan Mayer
Princeton University
jonathan.mayer@princeton.edu

Roya Ensafi
University of Michigan
ensafi@umich.edu

ABSTRACT

Advances in networking and firewall technology have led to the emergence of network censorship devices that can perform large-scale, highly-performant content blocking. While such devices have proliferated, techniques to locate, identify, and understand them are still limited, require cumbersome manual effort, and are developed on a case-by-case basis.

In this paper, we build robust, general-purpose methods to understand various aspects of censorship devices, and study devices deployed in 4 countries (Azerbaijan, Belarus, Kazakhstan, and Russia). We develop a censorship traceroute method, CenTrace, that automatically identifies the network location of censorship devices. We use banner grabs to identify vendors from potential censorship devices. To collect more features about the devices themselves, we build a censorship fuzzer, CenFuzz, that uses various HTTP request and TLS Client Hello fuzzing strategies to examine the rules and triggers of censorship devices. Finally, we use features collected using these methods to cluster censorship devices and explore device characteristics across deployments.

Using CenTrace measurements, we find that censorship devices are often deployed in ISPs upstream to clients, sometimes even in other countries. Using data from banner grabs and injected blockpages, we identify 23 commercial censorship device deployments in Azerbaijan, Belarus, Kazakhstan, and Russia. We observe that certain CenFuzz strategies such as using a different HTTP method succeed in evading a large portion of these censorship devices, and observe that devices manufactured by the same vendors have similar evasion behavior using clustering. The methods developed in this paper apply consistently and rapidly across a wide range of censorship devices and enable continued understanding and monitoring of censorship devices around the world.

CCS CONCEPTS

• **General and reference** → **Measurement**; • **Social and professional topics** → *Technology and censorship.*

*Both authors contributed equally to this research.

CoNEXT '22, December 6–9, 2022, Roma, Italy

ACM ISBN 978-1-4503-9508-3/22/12.
https://doi.org/10.1145/3555050.3569133

KEYWORDS

Measurement, Censorship, Network Fingerprinting

ACM Reference Format:
Ram Sundara Raman, Mona Wang, Jakub Dalek, Jonathan Mayer, and Roya Ensafi. 2022. Network Measurement Methods for Locating and Examining Censorship Devices. In *The 18th International Conference on emerging Networking EXperiments and Technologies (CoNEXT '22), December 6–9, 2022, Roma, Italy*. ACM, New York, NY, USA, 17 pages. https://doi.org/10.1145/3555050.3569133

1 INTRODUCTION

Recent years have witnessed censorship and surveillance events of unprecedented scale, such as the HTTPS interception attack of popular domains in Kazakhstan [60], and the throttling and censorship of social media domains in Russia [50, 79]. These events are enabled by the proliferation of *censorship devices*, software or hardware deployed on the network that inspects connections with the goal of filtering access to undesired content. Today, these devices have the ability to inspect large amounts of network traffic and enact fine-grained interference. Advances in networking devices and the commoditization of deep packet inspection (DPI) techniques have made this censorship capability increasingly available for governments and ISPs [62].

Most research on censorship measurement so far has focused on detecting which websites are blocked on which protocols [3, 61, 63] and how the censorship can be circumvented [11, 33, 67, 75]. While the emergence of censorship measurement platforms such as OONI [63] and Censored Planet [61] has provided the censorship measurement community with accurate, reliable and scalable tools and data to understand website reachability, general-purpose solutions for studying the devices that perform censorship themselves are lacking. Due to the opaque nature of censorship, the variety of devices and censorship methods, and the lack of transparency by vendors, collecting features about censorship devices, such as their network location, rules and triggers, and capabilities requires herculean manual effort.

Because of these challenges, previous work has focused on studying *specific* devices and censorship deployments. While some studies have focused on identifying specific device manufactures using visible signatures [16, 17, 44, 46, 62], others have developed ad-hoc methods to explore the network location of specific censorship systems [45, 60, 77, 79], and their rules and triggers [38, 41]. While these studies have been instrumental in shedding light on network devices that perform censorship and have helped deliver major

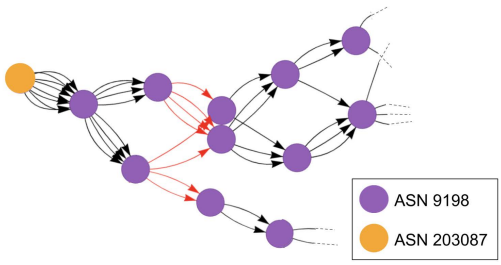

Figure 1: CenTrace measurements from a client in KZ. Red links indicate location of blocking ⋄

policy change [70, 81], they rely on distinct characteristics of certain censorship systems, require large amounts of manual effort, and do not scale across devices. Research on censorship devices continues to focus on a small set of well-known cases (such as the Great Firewall [36, 45]), and there still exists no rigorous, reusable, and scalable methods for us to locate censorship devices and collect features about them.

In this paper, we build robust, general-purpose methods and tools to answer three primary research questions regarding censorship devices: (*1*) Where are censorship devices located in the network? (*2*) What are their blocking rules and triggers? and (*3*) Which commercial firewall software are used for censorship, and which characteristics are similar across deployments? We show the capability of our methods to answer these questions by collecting both in-country and remote measurement data using our methods in four countries with regional and economic ties whose censorship systems have been under the scrutiny due to recent events: Azerbaijan (AZ), Kazakhstan (KZ), Belarus (BY), and Russia (RU) [1, 64].

To understand the network location of censorship devices, we design and build a censorship traceroute tool, CenTrace, that uses traceroute-like TTL-limited measurements to automatically identify where HTTP(S) blocking occurs, without any information about what censorship devices are used or how the censorship is implemented. By varying both packet TTL and request content, we build network paths and identify where censorship occurs on those paths. Our tool accounts for network path variance and considers different types of devices such as ones that are deployed in-path vs on-path, ones that drop packets, and ones that copy TTL values from incoming packets. Using CenTrace, we collect over 12,600 traceroute measurements in AZ, KZ, BY, and RU, among which 1,430 show clear signs of blocking. Figure 1 shows the location of blocking detected by CenTrace as seen from a client inside KZ. We identify that most blocking occurs close to the host inside the country (within 1–10 hops), but sometimes in an upstream ISP. In an area where attribution is difficult and censorship is often reported by the client ASN, our findings shed light on the importance of understanding where censorship occurs. Moreover, we observe apparent extraterritorial imposition of information controls—CenTrace remote measurements to 21.81% of hosts in Kazakhstan are actually blocked in Russia.

While CenTrace paves the way for identifying the location of censorship devices, there is still a need to collect more features about the device itself to learn about the capabilities and characteristics of censorship deployments. We collect HTTP(S), SSH, Telnet, FTP, SMTP, and SNMP banners from 163 potential censorship device IP addresses identified using CenTrace to identify device vendors. Using banner grabs and injected blockpages, we identify 23 commercial filtering devices in AZ, BY, KZ, and RU.

To collect additional features, we design and build a censorship request fuzzing tool, CenFuzz, which identifies the rules and triggers of various censorship devices. We implement 16 HTTP request and 8 TLS Client Hello fuzzing strategies that automatically attempt to evade detection by censorship devices. We perform 221,786 CenFuzz measurements in AZ, BY, KZ, and RU, and observe that certain strategies result in successful evasion. For example, using alternate HTTP Methods other than `GET` (such as `PUT` and `PATCH`) in 0.44%–90.58% of fuzzed requests evade censorship deployments. We identify several strategies that evade the censor but are parsed correctly by web servers, resulting in circumvention.

We use the features collected from CenTrace, CenFuzz and banner grab measurements to cluster censorship device deployments in AZ, BY, KZ, and RU, in order to study their censorship device deployment patterns. We find that devices deployed in the same ISP form tight clusters, indicating that censorship policies are implemented at the ISP level. We also find that certain evasion strategies are more likely to succeed against devices that may be manufactured by the same vendor, which could provide researchers with an efficient method to fingerprint them.

The methods developed in this work are designed to be robust, general-purpose and applicable to a wide variety of censorship devices. We open-source and maintain our CenTrace, CenFuzz and banner grab tools at https://censoredplanet.org/censorship-devices to enable the community to continuously monitor censorship devices. Our techniques can be used to advance the understanding of censorship deployments in countries around the world, and can provide researchers, journalists, policymakers, and Internet freedom advocates the means to ensure accountability for the manufacturers of these devices, and the authorities that deploy them.

2 ETHICAL CONSIDERATIONS

Measurement of Internet censorship requires careful consideration of safety for end-users, as it requires triggering censorship devices situated in the network multiple times. We conduct both in-country and remote measurements of censorship, primarily in four countries—AZ, BY, KZ, and RU. Institutional Review Boards (IRBs) consider Internet measurement studies such as ours as outside their purview, since we do not collect any personally identifying information, and hence we rely on and follow guidelines and safeguards suggested by previous work performing similar measurements [3, 55, 61, 63] and those outlined in the Menlo and Belmont reports [21, 49].

We purchase our in-country vantage points from popular commercial VPS providers, similar to previous work [3, 55]. We confirm that these machines are located in data centers and not in residential networks (i.e. they do not belong to end-users), and we conduct our measurements in accordance with their terms of service. We ensure that we do not cause operators of these platforms any more

risk than they would incur when operating commercial computing services.

For remote measurements to endpoints in these countries, we follow the selection process adopted by the Censored Planet measurement platform [61], and only send measurements to machines that are part of *organizational or ISP infrastructure*. Specifically, we select endpoints for our HTTP and TLS measurements by identifying web servers in these countries that present a valid Extended Validation (EV) TLS certificate or those that host domains in PeeringDB [52]. Typically, only large organizations which receive a significant amount of traffic obtain EV certificates, and their administrators possess the required skills and resources to understand the traffic sent to their services. PeeringDB contains websites of ASes, whose web servers are typically part of an ISP's infrastructure.

Apart from applying these safeguards to our client and endpoint selection process, we also follow best practices in conducting measurements [23, 61]. We set up WHOIS records and reverse DNS pointers on our measurement client and host a web server on ports 80 and 443, all indicating that our measurements are part of a research project, and offer measurement targets the option to opt-out. We did not receive any opt-out requests during our study.

3 BACKGROUND & RELATED WORK

In this section, we provide an overview of work related to censorship measurement, traceroutes, evasion, and identification of censorship devices. We compare and contrast our approach to each of the relevant previous work.

3.1 Censorship Measurement

Website blocking is commonly implemented by a network intermediary, such as a middlebox or DNS resolver deployed in an ISP, that interferes with the DNS request, TCP handshake, TLS handshake or HTTP request [25, 36, 51, 55, 68]. DNS censorship is commonly implemented by ISP resolvers that return errors or incorrect IP addresses for DNS requests corresponding to censored domains [51, 55], rather than by middlebox tampering [5, 6, 36]. As such, we do not study DNS censorship in this paper. IP blocking has seen a decline in adoption over the past few years due to the emergence of CDNs that can serve multiple domains from the same IP address [2, 80]. In contrast, censorship is increasingly performed by network devices blocking TLS handshakes and HTTP requests by inspecting the Client Hello Server Name Indication (SNI) field and the Host Header fields respectively [8, 12, 62, 68]. Devices performing censorship of TLS and HTTP requests often drop packets inducing a timeout, terminate connections using TCP RST messages, or inject a blockpage into plaintext communication. Although previous measurement studies have studied the targets of HTTP(S) blocking in different countries, *very few studies have focused on the devices performing blocking, and such devices are the primary subject of our study.*

Internet censorship practices in a country have been studied through both in-country measurements collected using volunteers or accessible vantage points inside the country [3, 36, 39, 55, 63, 80], and through remote measurements originating from outside the country [25, 51, 60, 62, 68]. These techniques complement each other well. In-country measurements such as those performed by the OONI volunteer network can provide an in-depth analysis of censorship in a country, but are limited in scale [63]. On the other hand, remote measurement platforms such as Censored Planet can scale up measurements by sending requests to public *infrastructural* machines on the Internet i.e. machines that belong to large organizations or the ISP infrastructure itself [61]. In this paper, we perform both in-country measurements as well as large-scale remote measurements for the countries under study, but focus on measuring censorship devices. Currently, measurements performed by OONI and Censored Planet measure reachability to websites from different countries and investigate the cause of blocking. However, they do not have the capability to detect and study the devices that perform censorship. As shown in this paper, integrating the measurement methods proposed in this paper into platforms such as OONI and Censored Planet can significantly improve the accuracy and detail of data produced by these platforms.

3.2 Traceroutes

Traceroute techniques are core to not only network diagnostics, but are essential for modern network and Internet analyses. Recent work has demonstrated various ways to improve traceroutes for modern, load-balanced networks [9, 69]. Tools such as Tracebox utilize the quoted packet returned in ICMP in order to identify middlebox interference [18]. We utilize and extend the methods described in these studies to construct our own censorship traceroute. In censorship studies, traceroute-like techniques have been used to study the location of specific filtering devices, such as those of the Great Firewall, China's "Great Cannon" infrastructure, the location of Kazakhstan's HTTPS interception system [45, 60, 77]. Jin et al. recently proposed Disguiser, a framework that detects censorship by sending requests to a control server that responds with a static payload, and they perform application-layer traceroutes in some cases to explore the deployment of censors [39]. However, all of these studies focused on studying specific censors whose censorship signatures were already known. In this paper, *we build a general-purpose method for identifying the network location of a variety of devices (§4).*

3.3 Censorship Devices

Recent studies have found that many commercial network security and firewall devices are used by network operators to perform censorship in large ISPs [62, 65, 71]. Research to date in identifying network censorship devices has approached the problem on a case-by-case, ad-hoc basis [1, 16, 44, 46, 81]. In prior work, researchers manually engineer network fingerprints for known censorship technologies from features such as the TCP and HTTP headers in order to investigate deployment patterns [17, 73, 74, 76]. For instance, in 2013, Dalek et al. manually created fingerprints for four popular network filter devices, and measured their deployment in several countries [17].

Scaling up such identification has been a long-standing challenge in the community. In 2020, Sundara Raman et al. developed FilterMap, a framework for clustering filters that are configured to censor with user-observable blockpages [62]. FilterMap identified various blockpage clusters in different countries, including blockpages from commercial web filtering products, Government and

ISP blockpages, and organizational blockpages. While FilterMap provides valuable increase in scale, the technique is dependent on censors injecting identifiable blockpages, which is not feasible in encrypted communications. *To resolve these challenges with previous work, we develop banner grab measurements that scan network devices themselves, and utilize clustering to identify devices that might not inject identifiable blockpages (§5).*

There is a wealth of literature on fingerprinting network devices via active probing techniques, such as using Nmap or ZGrab [43, 82]. Researchers have analyzed the utility of various IP packet header fields as well as more specifically crafted application-layer probes like SNMPv3 [4, 66]. More recently, machine learning techniques have been used to refine network device classification [15, 37]. *We utilize and extend these techniques in our work (§7).*

3.4 Censorship Evasion

Previous work on censorship circumvention has focused on modifying censored requests to evade censorship by exploiting idiosyncracies in censorship implementations. Bock et al. developed Geneva, a TCP blocking circumvention tool that utilizes genetic algorithms to optimize the discovery of TCP packet modification circumvention strategies [11]. Li et al. investigate rules that trigger network classifiers and discover methods to automatically evade them [40, 41]. *In contrast to these studies, we do not focus on strategies for circumvention. Our censorship fuzzing tool, CenFuzz, deterministically tests the same, sometimes invalid, requests across all censorship devices (§6).* Most related to our work is Autosonda, an automated fuzzing technique proposed by Jermyn et al. to discover and study the decision models of censorship devices and identify circumvention paths [38]. While their results motivate the use of fuzzing in identifying rules of censorship devices, the study was only conducted in one metropolitan city, and the network features that were studied are limited. For instance, they do not fuzz HTTPS requests. We extend this work by developing a variety of additional strategies for HTTP, and new strategies for fuzzing TLS Client Hello (HTTPS).

4 CENSORSHIP TRACEROUTE

A key challenge in understanding censorship devices is determining their location in the network. This includes identifying the exact IP address or IP ranges where the censorship device is located, enabling understanding of which ISP and country the blocking occurs in. Attributing the location of censorship has been a major limitation in previous work and existing censorship measurement platforms [61, 63], which currently characterize blocking based on the location of the host. This could lead to incorrect reporting of censorship, as the blocking may be occurring in an upstream ISP, maybe even in a different country, instead of the host network.

Determining the location of censorship devices is challenging due to the fact that there are numerous censorship devices with a myriad of characteristics, each of which may be configured differently in the network. While previous work has focused on determining the location of specific censorship devices, they rely on particular characteristics and behavior of the censorship device under study [39, 45, 62, 77]. We develop and implement CenTrace, a general-purpose method for determining censorship device location using TTL-limited probes. We focus on censorship devices

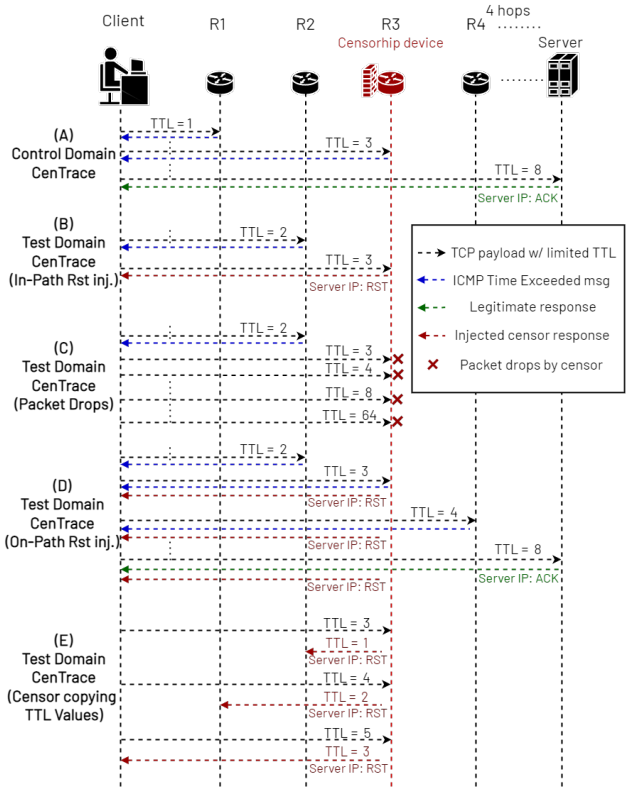

Figure 2: CenTrace operation under different censorship device behaviors ⋄

performing censorship on the HTTP Host header or the SNI extension in the TLS client Hello, as mentioned in §3.1. However, our technique can be easily extended to other protocols such as DNS and SSH.

4.1 Methods

CenTrace uses probe packets with varying TTL and request content in order to build a network path and identify where censorship is occurring on that path. An overview of CenTrace's operation is shown in Figure 2. We first select a *client* that is under our control, and then select a remote *endpoint*, and a *Test Domain* that is likely to be censored on the path from the client to the endpoint.

CenTrace measurement. First, we provide an overview of the basic structure of our TTL-based probing technique. We begin our by sending a *probe* with a TLS Client Hello or HTTP GET request for a Control Domain with incrementing values (starting from one) in the TTL field in the IP header (Figure 2 (A)). When setting a low TTL value in the IP header, we expect routers on the path from the client to the endpoint to respond with an ICMP Time Exceeded (Type 11) message [53]. This allows us to identify IP addresses on the path between the client and the endpoint. After performing probes with the Control Domain, we next repeat the same incrementing TTL probes for the Test Domain (Figure 2 (B)). Our method generally follows insights from Paris traceroute [9, 69],

but necessarily differs in reckoning with path variance since our probe contents are stateful in order to measure certain types of censorship, which we discuss later. Throughout our measurements, we perform packet captures and store all responses. We wait 120 seconds before consecutive CenTrace probes to account for stateful blocking by a censorship device, as done in previous work [68]. Next, we describe several design choices for our CenTrace measurement based on the properties of censorship devices.

Detecting blocking. We scope this work to only consider explicit cases of Internet censorship, as network interference could be due to a variety of reasons such as temporary network failures and unexpected endpoint behavior for a certain domain. We define a Test Domain CenTrace as *blocked* when we obtain a clear response that indicates a network intermediary interfering with connections. We consider all cases of connection resets and repeated packet drops as explicit cases of blocking. In case we obtain a TLS or HTTP response, we consider the response as *blocking* only when we obtain a response that matches a known blockpage recorded by Censored Planet, which maintains a curated, comprehensive list of blockpage fingerprints observed in Censored Planet data [13, 62]. Note that our definition of blocking is conservative as censorship devices may inject HTTP responses with empty content to evade detection, which we cannot confidently conclude as censorship (ref §4.2).

Terminating and non-terminating responses. The source IP address of an injected response (e.g. a TCP RST or HTTP blockpage) from a censorship device will be spoofed with the IP address of the endpoint. CenTrace continues probing with incrementing TTL values, either until we receive only a *terminating response or* we reach a maximum TTL value limit (64). We define a *terminating response* as a TCP packet from the endpoint IP address (e.g. TCP RST, TCP FIN, TCP ACK, TCP PSH). This scenario is shown in Figure 2 (B). A *non-terminating response* could either be an ICMP Time Exceeded error, or a probe timing out without any response, and in this case we continue probing with incrementing TTL values. A *terminating hop* is the hop at which the terminating response is observed. In Figure 2 (B), the terminating hop is R3, and the terminating response is a TCP RST.

Accounting for packet drops. In case our probe times out without receiving a response, we retry the request up to three times to account for transient network failures. If we do not receive a response after retrying, this may either indicate intentional packet drops by a censorship device, or a router that does not respond with ICMP Time Exceeded packets. If a probe experiences a timeout at a particular TTL, it is not considered a *terminating* response (as described above) if there is a terminating response at a subsequent TTL. However, if all subsequent TTL probes are timeouts, we consider the first timeout to be the *terminating* response. This scenario is depicted in Figure 2 (C). The terminating hop here is R3, and the terminating response is a Timeout.

Detecting in-path vs. on-path devices. Devices performing censorship can be situated either in-path or on-path, as defined in previous work [3, 45]. In-path devices sit in the network link and operate on network traffic passing through the link at line rate, and can inject, modify, or drop packets. On the other hand, on-path devices sit outside the link and only receive a copy of passing packets. On-path devices can inject packets into the link, but cannot modify or drop packets at line rate. Therefore, on-path censorship techniques may allow censored requests to pass through to the endpoint and the client will receive injected packets alongside legitimate packets. Generally, these on-path censorship techniques are also stateful and rely on TCP flow control to manage censored flows. It is important to differentiate between in-path and on-path devices when determining device location, since in-path devices may have a public router IP address, but on-path devices usually do not.

To detect on-path devices (say, between R2 and R3 in Figure 2 (D)), we capture all packets received after sending a CenTrace measurement, and check whether we receive both an injected terminating response from the endpoint IP address as well as an ICMP Time Exceeded message from hop R3. This indicates that there is an on-path device between hop R2 and hop R3, inclusive, that allows our request for the Test Domain to pass through to Hop R3, but injects a terminating response into the stream. If there is only an injecting terminating response from hop R3, we conclude that the device is in-path. Note that our logic could mistakenly classify an on-path device as in-path if the router at hop R3 does not respond with an ICMP Time Exceeded message resulting in a false positive, since we would only observe the packet injected by the censorship device. However, we account for this case using our Control Domain CenTrace and our results (§4.3) suggest that there are very few cases where the terminating hop does not respond with ICMP errors.

Censorship device location. Once we identify the terminating hop for a blocked Test Domain CenTrace (R3 in Figure 2 (B,C,D)), we then extract the IP address, AS information, and response from the corresponding hop in the Control Domain CenTrace (R3 in Figure 2 (A)). We define this hop as the *blocking hop* and it indicates the approximate network location of the device. In case the device is in-path, we are able to extract the potential IP address of the device. In case the device is on-path, we are only able to extract the location.

Quoted packets in ICMP. Most routers support RFC 792 [53] and RFC 1812 [10], which specify that routers quote parts of the received packets in their ICMP error responses. Following the insights from Tracebox [18], we utilize changes in quoted packet in the ICMP error response to identify at which hops the probe packet is altered. Like Tracebox, we compare fields in the IP header, TCP header, and Application layer payload of the sent probe with the quoted packet in the received ICMP message.

Network path variance. We observe that some stateful censorship devices track packets across the same flow, and react differently once the state has been changed by a flow. Such behavior affects our observations if the methods of censorship are not immediately observable, for instance, in cases with packet drops. Moreover, we observe that some middleboxes only inject censored responses a certain number of times per TCP connection.

Therefore, CenTrace performs each TTL-limited probe over a new TCP connection. However, other work using TTL-based techniques to build up paths have noted that keeping the source and destination ports the same is important for ensuring the packets

Table 1: CenTrace (CT) measurements collected ◇

Co.	In-country			Remote			
	Clients	CTs	Blocked CTs	Endpoints	Endpoint ASNs	CTs	Blocked CTs
AZ	1	18	6	29	10	227	96
BY	-	-	-	123	19	1,040	287
KZ	1	14	8	95	29	868	748
RU	1	14	0	1,291	498	10,488	418

traverse a consistent network path [9, 69]. Since our measurement is entangled with particular TCP sessions, we cannot keep the source port consistent for all measurements. We resolve this challenge by repeating both our Control and Test Domain CenTrace multiple times to estimate all paths to the endpoint, and calculate device locations based on likelihood of paths followed. We use the most commonly observed terminating hop information as the final location of the censorship device.

To estimate the number of repetitions required, we perform an experiment to quantify typical path variance. Our CenTrace measurements are typically conducted to infrastructural endpoints, as described later in 4.2. Over different times of day, we perform 200 traceroutes each to 20 infrastructural endpoints that we control in 20 different countries, and calculate the number of unique paths and the number of times each unique path was followed to an endpoint. We observe that 90% of all paths to each endpoint are covered in 11 traceroutes on average. Only one endpoint faced a really high path variance, with more than 100 unique paths. Therefore, we repeat both our Control and Test Domain traceroutes 11 times, and create a probability distribution of IP addresses at each hop. Then, we extract the most likely IP address at the blocking location as the terminating hop. Note that this is a conservative estimate, as we consider the full path variance for this experiment, but we are more concerned with path variance closer to the censorship device. However, there is still the possibility that certain repeated measurements are more affected by large variances in network paths, resulting in a false positive device location. We report results for CenTrace measurements in aggregate to reduce the effects of such variances.

4.2 Implementation and Measurements

We implement CenTrace in Python 3.9, and use `tcpdump` to store all packet captures and process them. We perform the Control Domain CenTrace probes first and then immediately perform the Test Domain CenTrace probes. A full set of CenTrace probes, including all TTL probes to the Test and Control Domains, runs over approximately 5 minutes. We perform measurements to multiple endpoints concurrently to speed up our data collection.

Using CenTrace, we perform a case study of censorship device location in 4 countries in Eastern Europe and Central Asia with extensive censorship systems, Azerbaijan (AZ), Belarus (BY), Kazakhstan (KZ), and Russia (RU) [39, 55, 60]. We perform in-country measurements from one vantage point each in AZ, KZ, and RU, and remote measurements to hundreds of endpoints in all four countries. When selecting the endpoints, we follow the selection process specified in §2 i.e. we rent in-country vantage points from commercial VPS providers and send remote measurements to infrastructural webservers. We identify potential Test Domains for each country from recent Censored Planet data [14] by selecting five domains for each country and protocol (HTTP, HTTPS) that show the highest blocking in Censored Planet data. Once the Test Domains are selected, we perform our HTTP and TLS CenTrace measurements for all of the endpoints. Our measurements were conducted between May 1 and June 1, 2022. Table 1 shows a summary of our CenTrace measurements in the four countries.

Limitations. We describe limitations of CenTrace to add context to our results. Due to the nature of traceroutes, CenTrace depends on routers near the censorship device responding with ICMP errors. Certain limitations are fundamental to the conceptual approach. For instance, we are only able to extract the potential IP address of in-path devices, which we find to be more common than on-path devices. CenTrace will also not be able to identify censorship devices that are behind NATs or other private network firewalls. While we account for network variance using repeated measurements, paths may still vary to certain endpoints. However, we observe that our results are consistent across multiple domains for the same vantage points.

In addition, full validation of our methods here may be limited to ground-truth knowledge of the censorship systems of the countries we are studying. Since it obtains much more data about the nature and location of censorship, CenTrace can reduce the number of false positives for large-scale censorship platforms like Censored Planet. However, without ground-truth or local validation, we may not be able to detect false positives encountered by CenTrace (i.e. it may detect censorship when there is none).

Other limitations arise from the experimental methods. For isntance, to map IP addresses to ASNs and geolocations, we rely on metadata from both Maxmind and the Routeviews project [47, 58], which can be inaccurate, especially for border router IP addresses [42]. We try to minimize errors by compiling metadata from both data sources and manually validating a small sample of IP to ASN translations.

Our endpoints for remote measurements are also chosen from a list of *infrastructural* machines (§ 2), and this could bias our view of censorship towards devices deployed in organizational networks. Moreover, our remote measurements assume that most censorship devices consider traffic in *both* directions (similar to previous work [62]), however, this may not always be the case (e.g. [79]). We account for this partially using in-country measurements. Due to our strict ethical constraints, our in-country vantage points are also primarily located in datacenters, which means we may not be able to capture censorship exactly as it appears on residential or mobile networks.

As mentioned in §4.1, we only consider blocking when we can identify explicit signals of packet drops, connection resets, or blockpage injections. This could result in our technique missing certain types of censorship, for instance, once that inject illegible content. We also note that a dedicated censorship device can evade our detection by exploiting limitations with CenTrace, such as not responding to queries with expiring TTL values. However, since censorship devices usually only search for the presence of a censored domain [62], we do not believe evasion against our technique is likely to be the priority for censors. Indeed, our results in the

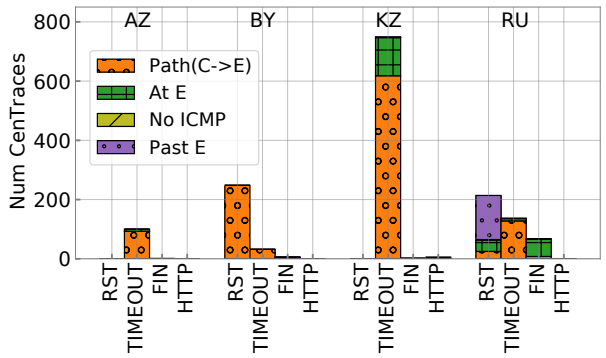

Figure 3: Distribution of blocking type and location with respect to the client (C) and endpoint (E) ⋄

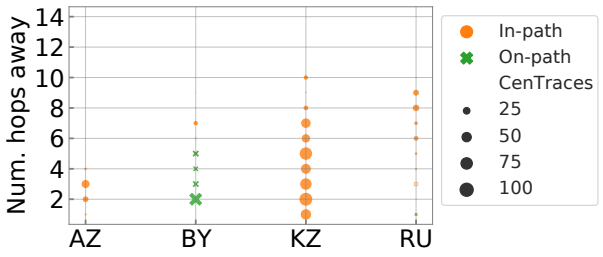

Figure 4: In-path vs on-path devices and hop difference between blocking location and endpoint ⋄

four countries are similar to those obtained by Censored Planet and OONI [14, 63].

4.3 Results

In this section, we describe results from CenTrace. In general, our findings align well with recent research on the censorship systems in the countries we studied [39, 78]. Determining exactly where on the path censorship is occurring helped us identify that some interference upstream of the country is causing censorship-like behavior for particular domains in BY and KZ.

Where are censorship devices located? Figure 3 shows the type of blocked terminating response we receive and the location of the blocking hop with respect to the client and endpoint. In the countries we study, most (94.75%, 1,481) CenTrace measurements experience blocking through packet drops and reset injection. We find only one traceroute where both the terminating hop and the preceding hop did not respond with ICMP Time Exceeded messages ("No ICMP" case). While the majority of blocking hops (73.97%, 1,156) are in the path from the client to the endpoint, there is also a significant portion of traceroutes (16.19%, 253) where the blocking occurs at the endpoint IP itself ("At E" case). In such cases, we observe that the endpoint (or a NAT in front of the endpoint) responds differently (or does not respond) to the Test Domain. While we consider these cases as important observations of network blocking, these usually do not represent cases of ISP or state-sponsored censorship, which many studies focus on. This shows that remote measurement platforms such as Censored Planet can use tools such as CenTrace to understand where the blocking occurs and add more context to their data.

In-country measurements: The censorship devices performing blocking of CenTrace measurements from our AZ and KZ in-country clients are located 2 and 3 hops away from the client respectively, and both drop packets to censored domains. Our RU in-country client does not observe any censorship. According to our AS mapping, the censorship device in AZ lies in Delta Telecom (AS29049), one of the large ISPs in AZ, in which our in-country client is also located. The censorship device in KZ is located in JSC-Kazakhtelecom (AS9198), the state-owned and largest ISP in Kazakhstan which is known to implement censorship policies [7, 60] (See Figure 1). However, our client is located in a downstream hosting provider (AS203087). This shows that censorship may be implemented by an upstream AS, and measurement platforms such as OONI that only consider the client's ASN may not be attributing blocking correctly and may not present a complete picture of censorship policies and devices in a region. We note that the location of the censorship device we find in JSC-Kazakhtelecom is consistent with that found in previous work [39, 60].

Figure 4 shows the hop difference between the blocking hop and the endpoint in our remote measurements when the blocking location is between the client and endpoint. More than 35% of the blocking happens one or two hops away from the endpoint.

AZ remote measurements: We observe that the blocking in AZ primarily occurs in the first few hops after entering the country (Figure 10 in §D). Our AS mapping suggests that packets to 88.89% (24) of the endpoints (that are in 6 different ASes) get dropped in the link from the ISP Telia (AS1299) to Delta Telecom (AS29049, AZ), indicating a centralized censorship infrastructure of blocking within Delta Telecom, as also evidenced by our in-country measurements.

BY remote measurements: In contrast to AZ, measurements to most (91.80%, 56) BY remote endpoints fail in the endpoint AS, through RST injections (Figure 11 in §D). Again, we note that we find device locations in AS 6697 (Beltelecom) that are reported in previous work [39], among others. Interestingly, we find an anomaly for measurements for the Test Domain `bridges.torproject.org` to endpoints in BY. CenTrace measurements to the Tor Bridges domain experience packet drops in AS 174 (COGENT) in 87.88% (29) of the cases. This packet drop occurs even before traffic enters into BY, and the change in censorship methods suggests that the failure is due to a reason other than ISP censorship. Such anomalies are difficult to identify without the use of CenTrace.

KZ remote measurements: In KZ, we find that measurements to 34.07% (31) of remote endpoints time out in two ASes in Russia, AS31133 (PJSC Megafon) and AS43727 (JSC Kvant-telekom) (Figure 12 in §D). This shows that remote censorship measurements to a certain country may be affected by censorship policies in a different country on the path, and platforms such as Censored Planet can benefit from using CenTrace to quantify this phenomenon. We observe the same hops in JSC Kvant-telekom dropping probes in our RU measurements as well. Similar to our in-country measurement in KZ, we find that a large number (31.26%) of our remote CenTrace measurements also terminate in AS 9198 (JSC-Kazakhtelecom).

RU remote measurements: In our RU remote measurements, we observe an interesting phenomenon where CenTrace measurements to 32 endpoint IPs experience terminating hops that seem to be past the hop at which the endpoint is located ("Past E" case in Figure 3). In exploring these cases further, we find that all the injected RST packets we receive have their TTL value set to one i.e. the censorship devices copy the IP header, including the TTL value, from the censored packets they receive (i.e. our Test Domain traceroute) into their injected packets.Therefore, we only receive the injected reset when the TTL value for the Test Domain CenTrace is at least twice the hop distance from the client to the censorship device. This case is illustrated in Figure 2 (E), and is also reported by recent work [39]. We take this behavior into account when determining the terminating hop from the control traceroute. We find censorship devices located in more than 65 ASNs in Russia, many also studied in previous work [55].

Are devices deployed in-path or on-path? We investigate whether censorship devices are deployed in-path or on-path. As shown in Figure 4, censorship devices in AZ and KZ are deployed exclusively in-path, and most censorship devices in RU are also deployed in-path. We primarily observed these censorship devices dropping packets to interfere with communications. On the other hand, most censorship devices in BY are deployed on-path, and inject RST packets into flows. Censorship devices in RU vary in their censored response type and deployment characteristics, which could be an effect of their decentralized censorship policies [55].

Do devices sending ICMP errors quote sent packets? For blocking hops, we extract the quoted IP packet from the ICMP Time Exceeded message returned in the Control Domain traceroute, and compare it with the sent packet. 57.6% of the quoted packets follow RFC792 [53], and only return the first 64 bits of the TCP payload, containing the source and destination ports and the sequence number. The rest of the quoted packets return more fields from the TCP and application-layer payloads, following RFC1812 [10]. Surprisingly, we find that 32.06% of the quoted packets contain a difference in the IP Terms of Service (TOS) field, and one packet even contains different IP flags. We find that these differences in quoted packets can act as useful features in clustering similar devices (refer §7.2).

5 DEVICE BANNERS

Most prior work in identifying network devices use specific device fingerprints that are tuned or discovered manually in order to scan the Internet for the presence of these devices [16, 17, 46, 74]. Recent research has found that many injected blockpages include vendor information, or contain certain patterns that are unique to particular vendors [61]. However, with the increasing adoption of HTTPS and HSTS, where most pages are loaded only over encrypted connections, censorship devices cannot inject any blockpages, which calls for alternative approaches to identifying devices. In this work, we investigate the usefulness of protocol banners for this purpose. In the cases where CenTrace detects the censorship occurring in-path (refer §4.3), we are able to extract the potential IP address of the device. This allows us to probe the potential IP address, and collect additional features for identifying device vendors.

5.1 Method

We actively probe potential IP addresses of in-path devices to check whether we obtain any indication of filtering software. We first use Nmap to scan the top 1,000 ports on all devices and extract the ports that are open, a strategy used by previous work [37]. Nmap then sends up to 16 specially crafted TCP, UDP, and ICMP probes to the device, on both open and closed ports. These probes are each intended to invoke a unique and potentially fingerprintable response. Nmap then transforms these responses into features based on manual tests that have been developed over the past 20 years. On the portion of devices that have ports open, we perform application-layer banner grabs using ZGrab [82]. Specifically, we collect the handshake information and initial responses on the HTTP(S), SSH, Telnet, FTP, SMTP, and SNMP protocols from each device that has the corresponding service open. These protocols have been used by previous work for collecting features for network device fingerprinting [37]. We also extract information about services running on non-standard ports from Censys [22], which scans all ports on every IPv4 address for banners regularly. Then, we use manual investigation and Rapid7's Recog [56], a public fingerprint repository, to label devices with filtering technology that respond to our banner grabs.

5.2 Measurements

AZ, BY, KZ, RU. We collect banners and network fingerprints from 163 IP addresses that could potentially host in-path censorship devices based on CenTrace measurements in AZ, BY, KZ, and RU. These are the IP addresses of the terminating hop in our Control Domain CenTrace measurement.

Limitations. In the banner collection process, we are only able to collect banners for *potential* IP addresses of a censorship device from our CenTrace measurements. Certain devices may not have publicly visible IP addresses. Thus, we only report on devices that we can explicitly identify as running firewall software. Moreover, our view of censorship devices is biased towards devices that have open ports that we can collect banners from. We note that making stronger claims with certainty about device provenance and attribution still requires considerable manual work and collection of ground truth.

Comparison with Blockpages. As additional validation, we compare labels extracted from active probing with labels extracted from blockpages, when available. Since the censorship devices in the four countries we study only inject blockpages in very few (5) cases, we run an additional set of CenTrace and banner grab measurements to endpoints around the world where recent Censored Planet measurements identified blockpage injection [14]. In the first week of June 2022, Censored Planet HTTP measurements identified connections to 126 remote endpoint IP addresses where an in-path device presented a known device blockpage. Therefore, as a case study, we choose a subset of one endpoint IP address from every endpoint ASN that observed a blockpage, which results in 76 endpoint IP addresses, and run CenTrace HTTP measurements to these IP addresses. We then collect banners from the resulting set of 71 potential censorship device IP addresses that are in-path.

5.3 Findings

Are device banners useful in device identification? We rely on our measurements to endpoints with known blockpage injection to understand if banners collected from devices can complement the use of blockpages in identifying devices. 87.32% (62) of the potential device IP addresses support at least one of the services we collect banners from. We manually investigate and label these banners, and find that 38.71% (28) of devices show a clear indication of firewall software used for blocking website requests. Moreover, these device labels match exactly the device identification from the blockpage. Our case study shows that extracting information from device banners can act as a valuable approach in device identification.

What vendors are censoring network traffic in AZ, BY, KZ, and RU?. Overall, potential censorship device IP addresses in these countries do not always host public services, nor do they frequently respond with blockpages. Out of these 163 potential device IP addresses in our four countries, only 68 (41.72%) have at least one SSH, Telnet, FTP, SMTP, or SNMP port open. On investigating the banners manually, we find 19 devices with explicit indication of device vendors: Cisco (7 devices, AZ, KZ and RU), Fortinet (5 devices, AZ, KZ, and RU), Kerio control (2 devices, KZ), Palo Alto (2 devices, AZ and RU), DDoSGuard (1 device, RU), Mikrotik (1 device, KZ) and Kasperky (1 device, RU). Importantly, other than the Fortinet devices, the others do not inject any blockpages, and drop packets instead. This shows the importance of performing banner grabs in addition to collecting injected blockpage responses. In addition to these 19 devices, we find 4 other Fortinet devices sending blockpages but not presenting banners.

6 CENSORSHIP FUZZING

Identifying the triggers and rules of censorship devices is key to understanding how network traffic is blocked. This not only informs methods of censorship circumvention, as has been explored in prior work [11], but can also provide additional features for device identification. Network censorship devices, especially those manufactured by commercial vendors such as the ones we find in §5.3, are frequently implemented with special TCP, TLS, and HTTP stacks that are distinctive to a particular vendor, or are configured to block certain requests by actors deploying these devices. Previous work on censorship circumvention has shown that censorship implementations frequently contain idiosyncrasies in their connection parsing that may be utilized to evade blocking [11, 38, 72].

In this work, we develop a deterministic censorship fuzzing tool, CenFuzz, that performs application-layer fuzzing strategies on blocked connections such that the same strategies are performed across all tested devices. CenFuzz performs several modifications to the HTTP GET Request and the TLS Client Hello packets, based on the grammars of these protocols (ref. Appendix §B). We choose to employ deterministic fuzzing as compared to previous circumvention techniques for two reasons: (1) Circumvention tools gravitate towards strategies that exploit a difference in parsing characteristics between the censor and the endpoint which places a significant limitation on the number of strategies that can be applied, since strategies that do not elicit the correct response from the endpoint are invalid. Because of this differences, in this section, we use censorship *evasion* to mean that a particular probe has evaded a particular censorship rule. We use the term censorship *circumvention* to mean that a particular probe has both *evaded* a censorship rule, and that the probe loads the intended resource correctly. (2) Some circumvention tools focus on using machine learning to find optimal strategies that work on censors [11]; however, based on the censor's implementation and deployment characteristics, the strategies tested and paths followed by the machine learning model may be different since the process introduces randomness. If the goal is to produce a set of deterministic network fingerprints, we need a static set of strategies to test against the endpoint to ensure the feature space for each measurement is the same. CenFuzz aims to test the same strategies against every device to produce such a fingerprint.

Table 2: CenFuzz HTTP request and TLS client hello fuzzing strategies. 'NP'–Number of Permutations ⋄

Category	HTTP Strategy	Examples	NP
Alternate	Get Word	`POST`, `PUT`	6
	HTTP Word	`HTTP/ 1.1`, `XXXX/1.1`	16
	Host Word	`HostHeader:`	7
	Path	`?,z`	8
	Hostname	`www.example.comwww.example.com`	5
	Hostname TLD	`www.example.net`	10
	Hostname Subdomain	`m.example.com`	10
	Header	`Connection: keep-alive`	59
Capitalize	Get Word	`GeT`	8
	HTTP Word	`HtTP/1.1`	16
	Host Word	`HoST:`	16
Remove	Get Word	`GE`	7
	HTTP Word	`HTTP/.1`	167
	Host Word	`ost:`	63
	HTTP Delimiter	`\r`	3
Pad	Hostname Padding	`**www.example.com*`	9
Category	HTTPS Strategy	Examples	NP
Alternate	Min TLS Version	`TLS 1.1`	4
	Max TLS Version	`TLS 1.1`	4
	Cipher Suite	`TLS_AES_128_GCM_SHA256`	25
	Client Certificate	`CN=www.test.com`	3
	Server Name (SNI)	`moc.elpmaxe.www`	4
	SNI TLD	`www.example.org`	10
	SNI Subdomain	`wiki.example.com`	10
Pad	SNI Padding	`***www.example.com`	9

6.1 Method

Table 2 shows an overview of CenFuzz strategies and the number of fuzzed requests sent by each strategy. The strategies are chosen based on their likelihood to elicit different responses from censorship devices, based on examples from previous work [11, 38].

Alternate Data. Many of CenFuzz's strategies attempt to substitute data within the grammar of the protocol with some other valid or invalid data. Some of the HTTP fuzzing strategies include using a different HTTP Method instead of `GET` (e.g. `POST`, `PUT`, `XXXX`), using a different HTTP Word (e.g. `HTTP/3`, `HTTP/ 1.1`, `HTTP /1.2`), and using a different Host Word (e.g. `HostHeader:`, `XXXX:`). CenFuzz also tries providing different resource location paths (e.g. `?, z`) to check whether only the home page (i.e. the `/` path) is blocked. Another fuzzing strategy is to use additional headers such as `Connection: keep-alive` and `User-Agent: xxx`, and check

whether adding headers changes the blocking behavior. Some of the strategies adopted by CenFuzz change the hostname itself, which is usually the identifier for keyword-based content blocking. It attempts to omit the hostname completely, provide an empty hostname, reverse the hostname (e.g. `moc.elpmaxe.www`), and repeat it multiple times. It also attempts to provide different TLDs and subdomains for the hostname (e.g. `m.example.com`, `www.example.net`). In TLS, CenFuzz attempts to change the minimum and maximum accepted TLS versions, under acceptable values (TLS 1.0, 1.1, 1.2, or 1.3). It tests different ciphersuites individually, and also checks whether providing a client certificate (either for the requested domain or some other domain) changes the censorship device behavior. Similar to the HTTP strategies, CenFuzz applies changes to the domain in the SNI field, changing or omitting the domain itself, changing the subdomain, and changing the TLD.

Capitalize, Remove, and Pad Data. CenFuzz selectively capitalizes and removes different characters from the HTTP words to detect whether censorship devices fail to act on incomplete HTTP requests (e.g. `GE / HTTP/1.1 \r\n Host: www.example.com\r\n\r\n`, `GET / HTtP/1.1\r \n Host: www.example.com\r\n\r\n`). CenFuzz also pads the hostname (in HTTP) and server name (in TLS) with leading and trailing padding characters (e.g. `**www.example.com*`).

6.2 Implementation and Measurements

We implement CenFuzz in Go 1.16, using Go's `net` and Refraction Networking's `utls` [57] libraries to manage connections. For each strategy, we store the responses for a *Normal* request to the Test Domain and Control Domain (without any fuzzing modifications). Then we perform one measurement each for each variation of the strategy for both the Control Domain and the Test Domain. Finally, we compare the results for the Test Domain between the permutations and the Normal request, and also compare results between the Test Domain and the Control Domain for the same permutation. We define a permutation as *not successful* if both the Normal Test Domain request and the Test Domain permutation are blocked, while the Control Domain permutation is not blocked. Similarly, we define a permutation as *successful* if the Normal Test Domain request is blocked, but both the Test Domain permutation and the Control Domain permutation are not blocked. Similar to our process in § 4, we conservatively restrict our definition of *blocking* to consider only those responses where we observe a repeated packet drop, a connection reset or failure, or a known blockpage injected by a middlebox. We perform measurements to different endpoints in parallel, waiting for 120 seconds between measurements with different strategies in case of detected blocking (to avoid effects of stateful blocking as described in §4.1, and 3 seconds otherwise. We perform fuzzing measurements to the endpoints in Section 4 that observed blocking i.e. we performed in-country and remote fuzzing measurements in AZ, KZ, and RU, and only remote fuzzing measurements in BY. In total, we conduct more than 2.48 million CenFuzz measurements.

6.3 Results

Which strategies evade censorship rules? Figure 5 shows the distribution of fuzzing strategy permutations that are successful in

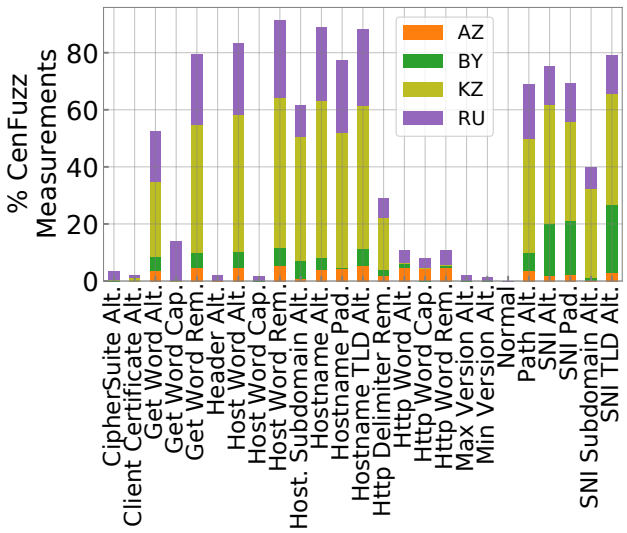

Figure 5: Success rates of CenFuzz strategies ◇

AZ, BY, KZ, and RU. We find that the success rate of an Alternate HTTP Method varies based on censor rules. For example, using the `POST` HTTP method instead of `GET` only evades devices 1.76% (10) of the time, while using the `PUT`, `PATCH` and an empty HTTP method evade the censorship device 21.63% (122), 82.15% (465), and 92.01% (518) of the times respectively. Note that other than the empty HTTP method, all others are valid HTTP methods. Adding additional headers (even invalid ones) do not result in evasion. Surprisingly, providing invalid alternates for the HTTP version (e.g. HTTP/9) results in very few successful cases (10.55%, 946). This shows that many censorship devices do not check for validity of the HTTP version, but trigger only on certain HTTP methods. Providing a different path other than the default (`/`) also succeeds in evading blocking 68.72% (3,080) of the time, providing a possible circumvention strategy if resources are hosted in multiple URLs.

Changing the hostname (the HTTP Host header) itself predictably evades the censorship devices in a large number of cases, as these exploit the censorship policies of the device. Padding the hostname with leading and trailing pad characters are successful 77.12% (3,924) of the time. We find that a large number of devices implement rules with leading wildcards rather than trailing wildcards i.e. rules of the form `*.blockeddomain.tld` over `blockeddomain.*`. Therefore, permutations with leading pads are mostly blocked, while those with any trailing pads often evade blocking. Due to the same reason, changing the hostname's TLD is a more successful strategy (88%, 4,905) than changing the hostname's subdomain (61.52%, 3,424).

CenFuzz's TLS strategies that work by changing parts of the server name (SNI) behave similarly to the hostname strategies in HTTP. However, we observe that changing other parts of the TLS Client Hello are not highly successful. This shows that censorship devices parse a variety of TLS configurations, and trigger only on the SNI. We find a few cases in RU and KZ where the use of certain ciphersuites (e.g. `TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384`, `TLS_RSA_WITH_RC4_128_SHA` evades the censor, and we

find a few cases in RU and BY where changing the TLS version results in evasion, such as setting the TLS Version to 1.0 or 1.3. We observe that both of these strategies work well in circumvention.

In HTTP CenFuzz measurements, we find that our `Remove` strategies manage to evade triggering most devices. For example, removing parts of the Host Word evades devices more than 91.3% (31,518) of the time. Removing parts of the HTTP Method (GET) and the HTTP Delimiter are also successful in all of the countries. Our `Capitalize` strategies are comparatively less successful, only evading censorship devices in a few cases. Only the HTTP Word Capitalization strategy works for more than 1% of the cases. This shows that the rules of most devices account for different character cases, but not incomplete words in the request line.

Which strategies are successful in circumvention? While circumvention is not our primary goal, we investigate the applicability of our CenFuzz strategies for circumvention using data from our in-country measurements, as we attempt to connect to a legitimate IP address for the domain. Our results are in line with recent work by Harrity et al. using the Geneva platform, which automates the discovery of application-layer circumvention techniques [35]. For example, from our KZ in-country measurement, we observe that padding the SNI and hostname for the domain `www.pokerstars.com` with leading pad characters result in evading the censor and successfully fetching legitimate content. Requests for `dailymotion.com` in KZ circumvent the censor and fetch legitimate content when certain subdomains (such as `wiki.dailymotion.com`) are used. Web servers for other domains do not necessarily recognize the same request and return errors with status codes such as `400 Bad Request`, `403 Forbidden`. `301 Moved Permanently` and `505 HTTP Version Not Supported`, therefore the applicability of circumvention varies by domain.

7 CLUSTERING DEVICES

We design a pipeline to cluster censorship measurements using features extracted from CenTrace, banner grabs, and CenFuzz. By clustering censorship device deployments with the same behavior, we aim to explore whether devices from the same vendor show the same properties and understand how censorship policies are deployed. Demonstrating that devices from the same vendor exhibit similar censorship policies and properties will help generate new features for recognizing and studying their deployment patterns.

7.1 Feature extraction

For *each endpoint that encountered blocking* in our CenTrace measurements, we extract features from our CenTrace measurement, CenFuzz measurement, and banner grab measurements. From CenTrace, we use the type of censorship that occurs (i.e. packet drops, RST & FIN injection, or HTTP injection) as a feature. In all cases other than packet drops, we also extract network features from the injected packet TCP/IP headers, such as the IP ID, IP flags set, TCP flags set, and TCP options set. We identify any changes in IP/TCP headers before and after the terminating hop using the quoted packet in ICMP. An one-hot encoding of these deltas are used as features. For measurements that do not encounter quoted packets, we impute the data. Since we perform the same censorship fuzzing strategies across all devices, we extract the list of strategies that are successful against a particular censorship deployment. From our Nmap probes and banner grabs, we extract the ports that are open, as well as features from Nmap fingerprinting. The full feature set can be found in Appendix C.

If any of the devices respond with an explicit vendor indication in an injected blockpage, or in a banner, we then extract this data as a label. For these labeled devices, we can then generate a fingerprint from any other network-level responses, the censorship response, and features from CenFuzz. Using these other network-layer and censorship features, we can then classify the vendors devices that do not inject blockpages, or do not explicitly display its vendor in banner responses.

7.2 Feature importance

We examine the relative importance of each of the features described above by training a classifier using all data that have labels obtained from our case study in comparing blockpages and banner grabs i.e. our dataset with known blockpage matches from Censored Planet (§5.2). We use a random-forest classifier for model interpretability and feature importance measurements. We measure the importance of each feature using the mean-decrease in impurity (MDI) calculated by the random-forest classifier, which measures the degree to which the classifier has learned to use that particular feature to perform a prediction. We impute missing features in the data via taking the median of other samples. Finally, we train the classifier three times using 5-fold cross-validation (for a total of 15 repetitions), then extract each feature's MDI across each tree in the random-forest classifier. Our results (Figure 9 in §C) indicate that the type of terminating response (i.e. packet drops vs TCP RSTs) is highly indicative of deployments of the same vendor. In addition, other important features include CenFuzz requests that behave differently for different censorship device vendors.

7.3 Unsupervised clustering of data

Based on the feature importance determined in §7.2, we pick the top 10 features that perform best to cluster data collected in AZ, BY, KZ, and RU. We use DBSCAN clustering, which uses a density metric (ϵ) to determine the number of clusters in the data rather than a pre-determined number of clusters [59]. Density-based clustering is more pertinent to our use case since we do not necessarily know how many types of devices there are in the unlabeled data. We use $\epsilon = 1.2$, determined using a technique established in prior literature that measures the average distance between each point and its k nearest neighbors, where k is the minimum number of points we expect to be able to form a cluster [54].

7.4 Findings

The devices that were labelled as being manufactured by the same vendor exhibited extremely similar censorship and network features. In addition, censorship devices in the same country and ASes often formed very tight clusters. There were a few clusters, however, that contained devices across different countries and ASes, implying some censorship devices across these countries may be manufactured by the same vendor, or at least implemented very similarly.

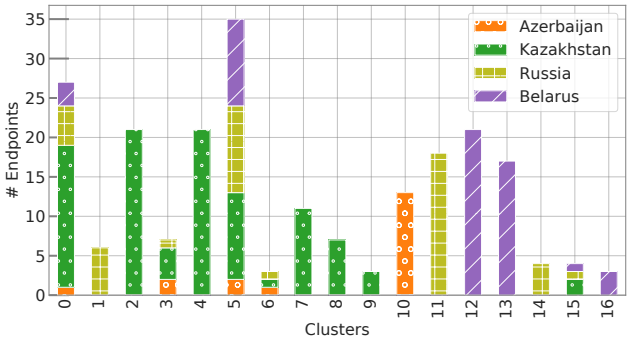

Figure 6: Clusters of endpoints based on features from CenTrace, CenFuzz, and banner grabs ⋄

Are censorship and network features similar across the same device manufacturers? We explore whether devices manufactured by the same vendor show similar censorship properties (see §7.1). Using Spearman's rank correlation coefficient and p-value (r_s and p) to calculate pairwise correlations of feature similarity, we observe strong positive correlations among devices labeled with the same vendor. All the Fortinet firewall devices display almost exactly the same censorship properties ($r_s = 1.00$, $p = 0.00$). The Cisco firewall devices ($r_s > 0.78$, $p = 0.00$) and the two Kerio Control firewall devices ($r_s = 0.98$, $p = 0.00$) also exhibit very similar censorship properties. Devices from different vendors, however, show weaker correlation (e.g. Fortinet vs Cisco, $r_s = 0.56$, $p = 0.02$). Among labeled devices, ones that exhibit the exact same features from CenFuzz are labeled as the same vendor. The type of censorship response and resulting features are also often different between the clusters.

Overall, the devices that we were able to label all exhibit similar censorship and network features. The censorship devices that we are able to identify as manufactured by the same vendor are always in the same clusters. Our results show that devices manufactured by the same vendor or those deployed by the same actor exhibit highly similar combinations of network fingerprints and censorship properties. This result from the unsupervised learning implies that that these properties can be used to fingerprint devices and identify more deployments for future research.

Do censorship devices vary by country or AS?. The results of clustering endpoints in AZ, BY, KZ, and RU per §7.3 are visualized in Figure 6. 69% of endpoints form tight clusters with other endpoints in the same country (and often in the same AS), showing that censorship policies and devices may be configured at the AS or country level. However, a few clusters (such as clusters 3, 5, 6, and 15) also indicate that endpoints even across different countries encounter similar blocking patterns. The smaller clusters 3, 6, and 15 consisting of measurements from multiple countries, exhibit very similar censorship properties ($r_s > 0.99$, $p \approx 0.00$, averaged across pairwise correlations). The measurements in the larger cluster 5 also exhibit similar censorship properties ($r_s \approx 0.82$, $p \approx 0.00$). This suggests that censorship devices in these clusters are implemented similarly, potentially by the same vendor.

8 DISCUSSION & CONCLUSION

In this paper, we have developed and implemented various techniques to locate censorship devices and understand their triggers and properties. Using the techniques we develop, we conduct a study of censorship devices in four countries (AZ, BY, KZ, and RU). We open-source and maintain our tools and data at https://censoredplanet.org/censorship-devices to enable the continued monitoring of censorship devices.

Future work. Future work can explore devices that perform DNS packet injection, TCP blocking and blocking of other protocols, as we only consider devices that perform HTTP and TLS censorship in this study. As mentioned in §4.2, our view is biased towards devices deployed in organizational networks due to our endpoint selection criteria, and future work can integrate our measurement techniques into measurement platforms such as OONI to collect more user-centric measurements. Our study is limited to primarily four countries in the same region and future work can explore whether our findings generalize across other countries and regions. Finally, we hope that future work can build on these techniques to develop censorship middlebox fingerprints at scale.

Implications. Our results further the study of censorship devices. First, we find that since a significant portion of censorship occurs at an upstream AS (which are sometimes even in a different country) of the Client or Endpoint, and we can identify these devices via CenTrace. In addition, large-scale censorship measurement platforms, such as OONI and Censored Planet, currently suffer from lack of knowledge about where censorship is being performed, and can benefit from using CenTrace. The location of the censorship device can also be used by circumvention tools based on content localization to better avoid blocking [34, 75]. The resulting network and censorship features from CenFuzz may be used to construct fingerprints for the vendors of censorship devices. Our results also reveal how censorship devices read requests, and using tools such as CenFuzz can help circumvention developers build better circumvention strategies that work against particular devices [11]. Our methods also advance the state-of-the-art in device identification, which can be used to monitor the proliferation of different censorship devices in countries around the world [62]. Rapid advancements in network technology have led to highly available, low-cost, extremely capable censorship software being available at censors' disposal. We hope that our work serves as a method for researchers to monitor and police the development and spread of censorship technologies.

9 ACKNOWLEDGMENTS

The authors thank the shepherd Georgios Smaragdakis and the anonymous reviewers for their helpful feedback. We are also grateful to Ron Deibert, Masashi Crete-Nishihata, and Anna Ablove for their help and support for this work. This work was supported by an Open Technology Fund (OTF) Information Controls Fellowship (ICFP), the Defense Advanced Research Projects Agency under Agreement No. HR00112190127 and a Bureau of Democracy, Human Rights and Labor (DRL) Grant (No. SLMAQM20GR2132).

REFERENCES

[1] Access Now. U.S.-Canadian firm Sandvine fosters Russian censorship infrastructure, 2022. https://www.accessnow.org/sandvine-russian-censorship/.

[2] G. Aceto, A. Botta, A. Pescapè, N. Feamster, M. Faheem Awan, T. Ahmad, and S. Qaisar. Monitoring internet censorship with ubica. In *International Workshop on Traffic Monitoring and Analysis*, pages 143–157. Springer, 2015.

[3] A. Akhavan Niaki, S. Cho, Z. Weinberg, N. P. Hoang, A. Razaghpanah, N. Christin, and P. Gill. ICLab: A Global, Longitudinal Internet Censorship Measurement Platform. In *IEEE Symposium on Security and Privacy (S&P)*, 2020.

[4] T. Albakour, O. Gasser, R. Beverly, and G. Smaragdakis. Third time's not a charm: Exploiting SNMPv3 for router fingerprinting. In *Proceedings of the 21st ACM Internet Measurement Conference*, pages 150–164, 2021.

[5] Anonymous. Towards a comprehensive picture of the Great Firewall's DNS censorship. In *Free and Open Communications on the Internet (FOCI)*, 2014.

[6] Anonymous, A. A. Niaki, N. P. Hoang, P. Gill, and A. Houmansadr. Triplet censors: Demystifying Great Firewall's DNS censorship behavior. In *Free and Open Communications on the Internet*. USENIX, 2020.

[7] APNIC. Visible asns: Customer populations (est.), 2022. https://stats.labs.apnic.net/aspop?c=kz.

[8] H. Asghari, M. Van Eeten, and M. Mueller. Unraveling the economic and political drivers of deep packet inspection. In *GigaNet 7th Annual Symposium, November*, volume 5, 2012.

[9] B. Augustin, X. Cuvellier, B. Orgogozo, F. Viger, T. Friedman, M. Latapy, C. Magnien, and R. Teixeira. Avoiding traceroute anomalies with Paris traceroute. In *Proceedings of the 6th ACM SIGCOMM conference on Internet measurement*, pages 153–158, 2006.

[10] F. Baker. Requirements for IP version 4 routers, 1995. https://datatracker.ietf.org/doc/html/rfc1812.

[11] K. Bock, G. Hughey, X. Qiang, and D. Levin. Geneva: Evolving censorship evasion strategies. In *Computer and Communications Security*. ACM, 2019.

[12] K. Bock, G. Naval, K. Reese, and D. Levin. Even censors have a backup: Examining China's double HTTPS censorship middleboxes. In *Free and Open Communications on the Internet*. ACM, 2021.

[13] Censored Planet. Censored Planet assets, 2022. https://assets.censoredplanet.org.

[14] Censored Planet. Censored Planet raw data, 2022. https://data.censoredplanet.org/raw.

[15] H. Cheng, W. Dong, Y. Zheng, and B. Lv. Identify IoT devices through web interface characteristics. In *2021 IEEE 6th International Conference on Computer and Communication Systems (ICCCS)*, pages 405–410. IEEE, 2021.

[16] J. Dalek, L. Gill, B. Marczak, S. McKune, N. Noor, J. Oliver, J. Penney, A. Senft, and R. Deibert. Planet Netsweeper, 2018. https://citizenlab.ca/2018/04/planet-netsweeper/.

[17] 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 *Internet Measurement Conference (IMC)*. ACM, 2013.

[18] G. Detal, B. Hesmans, O. Bonaventure, Y. Vanaubel, and B. Donnet. Revealing middlebox interference with Tracebox. In *Proceedings of the Internet Measurement Conference*, pages 1–8, 2013.

[19] T. Dierks and E. Rescorla. The Transport Layer Security (TLS) protocol version 1.1, 2006. https://www.rfc-editor.org/rfc/rfc4346.

[20] T. Dierks and E. Rescorla. The Transport Layer Security (TLS) protocol version 1.2, 2008. https://datatracker.ietf.org/doc/html/rfc5246.

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

[22] Z. Durumeric, D. Adrian, A. Mirian, M. Bailey, and J. A. Halderman. A search engine backed by Internet-wide scanning. In *Proceedings of the 2015 ACM SIGSAC Conference on Computer and Communications Security*, 2015.

[23] Z. Durumeric, E. Wustrow, and J. A. Halderman. ZMap: Fast Internet-wide Scanning and Its Security Applications. In *22nd USENIX Security Symposium*, pages 605–620, 2013.

[24] D. Eastlake. Transport Layer Security (TLS) extensions: Extension definitions, 2011. https://datatracker.ietf.org/doc/html/rfc6066.

[25] R. Ensafi, J. Knockel, G. Alexander, and J. R. Crandall. Detecting intentional packet drops on the Internet via TCP/IP side channels. In *Passive and Active Measurement Conference*. Springer, 2014.

[26] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. Hypertext Transfer Protocol – HTTP/1.1, 1999. https://datatracker.ietf.org/doc/html/rfc2616.

[27] R. Fielding, Y. Lafon, and J. Reschke. Hypertext Transfer Protocol (HTTP/1.1): Range requests, 2014. https://datatracker.ietf.org/doc/html/rfc7233.

[28] R. Fielding, M. Nottingham, and J. Reschke. Hypertext Transfer Protocol (HTTP/1.1): Caching, 2014. https://datatracker.ietf.org/doc/html/rfc7234.

[29] R. Fielding and J. Reschke. Hypertext Transfer Protocol (HTTP/1.1): Authentication, 2014. https://datatracker.ietf.org/doc/html/rfc7235.

[30] R. Fielding and J. Reschke. Hypertext Transfer Protocol (HTTP/1.1): Conditional requests, 2014. https://datatracker.ietf.org/doc/html/rfc7232.

[31] R. Fielding and J. Reschke. Hypertext Transfer Protocol (HTTP/1.1): Message syntax and routing, 2014. https://datatracker.ietf.org/doc/html/rfc7230.

[32] R. Fielding and J. Reschke. Hypertext Transfer Protocol (HTTP/1.1): Semantics and content, 2014. https://datatracker.ietf.org/doc/html/rfc7231.

[33] D. Fifield, C. Lan, R. Hynes, P. Wegmann, and V. Paxson. Blocking-resistant communication through domain fronting. *Privacy Enhancing Technologies*, 2015(2), 2015.

[34] D. Gosain, M. Mohindra, and S. Chakravarty. Too close for comfort: Morasses of (anti-) censorship in the era of CDNs. *Privacy Enhancing Technologies*, 2021(2), 2021.

[35] M. Harrity, K. Bock, F. Sell, and D. Levin. GET /out: Automated discovery of Application-Layer censorship evasion strategies. In *31st USENIX Security Symposium (USENIX Security 22)*, pages 465–483, Boston, MA, Aug. 2022. USENIX Association.

[36] 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*. USENIX, 2021.

[37] J. Holland, R. Teixeira, P. Schmitt, K. Borgolte, J. Rexford, N. Feamster, and J. Mayer. Classifying network vendors at internet scale. *arXiv preprint arXiv:2006.13086*, 2020.

[38] J. Jermyn and N. Weaver. Autosonda: Discovering rules and triggers of censorship devices. In *Free and Open Communications on the Internet*. USENIX, 2017.

[39] L. Jin, S. Hao, H. Wang, and C. Cotton. Understanding the practices of global censorship through accurate, end-to-end measurements. In *Abstract Proceedings of the 2022 ACM SIGMETRICS/IFIP PERFORMANCE Joint International Conference on Measurement and Modeling of Computer Systems*, pages 17–18, 2022.

[40] F. Li, A. M. Kakhki, D. Choffnes, P. Gill, and A. Mislove. Classifiers unclassified: An efficient approach to revealing ip traffic classification rules. In *Proceedings of the 2016 Internet Measurement Conference*, pages 239–245, 2016.

[41] F. Li, A. Razaghpanah, A. M. Kakhki, A. A. Niaki, D. Choffnes, P. Gill, and A. Mislove. lib• erate,(n) a library for exposing (traffic-classification) rules and avoiding them efficiently. In *Proceedings of the 2017 Internet Measurement Conference*, pages 128–141, 2017.

[42] M. Luckie, A. Dhamdhere, B. Huffaker, D. Clark, and K. Claffy. Bdrmap: Inference of borders between IP networks. In *Proceedings of the 2016 Internet Measurement Conference*, pages 381–396, 2016.

[43] G. F. Lyon. *Nmap Network Scanning: The Official Nmap Project Guide to Network Discovery and Security Scanning*. Nmap Project, Sunnyvale, CA, 12.2.2008 edition edition, Jan. 2009.

[44] B. Marczak, J. Dalek, S. McKune, A. Senft, J. Scott-Railton, and R. Deibert. Bad Traffic: Sandvine's PacketLogic Devices Used to Deploy Government Spyware in Turkey and Redirect Egyptian Users to Affiliate Ads? Technical report, Citizen Lab, University of Toronto, 2018.

[45] 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 *Free and Open Communications on the Internet*. USENIX, 2015.

[46] M. Marquis-Boire, J. Dalek, S. McKune, M. Carrieri, M. Crete-Nishihata, R. Deibert, S. O. Khan, H. Noman, J. Scott-Railton, and G. Wiseman. Planet Blue Coat, 2013. https://citizenlab.ca/2013/01/planet-blue-coat-mapping-global-censorship-and-surveillance-tools/.

[47] MaxMind. https://www.maxmind.com/.

[48] Mozilla Developer Network. Host, 2022. https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Host.

[49] National Commission for the Protection of Human Subjects of Biomedical and Behavioral Research. *The Belmont Report: Ethical Principles and Guidelines for the Protection of Human Subjects of Research*. National Commission for the Protection of Human Subjects of Biomedical and Behavioral Research, 1978.

[50] 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/, 2022.

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

[52] PeeringDB. Peeringdb, 2018. https://www.peeringdb.com/.

[53] J. Postel. Internet control message protocol, 1981. https://datatracker.ietf.org/doc/html/rfc792.

[54] N. Rahmah and I. S. Sitanggang. Determination of optimal epsilon (eps) value on DBSCAN algorithm to clustering data on peatland hotspots in sumatra. In *IOP conference series: earth and environmental science*, volume 31, page 012012. IOP Publishing, 2016.

[55] R. Ramesh, R. S. 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 System Security*. The Internet Society, 2020.

[56] Rapid7. Recog: A recognition framework, 2022. https://github.com/rapid7/recog.

[57] Refraction Networking. uTLS, 2022. https://github.com/refraction-networking/utls.

[58] University of Oregon Route Views Project. www.routeviews.org.

[59] E. Schubert, J. Sander, M. Ester, H. P. Kriegel, and X. Xu. DBSCAN revisited, revisited: why and how you should (still) use DBSCAN. In *ACM Transactions on Database Systems (TODS)*, volume 42, pages 1–21. ACM New York, NY, USA, 2017.
[60] R. Sundara Raman, L. Evdokimov, E. Wustrow, A. Halderman, and R. Ensafi. Investigating Large Scale HTTPS Interception in Kazakhstan. In *Internet Measurement Conference (IMC)*, 2020.
[61] R. Sundara Raman, P. Shenoy, K. Kohls, and R. Ensafi. Censored Planet: an internet-wide, longitudinal censorship observatory. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, pages 49–66, 2020.
[62] R. Sundara Raman, A. Stoll, J. Dalek, R. Ramesh, W. Scott, and R. Ensafi. Measuring the deployment of network censorship filters at global scale. In *NDSS*, 2020.
[63] The Tor Project. OONI: Open observatory of network interference. https://ooni.torproject.org/.
[64] A. Troianovski and V. Safronova. Russia Takes Censorship to New Extremes, Stifling War Coverage. New York Times, 2022. https://www.nytimes.com/2022/03/04/world/europe/russia-censorship-media-crackdown.html.
[65] UNHRC. Report of the Special Rapporteur on the promotion and protection of the right to freedom of opinion and expression, 2019. https://documents-dds-ny.un.org/doc/UNDOC/GEN/G19/148/76/PDF/G1914876.pdf?OpenElement.
[66] Y. Vanaubel, J.-J. Pansiot, P. Mérindol, and B. Donnet. Network fingerprinting: TTL-based router signatures. In *Proceedings of the 2013 conference on Internet measurement conference*, pages 369–376, 2013.
[67] B. VanderSloot, S. Frolov, J. Wampler, S. C. Tan, I. Simpson, M. Kallitsis, J. A. Halderman, N. Borisov, and E. Wustrow. Running refraction networking for real. *Privacy Enhancing Technologies*, 2020(3):321–335, 2020.
[68] B. VanderSloot, A. McDonald, W. Scott, J. A. Halderman, and R. Ensafi. Quack: Scalable remote measurement of application-layer censorship. In *USENIX Security Symposium*. USENIX, 2018.
[69] K. Vermeulen, S. D. Strowes, O. Fourmaux, and T. Friedman. Multilevel mda-lite Paris traceroute. In *Proceedings of the Internet Measurement Conference 2018*, pages 29–42, 2018.
[70] Vice. Netsweeper removes alternate lifestyle category, 2019. https://motherboard.vice.com/en_us/article/3kgznn/netsweeper-says-its-stopped-alternative-lifestyles-censorship.
[71] A. Vyas, R. Sundara Raman, N. Ceccio, P. M. Lutscher, and R. Ensafi. Lost in Transmission: Investigating Filtering of COVID-19 Websites. In *Financial Cryptography and Data Security (FC)*, 2021.
[72] Z. Wang, S. Zhu, Y. Cao, Z. Qian, C. Song, S. V. Krishnamurthy, K. S. Chan, and T. D. Braun. SymTCP: Eluding stateful deep packet inspection with automated discrepancy discovery. In *Network and Distributed System Security*. The Internet Society, 2020.
[73] N. Weaver, R. Sommer, and V. Paxson. Detecting Forged TCP Reset Packets. In *Proceedings of the Network and Distributed System Security Symposium, NDSS 2009, San Diego, California, USA*. The Internet Society, 2009.
[74] V. Weber. The Worldwide Web of Chinese and Russian Information Controls, September 2019. https://ctga.web.ox.ac.uk/files/theworldwidewebofchineseandrussianinformationcontrolspdf.
[75] M. Wei. Domain shadowing: Leveraging content delivery networks for robust blocking-resistant communications. In *USENIX Security Symposium*. USENIX, 2021.
[76] P. Winter and S. Lindskog. How the Great Firewall of China is blocking Tor. In *Free and Open Communications on the Internet (FOCI)*. USENIX, 2012.
[77] X. Xu, Z. M. Mao, and J. A. Halderman. Internet censorship in China: Where does the filtering occur? In *Passive and Active Measurement Conference*, pages 133–142. Springer, 2011.
[78] D. Xue, B. Mixon-Baca, V. , A. Ablove, B. Kujath, J. R. Crandall, and R. Ensafi. TSPU: Russia's Decentralized Censorship System. In *ACM Internet Measurement Conference (IMC '22)*, NYC, New York, 2022. ACM.
[79] D. Xue, R. Ramesh, L. Evdokimov, A. Viktorov, A. Jain, E. Wustrow, S. Basso, and R. Ensafi. Throttling Twitter: an emerging censorship technique in russia. In *Internet Measurement Conference (IMC)*, 2021.
[80] T. K. Yadav, A. Sinha, D. Gosain, P. K. Sharma, and S. Chakravarty. Where the light gets in: Analyzing web censorship mechanisms in India. In *Proceedings of the Internet Measurement Conference 2018*, pages 252–264, 2018.
[81] J. York. Websense bars Yemen's government from further software updates. ONI, 2009. https://opennet.net/blog/2009/08/websensebars-yemens-government-further-softwareupdates.
[82] ZMap. ZGrab 2.0, 2022. https://github.com/zmap/zgrab2/.

A ARTIFACTS

To enable reproducibility of our study and the continued monitoring of censorship devices, we open-source our code and data at https://censoredplanet.org/censorship-devices. Our tools are indexed in Zenodo with the following DOIs:

- CenTrace: 10.5281/zenodo.7260106
- CenFuzz: 10.5281/zenodo.7260100
- CenProbe: 10.5281/zenodo.7260104

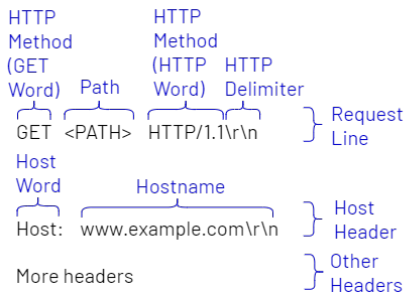

Figure 7: Parts of a HTTP GET request ◇

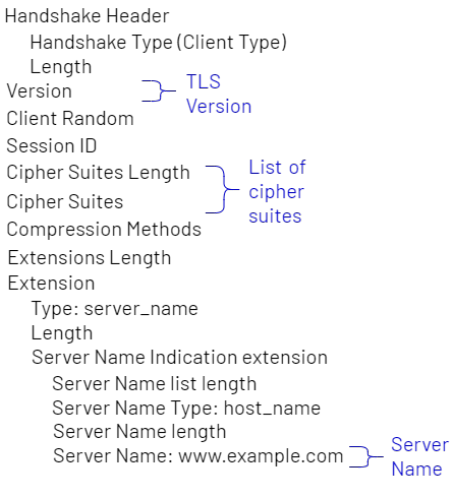

Figure 8: Parts of a TLS Client Hello ◇

B HTTP AND TLS GRAMMARS

In HTTP, CenFuzz applies fuzzing strategies to HTTP `GET` requests, which are the primary target of most devices performing HTTP blocking. Figure 7 shows the structure of a HTTP Version 1 (HTTP/1.1)) GET request specified in multiple RFCs [26–32]. A HTTP request starts with a Method, which specifies the type of request. The most common HTTP Methods are `POST`, `GET`, `PUT`, `PATCH`, and `DELETE`. Following the HTTP Method, the HTTP request path specifies the resource location and the protocol version specifies the version of HTTP used for this request. A HTTP delimiter `\r\n` is then added. Next, a series of HTTP headers may be included. The HTTP Host header is commonly included in all HTTP requests [48], and censors frequently use the Hostname in the Host header as the key for performing blocking. Each HTTP header is delimited with a `\r\n`.

Table 3: A description of each of the features we collect. Features marked with an asterisk are used as *labels* to perform the classification task ⋄

Feature origin	Feature description
CenTrace (4)	Labels from blockpages* Type of blocking (e.g. RST vs TIMEOUT) On-path vs In-path Network features from injected packet Quoted ICMP packet
CenFuzz (6)	Strategies successful in evasion
Banners (5)	Labels from banners* Open ports

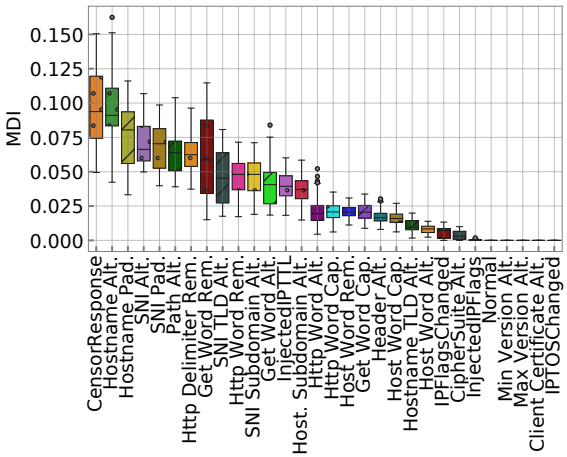

Figure 9: Importance of different device features ⋄

In TLS, CenFuzz applies fuzzing strategies to the TLS Client Hello request, which is frequently targeted by devices performing blocking of HTTPS connections, as all following Client → Endpoint communications are encrypted. Figure 8 shows the structure of an example TLS 1.2 Client Hello request [19, 20, 24]. The TLS packet starts with a Record Header, which specifies the message type, version, and length. A Handshake header then specifies that the message type is a Client Hello message, and the length of the following Client Hello message. A client version field then specifies the TLS version supported by the client (TLS 1.0, 1.1, 1.2, or 1.3), and the client then provides 32 bytes of random data. Next, the client may provide a Session ID and an ordered list of cipher suites supported. The client then provides an ordered list of which compression methods it will support. Following these fields, the client may optionally provide a number of TLS extensions. For our strategies, we always add the TLS server name indication (SNI) extension, which specifies the domain corresponding to the request, and which is used by censorship devices to block the Client Hello.

C FEATURES

Table 3 shows the list of features that we extract from our CenTrace, CenFuzz, and banner grab measurements for our clustering task (§7). Figure 9 shows the relative importance of each feature in our trained random-forest classifier, with 15 repetitions. This data shows that the type of censorship (e.g. whether the middlebox injected an RST, or dropped packets) was very useful in determining the device vendor. Similarly, many of our fuzzing features perform quite well, as well as the TTL of the injected packet, when available.

D CENTRACE VISUALIZATIONS

Figure 10, Figure 11, and Figure 12 show the CenTrace measurements from our client in the United States to endpoints in AZ, BY, and KZ respectively. The red links indicate where the blocking occurs, and the geolocation of each of the nodes are annotated. In AZ, we see that the blocking happens at the link entering into the country, while for BY and KZ, the blocking happens closer to the endpoint AS.

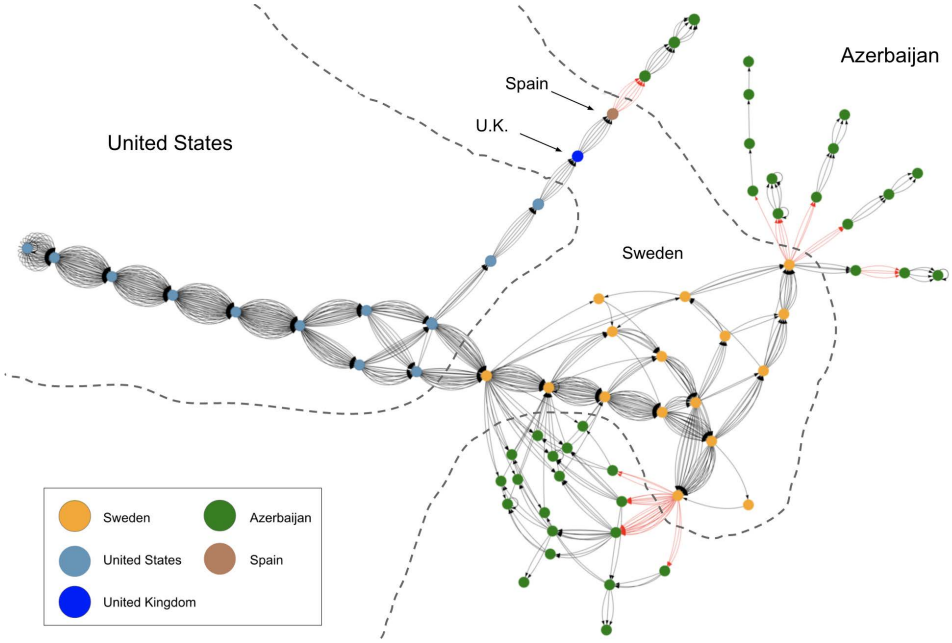

Figure 10: Remote CenTrace measurements in Azerbaijan ◇

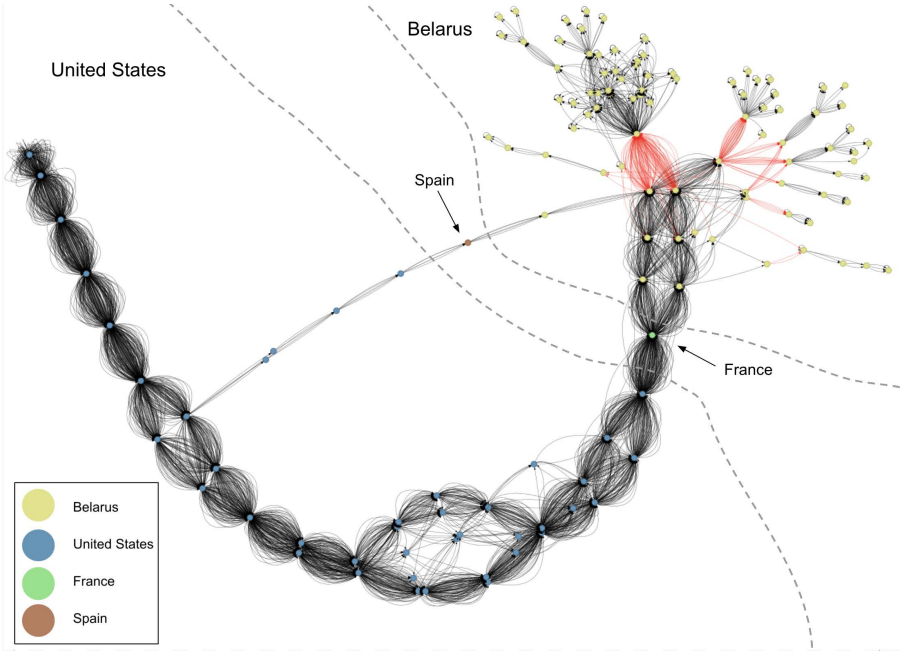

Figure 11: Remote CenTrace measurements in Belarus ◇

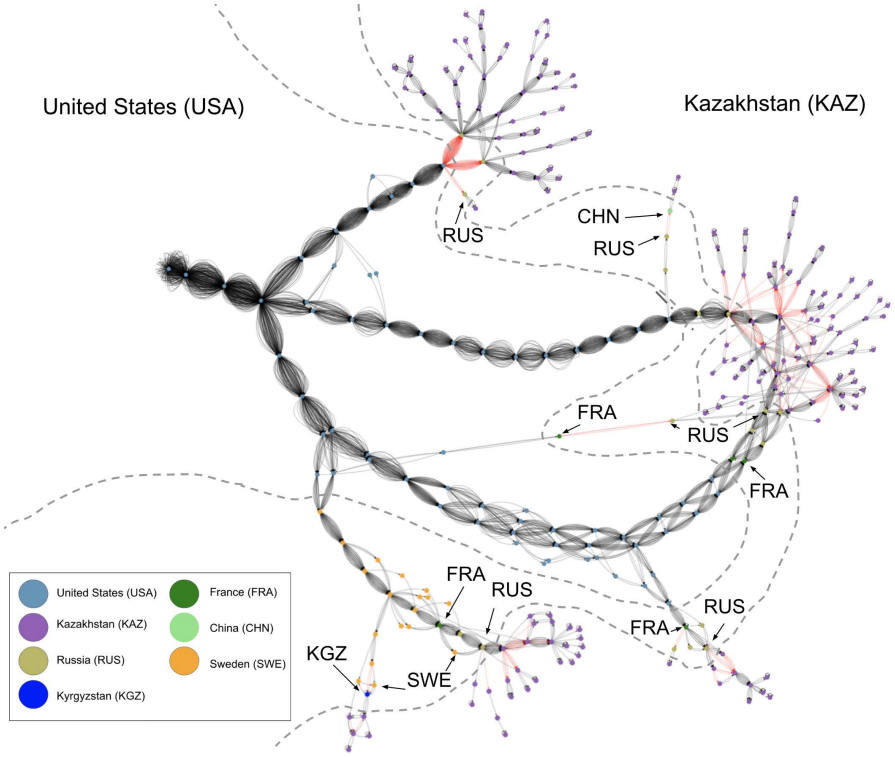

Figure 12: Remote CenTrace measurements in Kazakhstan ◇