Analyzing China's Blocking of Unpublished Tor Bridges

Arun Dunna
University of Massachusetts Amherst

Ciarán O'Brien
University of Massachusetts Amherst

Phillipa Gill
University of Massachusetts Amherst

Abstract

At the end of 2011, China's Great Firewall (GFW) began to block unpublished Tor bridges. Past studies of this blocking have found that the firewall implements both deep packet inspection (DPI) and active probing in order to identify and block usage of the Tor protocol. We build upon the information from previous studies conducted in 2012 and 2015, using a vantage point in China, and bridge relays that we deploy in the US, Canada, and the UK. We determine the extent to which both published and unpublished relays, specifically bridges, are currently blocked by the GFW. We also analyze the active scanners employed by the GFW, and determine the viability of various deployed circumvention methods. We specifically observe that a simple technique to identify and not respond to the GFW's scanners is effective in keeping a bridge relay from being blocked. We conclude by discussing the current circumvention methods, and how best to implement these circumvention methods to improve the accessibility of the Tor based on our measurements.

1 Introduction

Of the countries that perform censorship, China has long been on the forefront in terms of its development and deployment of techniques to identify and filter censored content [3, 10, 15, 21]. One of the key protocols filtered by China is the Tor anonymity system. Indeed, for the past five years, China and Tor have been in an arms race to block or stay one step ahead of the censor, respectively [9, 11–13, 17, 20]. As Tor develops new approaches to circumvent Chinese filtering (*e.g.,* unpublished bridge relays [2], covert channels [11, 13, 17]) the Chinese filtering apparatus evolves to detect and block Tor [6, 20]. The ongoing nature of this arms race makes measuring how China detects and blocks Tor a moving target.

In this study, we revisit the question of how China's Great Firewall (GFW) detects and filters Tor, following in the line of related work carried out over the past six years [6, 20]. To gain access to a vantage point in China, we use a virtual private server (VPS) located in a Chinese cloud provider's infrastructure. We also deploy three servers that act as bridge relays to allow us to monitor how the GFW probes/scans suspected bridge nodes.

We use this infrastructure to answer the following questions about how the GFW blocks Tor. Specifically, we investigate to what extent the public Tor network is reachable (§4.1); how unpublished (bridge) relays are blocked (§4.2); and how GFW scanners have changed from previous studies (§4.3). Finally, we investigate how Tor users can circumvent today's GFW and investigate approaches other than the popular pluggable transport approach to keep the GFW from detecting Tor bridges.

Table 1 compares our results to two prior studies on how the GFW blocks bridge relays. We note that many aspects of bridge relay blocking are similar to 2015 with the notable difference that now relays are blocked on a per IP basis. This requires us to develop a new method to fingerprint scanners in our study (Section 4.3). We also observe that bridge relays remain blocked for 12 hours, after which they are re-scanned by the GFW and unblocked if they are no longer acting as relays. Interestingly, we make a similar observation with public relays which are blocked for 12 hours after the Tor service is disabled. This suggests that the GFW may be using a common list to monitor Tor relays whether they are discovered via DPI or by scraping the Tor Consensus.

In fingerprinting 934 unique scanner IPs (Section 4.3), we find that all active scanners are located in China. Scanner traffic is distinguishable from legitimate Tor traffic through several TCP packet options such as MSS and window scaling (Table 2). Each IP only conducts one or two scans, which details the uniformity of the scanning system and shows measures taken to avoid circumvention by blacklisting scanner IPs.

Finally, we investigate circumvention methods (Sec-

	Year		
	2012	**2015**	**2018**
Block Method	(IP, Port)	(IP, Port)	IP
Block Duration	12 hours	12 hours	12 hours
Scanning Queues	15 minutes	Instantly	Instantly
Scanner Distribution	Not Uniform	Uniform	Uniform
Most Common Scanner	202.108.181.70	202.108.181.70	111.202.242.93
Common Scanner ASes	4134, 4837, 17622	4134, 4837, 7497	4134, 4837, 17676

Table 1: Comparison of results from GFW studies in 2012 [20], 2015 [6], and now 2018.

tion 5). We find that the two most popular pluggable transports (Meek [7] and Obfs4 [18]) are still effective in evading GFW's blocking of Tor (Section 5.1). However, we also observe that having a bridge relay not respond to scans from the GFW appears to be an effective and low-cost method to avoid having bridge relays from being blocked based on scans (Section 5.2).

2 Background

Tor is the de facto network for providing anonymous communication on the Internet [5]. However, the fact that Tor conceals the destination of a network connection has also made it an attractive tool for individuals to circumvent Internet censorship around the globe [8]. This has led Tor developers and China's censorship apparatus–termed the Great Firewall (GFW)–to enter into an arms race to remain one step ahead of each other. One notable strategy employed by Tor developers has been the use of Tor bridges, unpublished relays that users in China can use to gain access to the Tor network. By using relays that are not published along with the list of Tor relays, the Tor developers aim to increase the effort required by the GFW to block these connections. The GFW has responded to the deployment of Tor bridges by combining deep packet inspection (DPI) with active scanning to identify these unpublished relays [6, 20].

When a client makes an initial connection to an unpublished Tor bridge the GFW uses DPI to identify that the connection matches signatures for Tor traffic. After detecting the presence of Tor traffic, the firewall deploys multiple scanners to test whether the relay is actually running the Tor service. We define a scanner as an IP address, which is geo-located to China, that attempts to connect to our relay and is not a legitimate Chinese user. By preventing the public listing of our relays, we ensure that all traffic received from China, outside of our client, is from a scanner. If these scanners detect the Tor service, then traffic is blocked to/from the bridge.

Prior work has characterized these scanners, how long they scan for, which IPs scan the suspected bridge relays, and for how long the relay will be blocked. We use our experiments to revisit these prior results. The ongoing arms race between Tor developers and the GFW requires ongoing monitoring, as behaviors of both actors evolve over time. In this study, we revisit results from 2012 [20] and 2015 [6], and highlight differences in how the GFW detects Tor bridges.

3 Methodology

We now describe our method to characterize blocking of Tor by the GFW. Unless otherwise specified, all pings are completed using both TCP and ICMP, and pings to specific ports use TCP.

3.1 Vantage points

We use a client machine as well as three bridge relays we deploy ourselves in different geographic locations. All systems in the experiment are running on Ubuntu 16.04 LTS or Ubuntu 14.04 LTS due to easy deployment and relative uniformity. No known filters are applied to our bridges during the experiment.

Bridges. We deploy three bridge relays running on VMs. These bridges are located at UMass Amherst in the Northeast United States, in OVH's data center in Montreal, Canada, and in Amazon EC2 in London, UK.

Tor client. We deploy a client machine on a virtual private server (VPS) in a cloud provider's infrastructure in Shanghai, China.

3.2 Testing Tor reachability

We use the Tor Connection Initiation Simulator (TCIS), developed by Winter and Lindskog [20] to test reachability of the Tor network and our Tor bridges from our client located in China. This simulator allows us to initiate Tor connections in a quick and lightweight manner, instead of running a full Tor client. When appropriate, we utilize the full Tor client to test circuit constructibility.

We perform the following tests from our client:

Attempt to retrieve the Tor consensus. Our client attempts to connect to the eight relays hosting Tor consensus data. It then attempts to download the Tor consensus from these relays if a successful connection is made.

Attempt to connect to published relays. We use TCIS to attempt to initiate Tor connections with the set of published relays (if we cannot retrieve the consensus data from Tor directly, we SFTP it to our client for testing).

Monitoring blocking of published relays. To understand how fast published relays are blocked and for how long they remain blocked we deploy two relays that are submitted to the Tor consensus to be published. These relays are hosted in the same networks as our Canada and US bridge nodes, but on different servers. We block connections on the Tor port originating from China towards these relays to isolate blocking caused by the GFW scraping Tor Consensus data.

Attempt to connect to our bridge relays. We also attempt to initiate Tor connections with our own bridge relays so that we can observe how the GFW reacts to new Tor connections to previously unknown bridge relays.

Monitoring for scanners. We instrument our bridge relays using tcpdump [19] to observe scanning performed by the GFW after our client in China attempts a Tor connection with our bridges.

Pinging relays and bridges. After the client attempts to initiate a Tor connection with a relay or bridge, it proceeds to ping the relay or bridge for up to 48 hours to observe for how long the relay or bridge is blocked.

Pinging from our bridges. We also perform ping measurements from our bridges to the client in China to observe whether blocking of Tor traffic is bidirectional.

3.3 Limitations

There are a few possible limitations with our research methodology due to the nature of China's network and their firewall.

Different treatment of bridges in different networks. The GFW may treat bridges in different locations differently. We deploy three bridge relays in different networks and geographic locations to mitigate this impact.

Location of the client in China. We do not have much control over the location of our Chinese client due to the difficulty of renting a server in China (business license, language barriers, etc.), which is why our Chinese server is located in Shanghai.

Occasional packet drops. We notice periods of high packet drops on our Chinese client. We do not consider measurements during these time intervals in order to remove any noise caused by potential network unreliability.

4 Results

In this section, we discuss the key results of the study. We determine how the firewall blocks both published relays (§4.1) and unpublished relays (§4.2). We determine the length of these blocks (§4.2), and find that Chinese scanners are used in the process. This leads us to perform fingerprinting of scanners (§4.3).

4.1 Reachability of the Tor Consensus and published relays

We check accessibility of relays hosting the Tor Consensus and relays published in the Tor Consensus file. This serves as a sanity check to ensure that our vantage point in China is indeed subject to filtering by the GFW. We attempt to connect to the eight relays hosting the Tor Consensus, used by the Tor client to download the latest list of active Tor relays. As expected, we find that the client is unable to connect to these relays which we observe being blocked via injected TCP reset packets.

To test connectivity to relays listed in the Tor Consensus file we download the Tor consensus file and transfer it to the Chinese server via SFTP. We then attempt to connect to each public relay using TCIS. We observe the initial connection attempt is blocked with the GFW sending a TCP reset packet to the client. After this initial connection attempt, we ping (including UDP) the published relay and observe no replies, indicating that traffic to this relay is being dropped. This observation confirms observations from prior studies [6, 20]. We observe a small number, less than 0.2%, of pings do make it through the GFW, but we note that this occurs very infrequently and we do not observe any trends in these successful pings.

Duration of published relay blocking. To understand the mechanism behind blocking of published relays, we deploy two published relays of our own which block Tor connections from China. This means that the relay is listed on the public relay consensus, but cannot be scanned by scanners nor connected to by Chinese clients. Once the relay is published, we ping the relay every 5 seconds on both the randomly selected Tor port and a web service on port 80 from our client in China to monitor its reachability. We find that on average, our published relays are added to an IP blacklist after 10 minutes.

Stopping the Tor service, which removes the IP from the consensus, does not result in the relay being unblocked immediately. We observe that our published relays remain blocked for an additional 12 hours. We observe a scan attempt from the GFW prior to our former relay being removed from the blacklist (observed via successful pings from our Chinese client).

4.2 Blocking of Bridge relays

As mentioned in Section 3.2, we deploy three bridge relays in three different geographically distributed networks. Our Chinese client attempts to connect to these bridges using TCIS. Similar to published relays, we observe that an initial connection attempt is blocked using a TCP reset packet[1] and after some amount of time the bridge is completely blocked. This blocking is triggered by scans of the bridge relays performed as part of the GFW's operation.

The scanning occurs on multiple ports: the port on which the initial Tor connection is observed, and common ports such as 80 and 443. Additionally, we see scans on "similar" ports to the initial port such as those off by two. Scanning begins almost instantaneously after the Tor connection attempt is made, and after about a minute, the scans completely stop.

Blocking at the level of an IP address. We observe the GFW blocking connections at the level of IP vs. using (IP, port) tuples. We observe this by performing pings from our Chinese client to our bridges on a variety of ports. We perform pings in both directions between our client and bridge relays and observe that the blocking behavior is unidirectional. Traffic is allowed from our client to our bridge, but not from our bridge to our client. We attempt to deploy services across other ports to determine if the firewall recognizes that the server was running more than just Tor, such as web services, but the firewall blocks the IP address completely regardless. This highlights the potential collateral damage if Tor bridge relays are hosted in cloud computing facilities where they may be colocated with Web services.

Duration of Bridge relay blocking. Disabling the Tor service does not result in immediate unblocking of the server. Similar to blocking of published relays, we observe the bridge relay is blocked for 12 hours after it is scanned. After 12 hours, the bridge is scanned again and becomes accessible if the Tor service is no longer running. This suggests that IPs gathered from the Tor consensus are put into the same blacklist as bridge relays. We also notice that if we attempt a Tor connection sometime during the block to any of the blocked bridges, the block duration renews to 12 hours from the last attempted Tor connection.

Attempts to conserve resources. Attempting to initiate a Tor connection to a port not running Tor does not trigger scanning, so a Tor connection needs to succeed from the client to attract scanners. The lack of scanning during the 12 hour block and the short scan at the 12 hour interval, as well as the IP blocking rather than (IP, port) tuple blocking, are indications of the firewall trying to save resources. This is probably related to the sheer increase in the number of Tor users and relays since 2012, from less than 1 million users per day in 2012 to upwards of 2 million users per day in 2018.

[1] Since we control the bridge relay, we confirm that the TCP reset packet is sent both to the client and the bridge relay.

4.3 Fingerprinting GFW scanners

We now characterize scanners we observe through the experiments described in the previous sections.

Unlike prior work, we face the challenge that the GFW blocks relays based on IP address and not (IP, port)-tuples. This means that once a connection attempt is made to our server, and a scanner is observed, our IP address will not be scanned again for 12 hours. Given that we have two bridge relays at our disposal, this would drastically reduce the scanners we can observe, relative to prior work that could initiate connections on multiple ports of their bridge relays.

We get around this challenge by having one bridge relay drop packets from scanners, which is feasible because we only expect legitimate connections to our bridge to come from our own test client. We elaborate on this idea and how it may be used to evade scanner-based blocking in Section 5.2. Dropping the scanner's traffic means that the scanner does not confirm that our relay is a bridge relay and does not add the relay to the black list. This allows us to repeatedly initiate Tor connections to the relay from our client and observe scan attempts.

We initiate connections to our bridge relay every 10 seconds over a period of 44 hours from our client in China. This allows us to observe a total of 934 unique IP addresses scanning our relays. We assign each IP address to one scanner to perform our analyses.

Comparison with prior scanners. We revisit the fingerprinting of scanners from the 2015 paper. We compare qualities of captured SYN packets in Table 2. Similar to 2015, all scanners we attract are located in China. We determine this through mapping IPs to countries with MaxMind GeoLite2 database [16], which is accurate enough for country mappings. Given the similarities of packet qualities and likelihood of colocation from TTL values, we agree with the hypothesis suggested in the 2015 paper [6] that the network is a distribution of proxies that forward packets from a centrally controlled system.

A separate scanning infrastructure? We observed 5% of SYN packets originating from 111.202.242.93, and the other 95% being almost uniformly distributed in origin. Upon closer inspection, we observe that unlike the other scanners, packets from 111.202.242.93 have an MSS of 1368 as opposed to 1400 seen from the other scanners. In the process of attracting scanners, we respond to SYN packets with an MSS of 1368 normally,

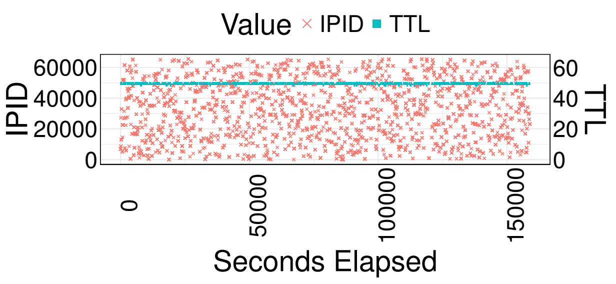

Figure 1: IPIDs and TTLs of SYN packets received by our relay from scanners over 44 hours.

and find that the results of scans from 111.202.242.93 do not result in our bridge getting blocked. This seems to indicate that this scanner may be operating separately from the main scanning and blacklisting infrastructure.

Trends in IPIDs and TTLs. To determine the underlying structure of the scanning system, we examine IPIDs and TTLs of scanning packets, shown in Figure 1. While we do not observe a trend in IPIDs, we find that the similarity in TTL values indicates that scanners are colocated. In comparing these TTLs with TTLs observed on reset packets from the GFW, we cannot determine whether the reset injectors are colocated with active scanners, with reset injectors having a TTL of 44 while scanners have a TTL of 48-50.

	Year	
	2015	**2018**
TTL Range	46 - 51	48 - 50
MSS Values	1400, 1460	1368, 1400
Window Scaling	7	7
Permit Selective ACKs	Yes	Yes
TCP Timestamp	Yes	Yes
No Operation	Yes	Yes

Table 2: Scanner TCP SYN packet qualities comparison.

Prevalence of specific scanner IPs. Additionally, we determine that out of the 934 IP addresses we see, 908 IPs conducted one scan and 26 IPs conducted two scans. No higher amount of scans were exhibited, detailing the uniformity of the system. Spreading the scanning across IPs may be a way of avoiding IP-based detection of scanners. Interestingly, we observe that the most frequent scanning IP observed in prior work (202.108.181.70 [6,20]) does not appear anywhere in the set of scanners we observe.

5 Circumvention

We now consider how well existing circumvention schemes, specifically Tor pluggable transports evade the GFW's blocking of Tor. We also describe a technique that attempts to ignore traffic from GFW scanners to avoid having the bridge IP added to a block list.

5.1 Pluggable transports

We consider how well two modern pluggable transports evade the GFW.

meek: *meek* is a pluggable transport that is actively maintained and recommended by The Tor Project for Chinese users to access the Tor network [7]. It utilizes domain fronting, routing traffic through a CDN or similarly popular service before the traffic is sent to a bridge. We test *meek* with the built-in public Azure server and successfully establish a Tor circuit. We are able to utilize Tor without issues through this method. We also test the built-in Amazon method, and determine that it is inconsistent in whether or not it will establish a circuit. Occasionally, we are able to establish a circuit, but most circuits do not successfully complete, failing at seemingly random points in the process.

We also test *meek* on our own bridge which, while successful, requires TLS. Without TLS, the firewall blocks the bridge, and subsequently purges it from the IP blacklist within a few minutes after the *meek* bridge is stopped. So, we theorize that there is a separate *meek* blacklist, and China is actively attempting to determine how to block *meek* users.

obfs4: *obfs4* is a pluggable transport that relies on obfuscation rather than domain fronting [18]. Using *obfs4*, we are able to establish Tor circuits on our own bridges, but are unable to connect to public bridges. This is likely due to public bridges being recorded and blacklisted by the firewall.

5.2 Mitigating scanners

While pluggable transports are successful in circumventing the firewall, we note that they must be deployed on both the server and the client. Inspired by the method we use to collect information about scanners in Section 4.3, we propose that bridge relays simply drop packets from the GFW's scanners, basically responding as if the bridge is not running a Tor service.

To do this, we write a series specific rules using *iptables* in order to drop packets from Chinese scanners. We differentiate scanner packets from legitimate packets through our observations from Table 2, specifically relying on the MSS value. We use a rule to drop incoming Tor packets with an MSS of 1400. Further investigation would be needed to analyze potential false positives, but related work [1] indicates that an MSS value of 1400 is generally infrequently observed. We note that this method of dropping scan traffic successfully keeps our bridge relays from being blocked and allows our client in China to maintain access to the bridge.

5.3 Discussion

We find that the easiest method for users to circumvent the firewall appears to be *meek*, as it can be used out-of-the-box by Chinese users. However, if the Tor Project is able to limit the distribution of bridge addresses, then rejecting scanners would be equally viable and significantly more cost effective. Additionally, *obfs4* was easier to deploy on an unpublished bridge than *meek*, as certificate management was not required. In practice, rejecting scanners would be more effective than *obfs4* as only the relay has to implement scanner rejection, whereas both the relay and the client have to be running pluggable transports.

6 Conclusion

Through our work, we gather an understanding of how the modern GFW blocks both unpublished and published relays. We compare our results to the results found in previous work, namely in 2012 [20] and 2015 [6], to give an updated perspective into how the firewall has changed over time. We find that the firewall blocks relay IPs through DPI, and to minimize collateral damage, purges these IPs when they are detected as no longer running Tor. We see that all scanners employed by the firewall are located in China, and that likely the large spread of scanner IPs is actually a diverse proxy network forwarding scanner packets from a centralized source.

We conclude with an analysis on circumvention of the current GFW, and find that the most effective method of circumvention for the average user is the pluggable transport *meek*, due to its inclusion in the Tor browser bundle and ease of use. However, due to the large costs associated with running *meek* domain fronting servers [7] as well as the recent collective disabling of domain fronting by cloud providers such as Google [4] and Amazon [14], we find that the most cost effective method would be to reject scanner packets and continue to work on improving methods of distributing unpublished bridge IP addresses. This would also be easier for bridge operators, as the rejection feature could be implemented natively in Tor rather than being added as a pluggable transport.

In future work, we hope to develop a more effective method of circumvention, retaining both the ease of client use of *meek*, and cost efficiency and ease of deployment of rejecting scanners. We also suggest developing alternative methods for distributing Tor clients to Chinese users, and advocate for developing better unpublished bridge IP distribution methods for use with scanner rejection or *obfs4*. Through these methods, we aim to combat censorship by increasing the availability of Tor for average Chinese Internet users.

We publish all datasets and code to maintain reproducibility, obtainable at `calipr.cs.umass.edu/research`.

References

[1] ALCOCK, S., AND NELSON, R. An Analysis of TCP Maximum Segment Sizes. `https://wand.net.nz/sites/default/files/mss_ict11.pdf`.

[2] Tor: Bridges. `https://www.torproject.org/docs/bridges`.

[3] CRANDALL, J., ZINN, D., BYRD, M., BARR, E., AND EAST, R. ConceptDoppler: A Weather Tracker for Internet Censorship. In *Proceedings of the 14th ACM Conference on Computer and Communications Security* (2007).

[4] Google ends "domain fronting," a crucial way for tools to evade censors. `https://www.accessnow.org/google-ends-domain-fronting-a-crucial-way-for-tools-to-evade-censors/`.

[5] DINGLEDINE, R., MATHEWSON, N., AND SYVERSON, P. Tor: The Second-generation Onion Router. In *Proceedings of the 13th Conference on USENIX Security Symposium* (2004).

[6] ENSAFI, R., FIFIELD, D., WINTER, P., FEAMSTER, N., WEAVER, N., AND PAXSON, V. Examining How the Great Firewall Discovers Hidden Circumvention Servers. In *Proceedings of the ACM Internet Measurement Conference* (2015).

[7] FIFIELD, D. Meek Pluggable Transport. `https://trac.torproject.org/projects/tor/wiki/doc/meek`.

[8] FIFIELD, D., LEE, L., EGELMAN, S., AND WAGNER, D. Tor's Usability for Censorship Circumvention. In *HotPETs* (2015).

[9] GEDDES, J., SCHUCHARD, M., AND HOPPER, N. Cover your ACKs: Pitfalls of Covert Channel Censorship Circumvention. In *Proceedings of the 20th ACM Conference on Computer and Communications Security* (2013).

[10] China's Green Dam: The Implications of Government Control Encroaching on the Home PC, 2009. OpenNet Initiative Bulletin `https://opennet.net/sites/opennet.net/files/GreenDam_bulletin.pdf`.

[11] HAHN, B., NITHYANAND, R., GILL, P., AND JOHNSON, R. Games Without Frontiers: Investigating Video Games as a Covert Channel. In *IEEE European Symposium on Security and Privacy* (2016).

[12] HOUMANSADR, A., BRUBAKER, C., AND SHMATIKOV, V. The Parrot is Dead: Observing Unobservable Network Communications. In *Proceedings of the IEEE Symposium on Security and Privacy (OAKLAND)* (2013).

[13] HOUMANSADR, A., RIEDL, T., BORISOV, N., AND SINGER, A. I Want my Voice to be Heard: IP over Voice-over-IP for Unobservable Censorship Circumvention. In *The 20th Annual Network and Distributed System Security Symposium (NDSS)* (2013).

[14] MACCARTHAIGH, C. Enhanced Domain Protections for Amazon CloudFront Requests. `https://aws.amazon.com/blogs/security/enhanced-domain-protections-for-amazon-cloudfront-requests/`.

[15] MARCZAK, B., WEAVER, N., DALEK, J., ENSAFI, R., FIFIELD, D., MCKUNE, S., REY, A., SCOTT-RAILTON, J., DEIBERT, R., AND PAXSON, V. An Analysis of China's "Great Cannon". In *Proceedings of the USENIX Free and Open Communication on the Internet (FOCI) workshop* (2015).

[16] MaxMind GeoLite2. `https://dev.maxmind.com/geoip/geoip2/geolite2/`.

[17] MOGHADDAM, H. M., LI, B., DERAKHSHANI, M., AND GOLDBERG, I. SkypeMorph: Protocol Obfuscation for Tor Bridges. In *Proceedings of the 19th ACM Conference on Computer and Communications Security* (2012).

[18] The obfourscator. `https://github.com/Yawning/obfs4`.

[19] TCPDump & LibPCAP. `www.tcpdump.org`, accessed May 2018.

[20] WINTER, P., AND LINDSKOG, S. How the Great Firewall of China is Blocking Tor. In *Proceedings of the USENIX Free and Open Communication on the Internet (FOCI) workshop* (2012).

[21] XU, X., MAO, Z. M., AND HALDERMAN, J. A. Internet Censorship in China: Where Does the Filtering Occur? In *Proceedings of the Passive and Active Measurement (PAM) conference* (2011).