SoK: Towards Grounding Censorship Circumvention in Empiricism

Michael Carl Tschantz*, Sadia Afroz*, Anonymous‡, and Vern Paxson*†
*International Computer Science Institute †University of California, Berkeley

***Abstract*—Effective evaluations of approaches to circumventing government Internet censorship require incorporating perspectives of how censors operate in practice. We undertake an extensive examination of real censors by surveying prior measurement studies and analyzing field reports and bug tickets from practitioners. We assess both deployed circumvention approaches and research proposals to consider the criteria employed in their evaluations and compare these to the observed behaviors of real censors, identifying areas where evaluations could more faithfully and effectively incorporate the practices of modern censors. These observations lead to an agenda realigning research with the predominant problems of today.**

I. Introduction

Censorship circumvention research seeks to develop approaches for facilitating access to banned Internet resources, a domain with a fundamentally adversarial nature arising from the ongoing interactions between circumventors and censors. Both parties find themselves locked in an arms race where each side must manage tradeoffs between efficacy and expenditure. These tradeoffs continually evolve in subtle ways as new technologies change the costs of various approaches.

Given this complexity, undertaking sound evaluation of potential circumvention approaches proves both crucial and difficult. Sound evaluation is crucial since, due to limited resources, the developers of circumvention approaches cannot implement and deploy every prospective approach; they need criteria for selecting the most promising. It is difficult, on the other hand, because unidentified weaknesses in an approach offer potential openings to censors, but worst-case analyses that presume censors will necessarily exploit such vulnerabilities ignore the realities of censors who aim to avoid blocking profitable traffic while staying within their budget constraints. Soundly incorporating these realities into evaluations requires grounding in empirical observations of real censors.

While the evaluation sections of research papers provide some insight into the promise of a given circumvention approach, each paper employs its own evaluation methodology, typically selected with the capabilities of the approach in mind but often not balanced against realistic models of censors. Furthermore, such approach-oriented evaluations make it difficult to compare across different approaches, or to determine how well an evaluation predicts real-world performance. Prototyped approaches meeting their own evaluation criteria can succumb to vulnerabilities not considered by their evaluation (e.g., [1]).

Approach. To address this disconnect between evaluation and the actual operating conditions of censorship circumvention approaches, in this work we seek to ground the evaluation of circumvention approaches in empirical observations of real censors. To do so, we systematically compare the behaviors of real censors to the evaluation criteria used by circumvention-approach designers.

Our work systematizes the evaluation of approaches for censorship circumvention in four ways:

1) We collect data on real-world attacks to show the current state of censorship practice (Sections II and IV),
2) We survey circumvention approaches and their evaluations to illuminate the current state of evaluation (Sections V and VI, respectively),
3) We compare the evaluations designed to assess the difficulty of blocking an approach to the actual actions of real censors (Section VII),
4) We point to open research problems whose resolution will improve evaluation (Section VIII).

Scope. We focus our discussion on censorship by governments attempting to prevent subjects from accessing particular web resources outside the government's jurisdiction. The censor seeks to detect and disrupt banned traffic by placing monitors at the edges of their network—just as customs inspectors intercept and examine physical goods at international borders.

We also limit the types of circumvention we consider to *channel-based* approaches that (1) bypass country-level censors that monitor network traffic between two end points, (2) communicate with resources outside the censors' borders, and (3) enable low-latency connections (roughly, fast enough for web browsing). Our scope excludes concerns such as internal censorship of newspapers or disruption of entirely domestic communication. Even so, the remaining space of censor activity and circumvention approaches is large, with 56 approaches or evaluations of approaches [2–57] falling within our scope. Figure 1 shows a generic model of censorship and circumvention under the scope we use.

Overview. We first take a detailed look at the arms race between Tor and China as an illustration of the cat-and-mouse nature of censorship and circumvention (Section II). After covering related work (Section III), we broaden our view by examining censorship incidents involving other channel-based circumvention approaches (Section IV).

‡ This coauthor chooses to forgo identification in protest of the IEEE Security and Privacy Symposium's acceptance of support from the US National Security Agency. While it pains us for his significant contributions to the work to go unrecognized here, we respect the heartfelt principles that led him to his position.

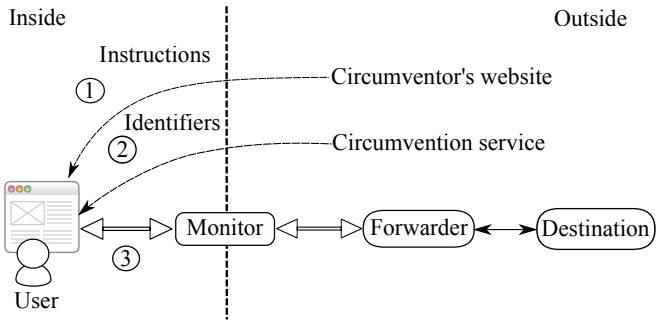

Fig. 1. An illustration of the type of censorship we study. First, censored users within a censor's jurisdiction gather information about how to use an approach, which may include a program download. Second, the users gain various identifiers, such as IP addresses and passwords. Third, they run their traffic through a client-side program that applies various transformations to hide the true destination and content. In the case of a banned destination, the approach sends the obscured traffic to some allowed destination that acts as a forwarder to the real destination.

Next, we provide a survey of channel-based censorship circumvention (Section V) and its evaluation (Section VI). Unlike other surveys on circumvention [58–60], we do not focus on comparing the approaches themselves, but rather on comparing their evaluations. We enumerate the criteria that the developers of each approach used in their evaluations. We find little commonality in the evaluation methods employed. While some diversity is to be expected given that approaches differ in goals and intended deployment environments, we find no globally organizing principles guiding the selection of evaluation criteria.

We focus on the criteria related to detecting circumventing traffic and compare them to the actions of real censors (Section VII). We observe that system designers tend to emphasize censor capabilities that may become important in the future, but not seen in practice today, with little assessment on actual detection techniques used by current censors. In particular, we identify three **disconnects** between practice and research:

1) Real censors attack how users discover and set up channels, whereas research often centers on channel usage,
2) Real censors prefer cheap passive monitoring or more involved active probing, whereas research often looks at complex passive monitoring and traffic manipulations at line speed, and
3) Censors favor attacks that do not risk falsely blocking allowed connections due to packet loss, whereas research considers many less robust attacks.

We end with a research agenda to realign the evaluation of circumvention approaches with the actions of real censors (Section VIII). We propose augmenting prior approach-specific methods of evaluation with a new methodology for creating evaluation criteria and interpreting results, and with tools for aiding evaluators.

Throughout, we provide recommendations on how to improve evaluations. However, we do not ultimately end with any list of "correct" evaluation criteria, nor actually evaluate any approaches. We believe that the range of approaches used is too wide for any one list to cover all use cases, or for a single study to attempt to comprehensively rank them. Rather, we hope to provide a systematic method of thinking about evaluation that will guide evaluators of circumvention approaches in their selection of appropriate criteria on a case-by-case basis.

Also, we do not intend for this work to discount the utility of forward-looking studies that anticipate the more advanced censors of the future. Rather, we aim to make the tradeoffs clear: some approaches considered by research are likely years ahead of the point where their overhead is justified by actual censors effectively blocking less advanced methods; meanwhile, censors block deployed approaches using simple attacks. We hope this observation inspires the research community towards also providing tools for preventing simpler attacks, which would yield immediate benefits for many real users.

We make details and our database available at
http://internet-freedom-science.org/circumvention-survey/

II. Illustrating the Problem Space: Tor and the Great Firewall

The problem space we consider has both disparate aspects and a complicated arms-race-driven evolution. In this section we frame the space through the lens of the Tor anonymity system and the "Great Firewall" (GFW), the primary national censorship apparatus of China. The conflict between these two actors is representative of the larger world of censorship and circumvention, and offers us a way of introducing different notions and the associated terminology we will use in our discussion, as well as providing some of the grounding in real-life censorship that underlies many of our perspectives.

Tor provides a convenient focus due to the relatively extensive documentation of its censorship and circumvention counter-responses. We mined blog posts, bug reports, and the Tor Project's public documentation to identify the censorship events that underlie our narrative. A progression emerges: Tor, which was not originally designed for circumvention, is used to evade the GFW. The censor blocks the Tor website and the servers that make up the anonymity network; Tor responds with mirrors and secret entry servers. The censor begins to identify Tor by protocol features; Tor deploys protocol encapsulation to hide those features. The censor starts actively scanning for Tor servers; Tor introduces protocols immune to scanning.

In this recounting, in our terminology the GFW plays the role of a *censor*: a government entity that disrupts access to certain Internet resources outside its jurisdiction (in this case, China). The censor employs *monitors*, traffic filtering devices at the edge of the network. Tor and its users are *circumventors* who seek to evade the censor's controls. We divide circumventors into *users*, those within the censor's jurisdiction who try to access blocked content; and *advocates*, those who develop, deploy, and maintain systems that enable circumvention. We term any means of evading the censor's blocking a *censorship circumvention approach*, or "approach" for short. Tor offers not a single approach, but multiple "pluggable" approaches,

each with advantages and disadvantages. A user's connection to the Tor network (however accomplished) is an example of what we call a *channel*. The Tor network itself is an example of a *forwarder*. A channel disguises user traffic so that it may reach a forwarder that then passes the traffic on to its ultimate destination. Refer to Figure 1 for the relative positioning of user, monitor, and forwarder.

The Tor network began operating in 2003 [61]. Designed for anonymity, early Tor had major deficiencies as a circumvention approach. Unsurprisingly, the first action the GFW took against Tor was simple: in 2008 it blocked the www.torproject.org website [62]. In response, advocates deployed website mirrors and email-based software distribution mechanisms.

The raw Tor protocol—today often called "vanilla" Tor to distinguish it from the more resistant approaches that followed—has many distinguishing features that make it easy to detect. We call any such distinguishing feature a *vulnerability*, by analogy to software security. Similarly, we refer to the effective leveraging of a vulnerability by the censor for detection and taking action against circumvention as an *exploit*. We call a chain of exploits aimed at disrupting a channel an *attack*. One of vanilla Tor's biggest vulnerabilities is its hardcoded list of public directory authorities, from which users download the similarly public list of relay IP addresses. In late 2009 the GFW exploited this vulnerability by blocking the IP addresses of directory authorities and relays [63, 64]. The steps of the attack in this case are (1) downloading the list of relays and adding their addresses to a blacklist; (2) dynamically blocking any access to those addresses when observed in a traffic stream.

In response to the blocking of its directory requests and relays, Tor introduced "bridges", secret relays without publicly listed addresses. (Bridges had in fact been prepared earlier, in 2007, in anticipation of this type of blocking [65, 66]—the arms race is not merely reactive.) Users must first learn the IP address of a bridge in some out-of-band fashion, for example by email, an instance of what we term an *identifier distribution mechanism* (IDM), the means by which a user learns the information required to establish a connection to a forwarder. After blocking the public relays, the GFW went after bridges by attacking the IDM, enumerating bridges from the centralized bridge database one by one [63, 64, 67].

Private bridges—those distributed by word of mouth rather than through a centralized database—remained unblocked for a time. Even with secret bridge addresses, though, Tor remains vulnerable to *deep packet inspection* (DPI), protocol-aware filtering that considers application-layer semantics. Tor uses TLS in fairly distinctive ways that make it relatively easy to detect. For example, the firewall began to identify the use of Tor by looking for Tor's distinctive list of TLS client cipher suites in 2011 [68]. For a time, Tor developers responded with incremental refinements to make Tor's use of TLS less distinctive [69]. A more lasting solution came in the form of "pluggable transports", modular circumvention approaches that form an additional layer around Tor TLS to protect it from protocol fingerprinting. Pluggable transports use a variety of techniques to hide the fact that Tor is in use.

The GFW's DPI came with a twist, however: The operators of the firewall used DPI detection of Tor as a cue to employ *active probing*, posing as a user and connecting to suspected circumvention forwarders to block them by address if confirmed. In late 2011 the first evidence of active probing against Tor appeared [70]. Later, the censors employed probing for certain pluggable transports, as well as non-Tor channels such as virtual private networks (VPNs) [71]. The latest pluggable transports are designed to resist active probing by incorporating per-forwarder secrets or by co-locating forwarders with non-circumvention-related services.

This brings us to the state-of-the-art for Tor-related circumvention: censors that employ website and IP address blocking, IDM disruption, deep packet inspection, and active probing; and circumvention approaches that use secret IP addresses, encrypt their payloads, and resist active probing.

III. Related work

To our knowledge, the literature lacks published surveys analyzing censorship attacks on circumvention approaches. The closest is Dingledine and Appelbaum's slide deck listing attacks on Tor [64].

Elahi et al. offer a report decomposing circumvention approaches into various phases and components to compare how they mitigate attacks on each [59]. For example, the forwarders, IDM, channel setup, and channel usage in our model of circumvention each has an analog in Elahi et al.'s model. More narrowly focused, a report by Khattak et al. covers *pluggable transports*, a subset of channel-based approaches designed to plug into a larger system, such as Tor [60]. Their work decomposes the channel of such transports into layers similar to a network stack to consider attacks and mitigations at each layer. Each of these two technical reports provides a survey of which tools offer which properties, similar to our Table IV. However, our survey differs in goal and scope: Rather than attempting to compare approaches, our study's primary goal is to compare evaluations in terms of how they relate to real-world concerns, grounding our discussion in empirical data regarding the behavior of actual censors.

Köpsell and Hillig proposed a taxonomy of circumvention approaches. They model censors as either blocking on *circumstances* or *content* [72], similar to our breakdown of channel setup and usage. However, they do not empirically study how their model compares to real attacks.

Prior works has also empirically evaluated approaches. Roberts et al. assessed deployed circumvention approaches by testing whether they work in various countries [56]. Callanan et al. used a combination of in-laboratory tests and user surveys to determine the usability, performance, and security characteristics of a variety of deployed approaches [54].

Robinson et al. [73] performed a study of 1,175 Chinese Internet users and found GoAgent [25] to be the most widely used tool. They concluded that collateral damage caused by an evasion tool should be the ultimate criterion for evasion, arguing that a censor will not block a technology viewed as economically or politically indispensable to the regime.

Leberknight et al. [74, 75] classify evasion tools and examine the relationship between a tool's classification and its

lifespan. Their study grouped 15 tools into general types: HTTP proxy, CGI proxy, rerouting, IP tunneling, and distributed hosting. Examining the lifespan of these tools and when new censorship technologies arose, they conclude that users gravitate toward the fastest unblocked approach Their work does not however develop corresponding measurements or delve into the technical issues of blocking and evasion.

By applying common methods to assessing circumvention technologies in their environment, these empirical works (i.e., [54, 56, 73–75]) assessed the success of deployed approaches. However, researchers and developers need criteria applicable to *undeployed* approaches. They must carefully select which proposals to develop and deploy due to the costs associated with such efforts. For this reason, we aim to understand how censors operate in order to predict which approaches will perform well if deployed.

Others have provided definitions of success or lists of criteria for circumvention approaches to fulfill. For example, Dingledine enumerates general properties that make for good evasion approaches [57]. While he presents anecdotal evidence, his work does not systematically explore the capabilities of censors to justify his criteria. Pfitzmann and Hansen provide definitions of security properties, such as *undetectability*, *unobservability*, and *unblockability*, but they do not ground these using empirical observations [76].

Houmansadr et al. [1] and Geddes et al. [77] empirically evaluate the vulnerability to blocking of a selection of mimicry-based approaches. Wang et al. [78] similarly study deployed circumvention tools. We compare these evaluation methods to real-world attacks in Section VII.

IV. Censorship As Practiced

In this section we examine censorship as practiced today more broadly than our previous sketch of Tor and China. We seek to illuminate the capabilities and limitations of current censors, with important implications for designing and evaluating effective approaches to circumvention.

We develop our knowledge of today's censors from two sources: measurement studies of censors, and reports from the field about actual blocking events.

We find little empirical analysis in the literature regarding how real censors block circumvention. Thus, we focus on collecting and organizing field reports, first for Tor, followed by those about other deployed approaches.

A. Measurement Studies

We examined 35 measurement studies [71, 79–112]. Table I provides an overview of what they show about such censorship. The majority of the papers assessed censorship of non-circumventing traffic. At the end of this subsection, we consider those that did look at the censorship of circumventing traffic.

Studies documented censors disrupting traffic by injecting fake DNS replies [83, 94], sending forged TCP Resets [81, 82, 85, 93], actively probing a suspicious protocol [71, 84] and using URL filtering systems, such as Blue Coat [97], Netsweeper [103], and SmartFilter [104]. While some have conjectured that the GFW uses machine learning based upon their interactions with it (e.g., [116]), we know of no rigorous studies suggesting such. A few papers also examined reverse-engineering the internal structure of censors [81, 85, 94].

Censor's capabilities	Seen
DNS injection	China 2007 [107], 2011 [91], China 2014 [94]; Pakistan 2010 [113], 2013 [83]; Iran 2013 [82]
HTTP injection	Pakistan 2013 [83]
TCP RST injection	China 2006 [85], China 2010 [92]
Packet dropping	Iran 2013 [82], China 2015 [79],
Stateless	China 2002 [80], 2006 [85]
Stateful	China 2007 [87], China 2012 [90], China 2013 [81]
Packet reassembly	China 2013 [81]
Using Netsweeper	Pakistan 2013 [103], Qatar 2013 [104], UAE 2013 [104], Yemen 2013 [104]
Using Blue Coat	Syria 2011 [98, 114]; Burma 2011 [104]; UAE 2013 [104], Qatar 2013 [104]
Using SmartFilter	Iran 2004 [115], Qatar 2013 [104], Saudi Arabia 2012 [104], UAE 2013 [104]

TABLE I
Censor Capabilities as Found in Prior Measurement Studies of Non-circumventing Traffic

In-path vs. On-path. Censorship system can operate in-path or on-path. An in-path monitor is a forwarding element between two networks through which all traffic must flow, such as Syria's and Qatar's employment of Blue Coat [104]. An on-path monitor passively examines passing traffic and can inject, but not remove, packets, such as China's GFW [93].

Each type of censorship system has advantages and limitations. On-path censors cannot conduct exploits requiring dropping packets (e.g., "packet dropping" [77]). On the other hand, in-path censors face exacerbated processing challenges due to the need to process all traffic at line rate lest the monitor introduces a performance bottleneck. Accordingly, attacks that rely on analyzing distributions of features (for example) could prove difficult to perform for an in-path monitor.

Stateless vs. Stateful. Stateless censors process packets individually, or at most perform limited packet reassembly. As such, circumventors can evade them by splitting sensitive strings into multiple packets. Stateful censors track transport-layer signalling and perform packet reassembly, with a corresponding processing and memory burden. The GFW operated in a stateless fashion before 2007 [80, 85], but stateful as of 2007, confirmed in 2013 [81, 87], indicating a system upgrade.

Whitelist vs. Blacklist. The majority of the censors use a blacklist to filter disallowed contents. We noted only two incidents of whitelisting based censors: Iran in 2013 [64, 82] and Tunisia in 2009 [64]. Aryan et al. studied censorship in Iran in 2013 [82] when non-whitelisted protocols were throttled. They found SSH file transfers got throttled to around 15% of its standard bandwidth and an obfuscated protocol, similar to the Tor's Obfsproxy protocol,[1] got throttled to near zero at about 60 seconds into the connection.

Blocking Timeline. Censors have particular motivations for censorship, which the onset of censorship can illuminate.

[1]The paper did not test any real circumvention protocol but used simple obfuscation approach, for example, XORing packet payloads with a predefined key, to obfuscate the SSH file transfer including the unencrypted portion of the handshake.

For example, China and Iran increase censorship activity to reduce political chaos, leading to blocking of circumvention systems before or during political events [117]. Aryan et al. studied Iranian censorship before the June 2013 presidential election and discovered that Iranian censors only allowed a small number of whitelisted protocols [82]. Such patterns of employment suggest that circumvention approaches that can distribute sets of new identifiers or new protocols only during critical times might succeed by denying the censors sufficient time to detect and block all of them.

Censorship of Circumvention. Winters and Lindskog studied the active probing by China that sends requests to suspected nodes to identify and block Tor when using its "Obfs2" circumvention extensions ("pluggable transports") [84], and Ensafi et al. studied the extension of active probing to Tor's Obfs3 transport [71]. Distinguishing the censor's probes from real users can prove challenging, as at least for China the censor draws upon a large number of IP addresses to originate the probes.

Ensafi et al. [105] studied reachability of Tor relays and directory authorities from China. Using "SYN backlog" scanning, they confirmed the GFW blocks access to Tor relays and directory authorities by dropping their SYN/ACK replies to clients.

From the leaked logs of Blue Coat deployed in Syria in 2011, Chaabane et al. found censorship of less that 2% of requests to the Tor network, but heavy censorship of other circumvention services such at Hotspot Shield [98].

We find it striking that just four papers have examined the censorship of circumvention approaches:

Research Gap 1. *Little research has examined how real censors exploit vulnerabilities in circumvention approaches, leading to a dearth of realistic censor models.*

B. Tor

To partly address Gap 1, we conducted and analyzed a survey of prior field reports on the blocking of deployed approaches. The story of Tor and the GFW in Section II forms a part of this larger analysis. Table II summarizes the censorship incidents we found and what we know about them.

The table is based upon field reports primarily coming from Tor advocates, developers, and users in the form of bug reports, blog posts, presentations, and comments. We undertook a comprehensive study of Tor's blog and bug tracker to find as many censorship-related reports as possible. We collected 747 blog posts and 13,337 bug-tracker reports from 12/2007 to 3/2015. We seeded a list of known censorship events with 9 blog posts and 11 bug reports [64]. Finding that grep-style searches yielded too many false positives, we used these manually labeled instances as a training set for a supervised machine-learning classifier, which found an additional 5 bug reports and 11 blog posts about specific censorship events.

We associate each event with its target (Tor, in most cases), as well as with the steps that make up the censor's attack. Some steps are vague because of a lack of documentation; for instance some are simply "Block" because we do not know exactly how the block was effected. Each step of an attack corresponds to some sort of detection, or an action taken based on a previous detection. Attacks can span multiple phases in the use of a circumvention approach. We observe a common pattern of the dynamic detection of traffic destined for a forwarder, followed by blacklisting of the forwarder's address to prevent any future communication.

In general, we see a progression from simple to more complex on the part of both censor and circumventor. Thailand blocked the www.torproject.org website in 2006 [64]. In 2007, Saudi Arabia [64] and Iran [121] began blocking HTTP requests with the string "/tor/" in the URL, intending to block communication with the Tor directory authorities.

DPI against the Tor protocol itself came only later, but took a variety of forms. Examples include checking for Tor's characteristic TLS renegotiation (Syria in 2011 [64]); checking for a specific Diffie-Hellman parameter during key negotiation (Iran in January 2011 [122]); measuring the TLS certificate lifetime (Iran in September 2011 [124]); and checking for specific TLS cipher suites (Ethiopia [136], Kazakhstan [131], the UAE [135], and the Philippines [132] in 2012).

Tor is occasionally caught up in more general censorship. In 2009 Tunisia blocked all but a few TCP ports [64]. Only relays running on those ports remained accessible. Throttling—deliberate slowing—of encrypted traffic took place in separate incidents in Iran in 2009, 2011–2012, and 2013 [64, 137]. Even more extreme, Egypt [138] and Libya [139] in 2011 and Syria [140] in 2012 completely disabled Internet access for a period of days or weeks. (We elide such total-censorship events from Table II.)

That instances of complete network blocking are rare and short-lived highlights an important general principle. Censors could prevent all circumvention by permanently disabling the network—but they do not, because Internet access brings general benefits that outweigh the harm of circumvention. The censor would prefer, ideally, to block forbidden sites and circumvention traffic, and nothing else. The purpose of circumvention is to make it difficult for the censor to distinguish these cases, thereby forcing the censor to allow some circumvention traffic (*underblocking*), or else block some non-circumvention traffic (*overblocking*), resulting in *collateral damage*. The higher the costs of overblocking, the more likely the censor will tend towards underblocking.

C. Approaches Other Than Tor

We also collected reports of censorship events against other deployed approaches. We find these harder to come by since most approaches do not offer as much public documentation as Tor does. Thus, here we offer not a comprehensive overview but illustrations of real censorship incidents that highlight additional facets regarding the operation of real censors.

Popularity-Driven Blocking. VPN Gate launched in March 2013 and quickly accrued over 5,000 unique clients from China [32]. Only three days after launch, and presumably as a result of this sudden popularity, the Great Firewall blocked the VPN Gate website and its central database of relay servers. Soon after, Chinese censors began crawling the database of

Event	Target	Steps
China 2008a [62]	Tor	Check requests for Tor ('.torproject.org') **then** Send TCP reset
China 2008b [62]	Tor	Check whether server IP is in blacklist **then** Timeout
China 2009a [63, 64]	Tor	Get Tor relays' IP addresses from public list **then** Blacklist server IP address **AND** Check whether server IP-port pair is in blacklist **then** Block
China 2009b [63, 64]	Tor	Get Tor bridges' IP addresses from webpage **then** Blacklist server IP address **AND** Check whether server IP is in blacklist **then** Block
China 2010 [64, 67]	Tor	Get Tor bridges' IP addresses from email **then** Blacklist server IP address **AND** Check whether server IP is in blacklist **then** Block
China 2011/10 [64, 68, 71, 84, 118, 119]	Tor	DPI for Tor's TLS 'Client Hello' for cipherlist **then** Graylist server IP-port pair **AND** Probe server for circumvention handshake looking for version cell **then** Blacklist server IP-port pair **AND** Check whether server IP-port pair is in blacklist **then** Block
China 2013/01 [71, 119]	Tor+obfs2	Observe suspected circumvention flow (method unknown) **then** Graylist server IP-port pair **AND** Probe server for circumvention handshake (looking for what?) **then** Blacklist server IP-port pair **AND** Check whether server IP-port pair is in blacklist **then** Block
China 2013/07 [71]	Tor+obfs3	Observe suspected circumvention flow (method unknown) **then** Graylist server IP-port pair **AND** Probe server for circumvention handshake (looking for what?) **then** Blacklist server IP-port pair **AND** Check whether server IP-port pair is in blacklist **then** Block
Ethiopia 2012/06 [120]	Tor	DPI for TLS 'Server Hello' for cipher 0x0039 sent by the Tor relay or bridge **then** Drop packet
Iran 2007 [64, 121]	Tor	Check GET requests for Tor ('/tor/') **then** Cut connection
Iran 2009 [64, 121]	SSL	Identify SSL (method unknown) **then** Throttle
Iran 2011/01 [64, 122]	Tor	DPI for Tor's DH parameter in SSL **then** Block **AND** Check whether server IP is in blacklist **then** TCP FIN
Iran 2011/10 [64, 123]	SSL?	Identify SSL for Tor (method unknown) **then** Throttle
Iran 2011/09 [64, 124]	Tor	DPI for Tor's SSL and TLS certificate lifetime **then** Block
Iran 2012/10 [125]	Tor	DPI for TLS 'Client Hello' for SNI that resolves to Tor relay/bridge **then** Block
Iran 2012/11 [125]	Tor etc?	DPI on TLS for client key exchange **then** Send a TCP reset and drop packet
Iran 2012/02a [126, 127]	SSL	Identify SSL handshake **then** Block
Iran 2012/02b [126]	Tor etc.	Check whether server IP-port pair is in blacklist **then** Block
Iran 2012/02c [126]	Tor etc.	Search for 'Tor' as a keyword, e.g., as a search term **then** Block
Iran 2013/03 [128]	Tor	DPI for Tor's SSL and TLS certificate lifetime **then** Block
Iran 2013/04a [129]	non-HTTP	Check for port 80 and whether protocol is non-HTTP (method unknown) **then** Send a TCP reset
Iran 2013/04b [129]	encryption	Check for encryption (method unknown) **then** Throttle
Iran 2014 [130]	Tor	Find IP addresses of Tor directory authorities **then** Blacklist server IP-port pair **AND** Check whether server IP-port pair is in blacklist **then** Block
Kazakhstan 2012a [131]	Tor	DPI for TLS 'Server Hello' for cipher 0x0039 sent by the Tor relay or bridge **then** Drop packet
Kazakhstan 2012b [131]	Tor	DPI for Tor's TLS 'Client Hello' for cipherlist **then** Block
Philippines 2012 [132, 133]	Tor	DPI for TLS 'Server Hello' for cipher 0x0039 sent by the Tor relay or bridge **then** Block
Saudi Arabia 2007 [64]	Tor	Check GET requests for Tor ('/tor/') **then** Cut connection
Syria 2011 [64]	Tor	DPI for Tor's TLS renegotiation **then** Blacklist server IP address **AND** Check whether server IP is in blacklist **then** Block
Syria 2012/12 [134]	Tor	DPI for Tor's TLS renegotiation **then** Blacklist server IP address **AND** Check whether server IP is in blacklist **then** Block
Thailand 2006 [64]	Tor	Check DNS requests for whether they are for Tor's website **then** Redirect to a block page
Tunisia 2009a [64]	non-web	Check whether port is not 80 or 443 **then** Block
Tunisia 2009b [64]	SSL+	Check whether port is not 80 (and client IP is on a graylist?) **then** Block
Turkey 2014 [96]	Tor etc.	Check DNS requests for whether they are for Tor's website **then** Block
UAE 2012 [135]	Tor	DPI for TLS 'Server Hello' for cipher 0x0039 sent by the Tor relay or bridge **then** Drop packet

TABLE II

STEPS IN REAL-WORLD CENSORSHIP ATTACKS AFFECTING TOR, INCLUDING DETECTING SUSPICIOUS TRAFFIC, BLACKLISTING IP ADDRESSES, AND DISRUPTION ACTIONS. The exploits of an attack are separated by "**And**," with "**then**" separating the detection and action steps of an exploit.

servers more than once a day to maintain an up-to-date blacklist.

In late 2013, Lantern [42] had a surge of Chinese users, who increased in number from 200 to 10,000 in just two weeks [141], followed soon after by blocking of the network and its website [142], with only a few users remaining able to connect.

Denial-of-Service Attacks. In 2015, GreatFire.org, a website offering information about and approaches for circumventing censorship in China, suffered from a DoS attack orchestrated by Chinese censors [79, 143, 144]. The major fallout for GreatFire was not downtime but rather a $30,000-a-day bill from their web hosting provider. The attack came shortly after the publication of a *Wall Street Journal* article regarding the website.

Man-in-the-Middle Attacks. In 2011, Iran launched a series of MITM attacks using fraudulent TLS certificates for many Internet services [145], including one for the Tor website. (This, however, did not affect the certificates used by the Tor network itself [146].) In 2013, China conducted an HTTPS MITM attack against GitHub that lasted a few days [147].

Malicious Software. Green Simurgh [51] is a circumvention

tool designed for users in Iran. In 2012, fraudulent copies of the software were found in the wild backdoored with a keylogger and other malicious features [148].

V. Channel-based Circumvention

We now survey approaches from both research papers and real deployments for "channel approaches" to circumvent censorship, which involve establishing a channel to a forwarder. In the next section we then analyze their evaluations.

Channel approaches vary in the form of censorship they aim to address, such as regarding the censor's motivations. For example, one approach may excel for use in the context of a highly repressive regime censoring and punishing political dissidents, while another may focus on causal users accessing social media in a moderately repressive regime.

Channel approaches also vary in their level of abstraction and implementation. Research papers sometimes propose channel protocols or schemes that could be instantiated in multiple ways (e.g., StegoTorus [19] and FTE [28] are parametrized). In some cases, researchers implement a protocol and benchmark it using a particular instantiation of an in-the-lab emulation of a censored network. Researchers rarely actually deploy their approaches for real users by setting up the necessary infrastructure. More often, activists deploy pragmatic homespun approaches due to the considerable effort required to actually deploy even a simple approach. Deployment typically involves a circumvention advocate setting up and maintaining the forwarder and associated infrastructure, providing documentation for users, and promoting the approach to attract users. In some cases, an advocate may skip setting up and maintaining the forwarder by directing users to a *found forwarder*, some pre-existing infrastructure maintained for some other purpose that can act as a forwarder (e.g., CacheBrowser [149]).

Table III provides an overview of circumvention approaches. We can divide the previously proposed circumvention systems into two main categories based upon what they primarily attempt to obfuscate: *setup* or *usage*. The setup category contains approaches that attempt to obfuscate the information about who will be communicating (e.g., IP address) and how (e.g., protocol identifiers). The usage category contains approaches that attempt to protect the usage of the approach during its employment. This entails obfuscating the user's behaviors to make them look non-circumventing. For tools that do both forms of obfuscation, we classify it based upon which form the tool designers focused on or presented as novel.

Additionally, we can split the approaches into those that focus on *polymorphism* and those that focus on *steganography* for obfuscation. (Most use a bit of both.) Both are methods of obfuscating a feature of the traffic that an approach produces, such as packet sizes or the value of parameters in a cryptographic handshake, that could reflect a vulnerability enabling identification of the approach producing the traffic.

Polymorphism is a way of spreading out behavior. To be polymorphic in a feature means that the feature takes on multiple values among different instances, such as messages. Spreading out the values of a feature used in a blacklist's signature can result in the signature no longer identifying disallowed traffic, increasing false negatives.

Steganography is a way of looking like allowed communications. To be steganographic in a feature means having values that are very close to allowed communications. The censor may fail to distinguish such steganographic traffic from genuine allowed traffic, resulting in false negatives.

The two concepts are not mutually exclusive. A polymorphic approach might steganographically match the characteristics of a generic class of traffic that censors allow due to their inability to identify it. Alternatively, matching an allowed protocol with random behavior requires polymorphism. However, the approaches we studied fall into two groups: those trying to not look like blacklisted traffic using polymorphism, and those trying to blend in with allowed traffic using steganography.

For example, ScrambleSuit [23], an approach polymorphic over usage, attempts to look random in hopes of having no easily recognizable behavior. SkypeMorph [18], an approach steganographic over usage, attempts to look like Skype traffic. Meek [26], steganographic over channel setup (not usage), also tries to look like allowed traffic, namely traffic headed to an allowed site hosted by a content delivery network (CDN). However, unlike SkypeMorph, Meek makes no effort to match the usage patterns of real CDN traffic, and instead just ensures that the connection setup looks similar by using the same IP address and URL as allowed traffic.

Looking at Table III, we see that research approaches (or at least their presentations) cluster in the area of steganography over usage, in which almost all approaches are only designs and are not deployed. It behooves us to consider the merits or drawbacks of this emphasis. We approach this question by analyzing how censors block approaches.

Approaches also vary in their identifier distribution mechanisms (IDMs). Such mechanisms include receiving IP addresses from friends by hand or via email. Some approaches include keys that authorize users to forwarders. Unlike a channel, an IDM does not need to be able to communicate arbitrary information to arbitrary destinations, nor are latency and bandwidth typically as salient concerns. Due to these differences, the problem of identifier distribution is largely orthogonal to channel setup and usage, with its own disjoint set of papers (e.g., [150–152]). To maintain focus in our work, moving forward we set IDMs aside except to comment on real censors' attacks on them when it sheds light on their abilities to also attack channels.

VI. Evaluation Criteria

To understand how advocates and researchers evaluate current circumvention approaches—both those presented in papers as well as used in practice—we identified 56 documents about circumvention approaches to study. We selected 34 academic papers by searching the top computer security conferences, Google Scholar, and Microsoft Academic search using keywords like "censorship circumvention" and "censorship resistance," taking those papers that present a channel-based circumvention approach. We selected 25 documents, such as webpages and posted specifications, about approaches

	Setup	Usage
Polymorphism	**Tor Sep, 2011**[a], **BridgeDB** [2], **CGIProxy** [38], **Flash Proxy** [17], **FreeGate** [40], **Green Simurgh** [51], **GTunnel** [44], **Hotspot Shield** [45], **JAP** [46], **Lantern** [42], **Psiphon** [41], **Ultrasurf** [39], **uProxy** [37], **VPN Gate** [32], **Your Freedom** [47]	Dust [24], **GoHop** [53], **MessageStreamEncryption** [8], **Obfs2** [20], **Obfs3** [21], **Obfs4** [22], **ScrambleSuit** [23]
Steganography	**Tor Jan, 2011**[b], **Tor Jun, 2012**[c], CacheBrowser [50], Cirripede [13], CloudTransport [36], Decoy routing [14], **GoAgent** [25], **Meek** [26], OSS [27], Rebound [52], TapDance [35], Telex [15]	Bit-smuggler [43], Castle [48], CensorSpoofer [16], Collage [12], DEFIANCE [3], Facade [33], Facet [34], **FOE** [9], FreeWave [30], **FTE** [28], Identity-based Steganographic Tagging [5], Infranet [4], **MailMyWeb** [10], Marionette [29], Message in a Bottle [6], Rook [49], SkyF2F [11], SkypeMorph [18], StegoTorus [19], SWEET [31], Trist [7]

TABLE III

PRIOR RESEARCH ON EVADING NETWORK-BASED CENSORSHIP USING OBFUSCATION, ORGANIZED BY PRIMARY OBFUSCATION METHOD. Columns show the primary type of obfuscated feature. **Bold** denotes deployed approaches. [a]Tor 0.2.3.4-alpha: changed TLS cert expiration time to make it less distinct. [b]Tor 0.2.2.22-alpha: changed TLS D-H parameter to one used by Apache's mod_ssl. [c]Tor 0.2.3.17-beta: updated TLS cipher list to match Firefox version 8 or later.

deployed in the wild by taking those that appear functional or to have had users. Of these deployed tools, 7 were also selected as academic papers counted above. Lastly, we added 4 reports on the *evaluation* of circumvention tools, looking for those that mention technical criteria. Amongst the papers we included the documentation of Bit-smuggler [43], which is a tool though not deployed. While we did not cover every existing approach, we believe we have covered the ones that effectively shape the circumvention arms race in research and practice.

Table IV shows the criteria related to censorship attacks, such as how easily a censor can detect it, disrupt it, or harm its users. To create this table, we started by reading the selected documents, paying particular attention to sections with titles such as "evaluation", "experiments", "threat model", and "design goals". We then made a large superset of all criteria discussed in any evaluation. We combined similar criteria where reasonable, losing some nuances and making adjustments to terminology when needed.

Next, we organized the criteria into two classes: (1) abstract *goals* and (2) concrete *metrics*. A goal motivates metrics that measure an aspect of how well an approach meets the goal. For example, some approaches have the goal of resistance to traffic analysis; their developers measured the satisfaction of this goal using various metrics, such as the packet size distribution produced by their approach, which show how similar it looks to allowed and disallowed traffic. Under metrics, we include not just traditional quantitative measurements, such as throughput, but also binary properties about the approach, such as whether it employs authentication. While ideally metrics objectively measure an approach, we allow a bit of vagueness in their definitions since obvious ways of making concepts like "popular hosts" precise exist despite some documents not discussing them. We used the motivations for metrics provided in the documents to categorize them under goals, per Table IV; where not all documents agreed, we used our judgement. Note that some metrics fall under two goals each.

To understand the evaluations that document authors had in mind, as opposed to how well an approach did, for each goal and metric we determined whether each document mentioned it, giving it a box □ in the table if so. We included discussed criteria regardless of whether a documented approach actually met it, or even tried to.

For documents about a single approach, we also assessed which metrics the authors "checked off" (denoted by ✓). For binary metrics, we checked off those that the document stated that the approach provided. For quantitative metrics, we checked off those for which the document provided a measured value, since these have no clear cutoffs for satisfaction. We did not check off goals, since there is no clear meaning of satisfying most of them due to their generality.

The reason we checked off metrics was not because we wanted to evaluate the approaches; rather, we did so to understand which metrics the authors took seriously enough to either meet or at least measure. As such, we only record the criteria as documented: we made no effort to infer undocumented relationships between goals and metrics, to discover undocumented features of approaches, to evaluate the correctness of the evaluations performed, or to rank approaches.

To improve our assessments, we did two rounds of emailing the authors of each document that included contact information. Each round led to corrections to our assessments of tools and to adjustments to our list of criteria. We re-examined the documents to check that we listed the correct criteria as our criteria list changed. Some authors sent us additional documentation to consider, which we accepted as long as it was publicly available and created by the same team as the primary document. Even with the two rounds, as of this writing some (non)assignments of criteria to tools have not been validated by the tools' authors, particularly those involving criteria newly introduced in response to the second round of replies. This process made clear to us the subjective judgement involved in deciding whether a "document" (such as an amorphous website) "discusses" a criteria in an "evaluation" given the vagueness of language (both ours and theirs).

In the end, we have 23 goals and 75 metrics. Of these, 15 goals and 47 metrics relate to how easily a censor can attack an approach. Because we focus on criteria related to attacks, we relegate the others to the appendix.

Since we consider most of the criteria in Table IV to be

	Deployed Tools																		Deployed & Papers						Academic Papers Proposing Approaches																												Evaluation Papers			
	[2]	[8]	[9]	[10]	[20]	[21]	[22]	[25]	[37]	[38]	[39]	[40]	[41]	[42]	[44]	[45]	[47]	[51]	[23]	[26]	[28]	[32]	[46]	[53]	[3]	[4]	[5]	[6]	[7]	[11]	[12]	[13]	[14]	[15]	[16]	[17]	[18]	[19]	[24]	[27]	[29]	[30]	[31]	[33]	[34]	[35]	[36]	[43]	[48]	[49]	[50]	[52]	[54]	[55]	[56]	[57]
Resistance to traffic analysis		□			□	□	□		□						□				□	□	□			□	□		□						□	□			□	□	□	□	□	□	□		□	□	□	□	□	□	□	□				
Serial connection count																																								✓	✓															
Matching allowed n-gram distribution																																													✓					✓						
Network stack fingerprinting																																														□						✓				
TLS characteristics																																		✓												✓										
Total payload length																																									✓															
Total TCP connection																																									✓															
Concurrent connection count																																									✓															
Inter-packet timing					□	□	✓												✓					□													✓		✓			✓				□			✓	✓						
Use encryption to resist traffic analysis		✓			✓	✓	✓								✓				✓					✓			✓										✓	✓	✓							✓										
Number of HTTP requests/responses																																									✓															
Packet size distribution		□			□	□	✓						✓						✓	✓				✓													✓	✓	✓						✓	□			✓	✓						
Connection length																				✓																		✓											✓	✓						
Lack of server response																																														✓										
Protocol misclassification rate																					✓																				✓															
Use random port					✓	✓	✓																	✓																																
Resistance to traffic manipulation													□												□	□								□			□							□		□			□	□	□	□				
Use UDP with reliability																																					✓												✓	✓						
Use authentication		✓					✓						✓						✓						□	✓																		✓					✓							
Use TLS for integrity																																		✓																		✓				
Use error correcting codes																										✓																		✓												
Limit service to each user ID (DoS)																														✓												✓	✓		□	✓										
Resistance to active probing		□					□						□	□					□			□			□	□							□								□			□	□	□		□	□	□		□				
Respond to probes like something else																						✓																			✓				(✓)	✓		✓		✓		✓				
Use authentication		✓					✓						✓						✓						□	✓																		✓					✓							
Resistance to security attacks		□			□	□	□		□				□	□		□	□		□				□	□	□					□	□	□	□	□			□		□	□		□	□	□	□	□	□				□	□			□	
Use encryption for confidentiality					□	□	✓						✓				✓		✓						□						✓	✓					✓			✓							✓									
Use client puzzle (DoS)																																		✓									✓		□						□					
Use authenticated key exchange (MITM)							✓																																																	
Limit service to each user ID (DoS)																														✓												✓	✓		□	✓										
Use certificate pinning (MITM)														✓																																(✓)										
Use shared secret (MITM)		✓											✓																																											
Ignore invalid connections (DoS)																																													✓	✓						✓				
Use strong third-party service (DoS)													✓	✓																																										
Use TLS for confidentiality																																														✓						✓				
Use timestamp (replay)							(✓)												✓															✓					✓					✓		✓										
Use trustworthy proxy									✓				✓																	✓	✓											✓														
Use block cipher (key reuse)																																		✓																						
Resistance to blocking									□	□	□		□	□						□	□	□	□		□	□		□		□	□	□	□	□	□	□	□	□		□	□	□	□	□	□	□	□		□	□	□	□				
Use many access points									✓	(✓)												✓	✓		□	(✓)		✓		□	✓					✓				✓		(✓)														
Use popular hosts													✓	✓						✓										✓														□			✓				✓					
Use a popular protocol																														✓	✓				✓		✓					✓	✓	✓	✓					✓						
Use network infrastructure																																✓	✓	✓												✓						✓				
Server obfuscation																																			□							□														
Indirect connection to forwarder																																			✓							✓														
Availability of infrastructure									□								□			□		□			□						□	□			□	□				□		□	□			□				□			□		□	
Rate of proxy churn																						✓														✓																				
Preponderance of suitable servers																				✓											✓				✓					✓						✓										
Stability of decoy hosts																																			✓																					
Geographic diversity of proxies																						✓																																		
Number of proxies									□								✓					✓			□																															
Independent deployment																																										✓														
End-user protection					□	□	□		□	□	□	□	□	□	□	□	□	□		□	□	□	□							□		□		□		□	□	□		□		□					□						□	□		□
Does not store user information									(✓)		□		✓	(✓)	□	(✓)	□					□	□																														□	□		(□)
Hide user information from end host					□	□	□		□	(✓)	✓	✓	□	□	(✓)	✓	✓	✓		✓	✓		✓									✓		□		✓	✓	✓		✓		✓					✓						□	□		□
Deniability under computer inspection																															□						□												(□)							
No installation				✓						✓	✓	✓	✓	✓																✓						✓									✓								□	□		□
Adaptability to blocking									□												□	□																			□		□	□	□				□	□				□		
Time to create an adaptation																																																	✓							
Resistance to insider attacks																																			□																					
Decentralization of trust									□																																											□	□	□	□	□
Self promotion																																																								□
Veracity of claims																																																						□		□
Diversity of users																																																						□		□

TABLE IV

Goals and Metrics Related to the Ease of Attack per Approach/Paper Evaluation. We list metrics under the goals for which papers have discussed them. □ denotes that the document mentions a criteria. ✓ denotes that the document either measured the metric, or satisfied it in the case of binary criteria. Ambiguous cases are denoted by using a lighter color (✓). Note that we do not list criteria for an approach/paper if the examined evaluation documentation did not mention it even if we know that the approach/tool has it. As such, this table should only be used to compare evaluations, not tools. *Preponderance of suitable servers* refers to how many servers exist in the wild with the needed functionality to support the approach. *Lack of server response* refers to an issue where a user of an approach might send requests but not receive responses do the approach failing; such unmet requests may appear suspicious over time. *Self promotion* refers to approaches promoting itself as a circumvention approach, which could lead to backlash from the censor.

self-explanatory, for reasons of limited space we forgo an enumeration of them here and only touch upon the interesting points. However, the table caption explains a few opaque ones and the interested reader can find the other definitions in the appendix. Below, we provide some observations about the evaluations we have summarized.

A. General Observations

The provided tables make a few general observations clear. First, publications tend to have more evaluation goals and metrics than deployed tools.

Second, evaluations share many of the same goals, but also differ greatly from work to work in terms of metrics used or even goals mentioned. In some cases, differing expectations about the targeted users' needs and differing ideas about the importance of criteria could justify these differences in goals and metrics. However, evaluations would ideally make such tradeoffs explicit by mentioning unmet goals, which would make comparing approaches easier. Furthermore, we suspect that often developers only mention criteria on which they expect their approach to perform well, given the small number of unchecked boxes for metrics in Table IV.

Recommendation 1. *The needs and censorship context of the intended users—not the capabilities of an approach—should govern the evaluation of the approach.*

Third, there is less agreement on what metrics to use or even which metrics map to which goals, making comparison even more difficult.

Fourth, no metric in use comprehensively evaluates undeployed approaches. The number of users, a reasonable holistic metric of success in use, cannot evaluate approaches for deployment since the approach must already be deployed to calculate it.

Research Gap 2. *Prior evaluations lack holistic metrics for undeployed approaches.*

B. Criteria Related to Attacks

Evaluations of approaches focus particularly (nearly 2/3s) on criteria related to attacks. Our survey of academic papers found that they are typically motivated by overcoming some real attacks on circumvention approaches. However, looking at their evaluations, they tend to focus on more complex hypothetical attacks, per our discussion in Section VII.

Evaluation by Techniques Used. Even without such empirical evidence, a noteworthy pattern emerges in Table IV: the large number of metrics starting with the word "Use". These correspond to binary metrics measuring whether an approach employed a technique, such as authentication or encryption, to avoid a type of exploit, rather than on considerations regarding the censor's capabilities. For example, one such metric is *Use popular hosts*. This metric suggests a different metric about a property of the censor: *Blocking requires disrupting a popular host*. This property does not presuppose any mechanism for achieving it, making cross-approach comparison easier. Furthermore, it considers all the vulnerabilities of the approach rather than focusing on a single technique. For example, a system using many hosts might still be blockable without disrupting a popular host if the circumventing traffic can be identified and dropped. Such censor-oriented versions of metrics push developers to fully consider the space of concerns.

Research Gap 3. *Approaches should be evaluated based on the properties they provide rather than the techniques they use.*

C. Other Criteria

While we focus in this paper on evaluation related to censorship attacks, some aspects of other criteria (detailed in Table VII in the appendix) merit discussion.

Ability to Deploy. In some approaches, advocates consume resources to maintain a forwarder (e.g., Psiphon [41]). In others, they pay for others to maintain it (e.g., Meek [26] and CloudTransport [36]). In others, they rely upon volunteers (e.g., Infranet [4] and Flash Proxy [17]). In rare cases, the forwarder might be found and free, but the advocate must still maintain an approach for making use of it (e.g., CacheBrowser [50]).

In all these cases, the advocate faces costs and inconveniences. While some commercial services explain their fees providing some light on their costs, few of the documents we examined discuss cost to advocate to maintain and run the system.

Research Gap 4. *Only 6 of 34 papers mention the costs for advocates maintaining the system.*

This observation reflects the nearly absent goals of *Client performance*, *Infrastructure cost*, and *Sustainable network and development*. The exceptions include the CloudTransport, which uses cloud services as a rendezvous point to transfer data between a client and a server [36], which reports the cost of browsing a webpage in USD; and Meek, which uses CDNs and App Engine, and reports the total cost of running the system from early 2014 until April 2015 [26].

The related criterion *Preponderance of suitable servers* measures the number of potential forwarders, and appeared in several works [12, 16, 26, 27, 35]. Some approaches may require particular functionality from servers that act as forwarders, such as specific implementation quirks. While this criterion may appear unsuitable for undeployed approaches, such as the number of users or forwarders, it differs by representing an *upper bound* on the number of forwarders possible that can be determined without deploying the approach.

Usability. For usability, previous research and deployed approaches have examined many metrics, such as cost to user, ease of setup, and usage flexibility. Cost-to-user looks at whether users must pay to use a deployment of an approach. Some tools like Psiphon [41], Ultrasurf [39], and Facet [34] do not require any installation, which relieves censored users from acquiring software, or require only a small download (e.g., 4.3MB for lantern-installer-beta.dmg) [42].

Some approaches have restrictions in terms of functionality, network, and system architecture. Facet only provides access

to videos [34]. CacheBrowser can only serve websites hosted on a CDN [50]. Flash Proxy is incompatible with NAT [17]. Ultrasurf only works on Windows [39]. Some tools are not accessible from certain countries. For example, in 2010 FreeGate and Ultrasurf outright blocked connections from all but the few countries that they cared to serve (China and, for Ultrasurf, Iran) [57].

Deployed tools are more likely to evaluate usability than are undeployed research proposals. General circumvention evaluation papers, not tied to any single approach, were the most comprehensive in their consideration of usability.

Research Gap 5. *Only 9 of 34 research papers mentioned the goal of usability, and none assessed it with metrics involving actual users.*

VII. Comparing Resistance Criteria to Real Censors

We now take a closer look at criteria assessing an approach's resistance to a censor attempting to block it. We wish to classify the vulnerabilities and exploits found in both real attacks and papers to better understand how they relate and what censors are doing in practice.

Unfortunately, we have limited information about the exploits used by real censors, forcing us to make tentative classifications when required details are missing.

Similarly, research papers often point to vulnerabilities (or partially specified exploits) rather than to concrete exploits. When a classification depends upon the implementation details of an exploit, we infer a reasonable exploit from the vulnerability (and any partial specification provided) to classify the vulnerability. Such classifications must be understood as predicated upon the exploit we inferred; future work might find more clever exploits for seemly expensive vulnerabilities.

Recommendation 2. *Papers identifying weaknesses in other approaches should provide not just vulnerabilities but specific exploits to illustrate the practicality of exploiting the vulnerability.*

In some cases more than one obvious exploit might exist, in particular, one for a censor preferring overblocking and one for a censor preferring underblocking. We use the one with a preference toward underblocking due to the sensitivities of the censors who primarily concern us.

Table V shows our classification of real-world censorship vulnerabilities (or, rather the most natural exploit for a vulnerability) under three criteria: the phase of the circumvention approach in which the vulnerability appears; whether the inferred exploit is passive, reactive, or proactive; and how robust the feature is to lost packets.

Table VI shows the same classification for tests of vulnerabilities found in the academic papers of Table IV, along with three additional papers. The additional papers focus on traffic analysis and not full approaches, but introduce a large number of additional criteria related to traffic analysis [1, 77, 78]. While Table VI shows vulnerabilities at more detail than the metrics in Table IV, some rows correspond to a combination of similar vulnerabilities.[2]

Before we explain our classifications in detail and the implications of how the two tables differ, we note a general difference. Many of the tests found in papers (Table VI) test whether the circumvention approach looks like some *cover protocol*, a generally allowed non-circumventing protocol that the approach attempts to steganographically match. Most, but not all, real-world exploits look for signs of circumvention rather than checking whether the traffic matches some allowed protocol.

A. Phase Exploited

First, we examine the phase exploited. These phases include acquiring needed identifiers (IDM), channel setup, and channel use. We also include a *subsidiary* phase, which refers to how the approach behaves when not engaged in any of its main phases, such as when contacted by a non-user with a malformed packet. While such subsidiary behavior does not play a role in facilitating the channels, a censor can fingerprint an approach by studying the subsidiary behavior it exhibits in response to active probes.

Table V shows real-world censors exploiting features of either the channel setup or the IDM. In some cases, when exploiting the setup the censor looks for features inherent to the setup, such as initialization parameters of a protocol handshake. In other cases, the censor looks at features present in all packets of a channel, such as the destination IP address, but exploits the setup simply because it occurs first. Either way, the censor need not examine a connection for long. When exploiting the IDM, the censor either harvests identifiers using methods similar to how a real user would acquire them, or attempts to make reaching the IDM impossible.

We conjecture that censors focus on channel setups and IDMs not only to reduce how long it must watch a connection, but also since such setup and IDM traffic shows little variation across all the users of an approach, unlike channel usage, which varies depending upon how each user uses the channel. While the latter variation of usage could be handled using statistics over long traces of usage traffic, such approaches require that a censor retain more state, and impose tuning issues for balancing false positives and negatives.

Recommendation 3. *Circumventors should concern themselves more with vulnerabilities of the channel setup than of the channel usage.*

Table III shows that that deployed tools obey this recommendation, numerous academic approaches do not.

Table VI shows that while papers also include numerous attacks on the channel setup, they often deal with features that require analyzing channel usage. We also find many exploits of subsidiary behaviors, which real attacks ignore. This discrepancy appears to be a case of research running ahead of practice. These attacks are from Houmansadr et al., who showed that mimicry-based approaches for looking like

[2]Furthermore, we did not add reactive exploits for which a closely related proactive attack exists, since the proactive exploits can be converted into reactive ones by waiting for the probe to arise from user traffic.

Description and where seen	Phase	Nature	Meas. loss	Network loss
Detect attempts to get instructions for using Tor				
Check DNS requests for whether they are for Tor's website [64, 96]	IDM	passive	UB	n
Check GET requests for Tor ('/tor/') [64, 121]	IDM	passive	UB	n
Check requests for Tor ('.torproject.org') [62]	IDM	passive	UB	n
Search for 'Tor' as a keyword, e.g., as a search term [126]	IDM	passive	UB	n
Detect identifiers needed to setup a Tor channel				
Find IP addresses of Tor directory authorities [130]	IDM	proactive	?	?
Get Tor relays' IP addresses from public list [63, 64]	IDM	proactive	ub	ub
Get Tor bridges' IP addresses from webpage [63, 64]	IDM	proactive	ub	ub
Get Tor bridges' IP addresses from email [64, 67]	IDM	proactive	ub	ub
Detect the Tor protocol				
Identify SSL for Tor (method unknown) [64, 123]	setup	passive	?	?
DPI for Tor's DH parameter in SSL [64, 122]	setup	passive	UB	n
DPI for Tor's SSL and TLS certificate lifetime [64, 124, 128]	setup	passive	UB	n
DPI for Tor's TLS renegotiation [64, 134]	setup	passive	UB	n
DPI for TLS 'Server Hello' for cipher 0x0039 sent by the Tor relay or bridge [120, 131–133, 135]	setup	passive	UB	n
DPI for Tor's TLS 'Client Hello' for cipherlist [64, 68, 71, 84, 118, 119, 131]	setup	passive	UB	n
DPI for TLS 'Client Hello' for SNI that resolves to Tor relay/bridge [125]	setup	passive	UB	n
DPI on TLS for client key exchange [125]	setup	passive	UB	n
Probe server for circumvention handshake (looking for what?) [71, 119]	setup	proactive	?	?
Probe server for circumvention handshake looking for version cell [64, 68, 71, 84, 118, 119]	setup	proactive	ub	ub
Observe suspected circumvention flow (method unknown) [71, 119]	chan.?	passive	?	?
Detect the destination of packets				
Check whether server IP is in blacklist [62–64, 67, 122, 134]	setup	passive	UB	n
Check whether server IP-port pair is in blacklist [63, 64, 68, 71, 84, 118, 119, 126, 130]	setup	passive	UB	n
Detect encryption				
Identify SSL (method unknown) [64, 121]	setup	passive	?	?
Identify SSL handshake [126, 127]	setup	passive	?	?
Check for encryption (method unknown) [129]	setup?	passive	?	?
Detect anything beyond basic web usage				
Check whether port is not 80 or 443 [64]	setup	passive	UB	n
Check whether port is not 80 (and client IP is on a graylist?) [64]	setup	passive	UB	n
Check for port 80 and whether protocol is non-HTTP (method unknown) [129]	setup?	passive	?	?

TABLE V

CLASSIFICATION OF EXPLOITS INFERRED FROM REAL-WORLD ATTACKS. "setup" means that vulnerability is exposed during channel setup (and possibly usage as well) whereas "use" means it is exposed only during channel use; "IDM" means that it is exposed during identifier distribution. "OB" means overblocking; "UB", underblocking; "ub", underblocking that a censor can easily recover from by retransmitting its own probe; and "n", neither under- nor overblocking.

allowed traffic is fingerprintable by such subsidiary behavior (as well as other means) [1]. While such approaches appear in the research literature, they have had little impact in practice, making the attacks unneeded by current censors.

We lack empirical evidence as to whether such mimicry approaches would enjoy a period of success long enough to warrant their deployment.

Research Gap 6. *We do not understand the speed with which censors block new approaches. Thus, we lack the ability to gauge the value of deploying low-overhead approaches with known weaknesses.*

We do have some information on this question. On March 8, 2013, Iran blocked most VPNs, forcing users to switch to more sophisticated circumvention tools [129]. The Iranian government adapted sufficiently to these new tools for users to complain of them no longer working within two months (by May 5 [129]).

Another illustration involves two Tor incidents: Iran 2011/09 and Iran 2013/03. In the first incident, Iran learned to fingerprint an abnormal TLS certificate lifetime used by Tor. It took Iran about 1.5 years to fingerprint the less odd but still static and easily identifiable lifetime Tor used next.

Finally, while the research papers we examined lacked IDM exploits, we note that this reflects an artifact of aforementioned split of research into papers looking IDM and those looking at channels proper, for which we only selected papers in the second category for this detailed analysis.

B. Nature of Exploit Detection Activity

Houmansadr et al. distinguish between passive, reactive, and proactive exploits [1].[3] An exploit can *passively* monitor traffic passing through the censor's border (say), or interact with clients and servers *reactively* to modify traffic or *proactively* by producing traffic. For example, suppose that a circumvention approach attempting to look like a normal web server reacts differently to a request for a non-existent webpage [1]. The censor could passively wait until a real user makes such a request and the deviation naturally arises to detect it. Alternately, the censor could reactively modify a request to point

[3]They call exploits "attacks" and reactive exploits "active attacks".

Description and where seen	Phase	Nature	Network loss
Detect a feature of a packet that differs from the cover protocol			
Different packet sizes for packets with fixed length from Skype [1]	setup	passive	UB
Absence of start-of-message fields of Skype UDP packets [1]	setup	passive	n
Different ciphersuite for TLS handshake than Chrome on Linux [35]	setup	passive	n
Detect a feature of content that differs from the cover protocol			
Different HTTP response length than Firefox downloading Amazon.com [29]	use	passive	OB&UB
Exploiting discrepancies in file format semantics [1, 78]	use	passive	UB
The value of the content length field matches the actual length of the content [28, 78]	use	passive	OB
Detect packets produced by a probe that differ from the cover protocol's			
Manipulating the `tag` field in SIP OK to close a connection that normally would be kept open [1]	setup	reactive	UB
Verify standard supernode behavior by flushing supernode cache [1]	subsidiary	proactive	OB
Check for the correct response to HTTP GET request for an existing page [1, 78]	subsidiary	proactive	OB
Wrong response to HTTP GET request for non-existing page or wrong protocol [1, 78]	subsidiary	proactive	n
Detect the presence of packets that the cover protocol would not produce			
Detect the presence of packets from a TCP close or delay that Skype would not produce [1]	setup	reactive	n
Detect the absence of packets that the cover protocol would produce			
Absence of standard Skype control traffic [1]	setup	passive	OB
Absence of standard Skype user traffic [1]	use	passive	OB
Absence of normal server replies to client [35]	setup	proactive	OB
Absence of expected Skype setup packets in response to network inferference [1]	setup	reactive	OB
Absence of expected SIP setup packets in response to malformed requests [1]	setup	reactive	OB
Absence of call termination after dropping SIP RTP packets [1]	use	reactive	OB
Absence of response to odd HTTP requests [1, 78]	subsidiary	proactive	OB
Detect making connections in a way that the cover protocol does not			
Connecting to a tainted IP during setup even if the channel does not [77]	setup	passive	UB
Many long-lived connections to one bridge node vs. few short-lived [77]	use	passive	n / UB
Check for abnormal number of concurrent connections while downloading [29]	use	passive	n / OB&UB
Has an abnormally large number of outgoing connections per session [27]	use	passive	UB
Many HTTP/Skype connections to a single server [1]	setup	passive	UB
Different number of TCP connections per session than Firefox downloading Amazon.com [29]	use	passive	n / OB&UB
Having a non-standard connection duration [26, 48, 49]	use	passive	n / OB&UB
Detect abnormal feature of packet			
Non-random-looking TLS handshake client nonce [15]	setup	passive	n
Payload length of 149 bytes for first packet [78]	setup	passive	n
The first packet looks random [78]	setup	passive	n
URI in the first GET request has length 239 bytes [78]	setup	passive	n
High entropy for the URI in the first GET request [78]	setup	passive	n
Detect abnormal traffic feature (e.g., timing or size) distributions			
Check for dependencies between supposedly separate connections [1]	setup	passive/reactive	OB&UB
Non-random packet length distribution [24]	use	passive	OB&UB
Different number of HTTP request-response pairs per connection when downloading Amazon.com [29]	use	passive	OB&UB
Different distribution of packet lengths from normal traffic [18, 19, 22, 23, 48, 49]	use	passive	OB&UB
Different distribution of flow sizes from normal TCP [19]	use	passive	OB&UB
Different distribution of connection times from normal TCP [19]	use	passive	n / OB&UB
Different distribution of interpacket arrival times or rate from normal traffic [1, 18, 22, 23, 30, 48, 49]	use	passive	OB&UB
Percentage of ACK messages that come a certain time after the ACK message that preceded it [78]	use	passive	OB&UB
Different average packet size than Skype [30]	use	passive	OB&UB
Different average difference in packet length over time from Skype voice [77]	use	passive	OB&UB
Different standard deviation of distribution of packet lengths from Skype voice [77]	use	passive	OB&UB
Fits the pattern of pre-recorded traffic [1]	use	passive	OB&UB
Different n-grams distribution over packet lengths than normal traffic [34, 49]	use	passive	OB&UB
Detect abnormal traffic statistic of feature distributions			
The entropy of packets [78]	use	passive	OB&UB
Percentage of TCP ACK packets sent in each direction [78]	use	passive	OB&UB
Five most common payload lengths of packets [78]	use	passive	OB&UB

TABLE VI

CLASSIFICATION OF EXPLOITS INFERRED FROM VULNERABILITIES FOUND IN PAPERS. The notation is the same as in Table V. "Subsidiary" means that it is an active exploit on how the approach behaves when not acting as a channel, such as in response to port scanning. The results of measurement loss may be inferred from those of network loss: cases where neither would be blocked ("n") become underblocking ("UB"). When there is not enough information to make a definitive classification, we separate the possibilities with "/". We put exploits discussed in only the traffic analysis papers [1, 77, 78] in grey.

to a non-existent page, which converts what might be a low probability vulnerability into a high one. Another alternative is to proactively probe the circumvention approach by sending it such a request at the censor's convenience.

Proactive exploits can operate indiscriminately by scanning the Internet looking for circumvention servers. However, given the Internet's size, we observe that such exploits tend to be triggered by some other event. For example, a passive exploit may identify suspicious traffic coming from a server, which might then trigger a proactive probe to confirm with higher confidence that the server is running a circumvention tool. The confirmation may then trigger the blacklisting of the server's IP address (as seen in China [71]). This overall attack starts with a cheap, low-confidence passive exploit, moves onto a more expensive high-confidence one, and ends with a high-confidence simple exploit that finally blocks traffic.

Table V shows real-world censors using passive and proactive exploits, but not reactive ones.

We conjuncture that censors avoid reactive attacks since such exploits must operate not just at line speed, like passive ones, but also manipulate traffic at that speed. Thus, censors may prefer to use proactive exploits of a vulnerability even when a reactive exploit for it also exists.

Recommendation 4. *Today's landscape indicates that circumventors should concern themselves more with low-cost passive and proactive exploits than reactive ones.*

Table VI shows 5 reactive exploits (all from [1]) out of 46 exploits found in the papers examined, even after we discarded reactive versions of proactive exploits.

C. Packet Loss

Lastly, we look at the robustness of vulnerabilities to lost packets. A censor's monitor can fail to observe a particular packet for a flow (whether benign or circumventing) due to several reasons: the measurement apparatus cannot keep up with the traffic stream and fails to capture the packet; a routing change, or multipath routing, causes the packets to take a route that does not transit the monitored link; or the packet is genuinely lost by the network, such as due to congestion.

In the first two instances, the loss is only from the perspective of the monitor, not from the client or server. For either, suppose that allowed traffic normally includes a packet absent from the circumventing traffic, a type of vulnerability flagged many times in Table VI. In this case, the apparent loss will make allowed traffic look disallowed, and simple exploits of the vulnerability will overblock. Exploits could attempt to avoid such overblocking by not acting immediately and checking for the presence of exculpatory packets over period of time long enough to produce multiples instances of it, or, in the case of proactive exploits, probing again. However, such complexity adds storage costs, slows blocking, and could still overblock in the face of multiple lost packets.

If instead the circumventing traffic has a packet not found in allowed traffic, such missing packets will cause simple exploits to underblock, which, per the empirical evidence presented previously, censors tend to prefer to overblocking. More complex vulnerabilities involving the distribution of some feature over time may cause both over- and under-blocking over time. On the other hand, genuine packet loss in some cases will not particularly affect the exploit. Such packets will register as dropped to the end-points, often causing retransmission, providing the exploit another detection opportunity.

Table V shows that real censors tend to use vulnerabilities that produce underblocking but not overblocking for lost packets. Table VI, on the other hand, shows papers focusing on vulnerabilities that may produce overblocking for lost packets.

Recommendation 5. *Censors use exploits for which packet loss results in underblocking instead of overblocking. Circumventors should protect against such exploits.*

VIII. Research Agenda

Our empirical investigations of censors and circumventors above has identified 6 research gaps and 5 recommendations. We now consider future directions for research in more detail.

A. Guiding Abstractions

Researchers often benefit from abstractions to guide their work. We already discussed a few that provide perspectives on the space of circumvention approaches: steganography vs. polymorphism, setup vs. usage of channels, and the nature of the exploit's detection activity.

We now consider abstractions designed to guide the researcher's thinking when selecting how to evaluate an approach. We view these through the lens of the central question for circumvention evaluation: *How do we define success?* Since environments and use cases vary, we do not seek to provide a fixed list of criteria that approaches must consider. Rather, we aim to illuminate the tradeoffs involved in adopting various evaluation criteria by considering notions of success concrete enough to serve as common guides, but flexible enough to adapt to different environments and use cases.

For example, we might consider the volume of *goodput* that a tool enables, where goodput refers to *productive evading traffic*. If we view our overarching goal as resisting the impediments to free communication that censorship imposes, then we might well deem deployed approach *A* as doing better in this regard than deployment *B* if, at the end of the day—and for whatever reasons—*A* will allow users to successfully conduct more overall circumventing traffic than *B* will.

We can refine this abstraction based on the premise that censors and evaders engage in an ongoing arms race with an ebb and flow largely determined by economic concerns. That perspective leads us to consider an abstract metric of success based on costs: the amount of goodput that an approach provides at a given cost to the advocates who must select among proposed approaches. This metric highlights that circumvention approaches are tools for converting resources (costs) into products (goodput).

We consider such metrics as *abstract* for two reasons. First, while intuitive and recognizable enough to help us organize the problem space, they are parametric in how we calculate goodput and cost. In this framework, just what constitutes *goodput* is by design a value judgement: a dissident communicating a single picture from a demonstration might have

much more productive value than thousands of users accessing banned YouTube videos. Similarly, different advocates may assign different prices to goodput for the same approach if they value resources differently.

Second, we do not expect evaluations to actually compute these values, since they require information often unknown. But these abstract metrics provide fruitful *touchstones*: they underscore how for assessing utility of an approach many considerations can come into play beyond purely technical concerns such as worst-case blocking vulnerabilities. From this perspective, we can rate the concrete evaluation criteria such as those identified in our survey based on how well in isolation and in combination, they predict abstract metrics such as goodput-per-cost. We might then look to approach-specific evaluations to provide evidence that an approach could *in practice* drive up a censor's costs.

Our perspective challenges the narrower views of some prior evaluations. By examining total cost, we remind the evaluator that every aspect of the traffic produced by the evasion approach matters, not simply those considered by its designer. We seek with such a universal view to encourage designers to widen their focus and identify often simple countermeasures that have undermined past approaches.

While as stated we pose the cost-of-goodput metric solely in terms of an advocate's costs, it naturally extends to the costs of censors and users. An approach inexpensively blocked by the censor will produce no goodput for the advocate's investment; an approach whose high cost to users drives them away will likewise make for a poor investment.

This relationship also highlights possibilities for *cost shifting*. Users could promote approaches they value by paying advocates to maintain them (such as reflected by the common practice of many paid VPN services).

Research Gap 7. *Little research exists in informing censorship circumvention approaches with cost shifting. Approaches aiming to also provide anonymity may pose additional research questions in this direction given the challenges of anonymous billing.*

B. Understanding Censors and Their Technical Measures

Since practical evaluation criteria will depend in part upon the nature of the relevant censors, we need good methods of understanding censors to determine the most effective ways to evaluate circumvention tools. As noted above (Gap 1), little research examines how censors operate.

To this end, the circumvention community would benefit from tools that systematically experiment on censors to determine how they block traffic from a given circumvention technology. Such tools will not only speed up the response to censorship events, but also complement *in situ* and retrospective measurement studies. The improved models learned from the responses of censors will enable developers to design evasion approaches that better anticipate future censorship countermeasures.

Furthermore, such tools would allow circumventors to respond more quickly to new censorship attacks. When a censor blocks a tool, circumvention developers respond by determining the features that the censor has started using to distinguish evading traffic from allowed traffic. For example, currently the Tor project mostly relies on end-users under a censorship regime reporting blocking events, and then finding a fix by tweaking evading traffic until it gets past the censor, making adjustments in an ad hoc fashion. This unsystematic process proves time-consuming and provides only narrow illumination of the censor's behavior. A survey we conducted of Tor's issues tracker [153] found that it can take from a couple of hours (e.g., Iran's DPI exploit based on certificate lifetime) to a couple of months (e.g., China's active probing of bridges) to identify the censorship mechanism. It also requires significant time and burdens the developer community. Thus, a debugging-like tool to identify how a censor blocks a given evasion tool would have significant utility.

C. Understanding the Arms Race

As noted above (Gap 6), we know little about how long it takes for a censor to deploy new technology in response to a new circumvention approach. Thus, it is difficult to know whether easy-come-easy-go approaches proposed by others would be worth the effort of deployment [59, 60, 154]. The circumvention community would benefit from automated systems for detecting new censorship actions, which would enable early detection of censorship events, detailed measurements as these events unfold, and comprehensive analyses to understand the speed of the arms race.

The community would also benefit from a better understanding of the internal dynamics of the organizations that implement the censor's policies. Understanding their organizational structure could lead to approaches that cut across it, leading to a diffusion of responsibility and perhaps delayed responses.

Furthermore, as observed in Section IV, the responses of censors appear as driven by political developments as technical. Much as they ratchet up censorship by deploying new attacks around politically sensitive times, circumventors could hold back new approaches for release during those times, helping channels to operate when they are needed most.

D. Evaluation Engines

Moving forward, researchers could develop evaluation engines that implement recommendations such as those we frame in this work. In particular, researchers could create automated evaluation systems that identify the types of vulnerabilities exploited by real censors. As discussed before Recommendation 5, academic work has largely examined complex but well-known features, such as packet size distributions and entropy (e.g., [18, 19, 22–24, 26, 34, 41, 48, 49, 53]). The community would benefit from engines that identify subtle but simple vulnerabilities, such as using telltale cipher suites. In particular, the employment of machine learning with an emphasis on using it to illuminate feature selection could provide a useful starting point for such engines.

IX. Conclusion

We have focused in this work on comparing theory to practice in order to stimulate research addressing the circumvention problems of today. We do not mean to suggest that

forward-looking research serves no purpose; clearly, censors continually evolve toward increasingly sophisticated blocking, and thus the future will require increasingly sophisticated approaches to circumvention. However, our examination highlights significant gaps in the research literature. Among these, we note that the field lacks methods of evaluating approaches against vulnerabilities that are difficult to find, but easy for the censor to exploit once found, like those used in practice. Given the limited resources available for research, our survey points up significant concerns that the current focus on sophisticated attacks that *could* arise in the future may come at the expense of more effectively addressing the realistic attacks of today.

Acknowledgements. We thank the numerous tool authors who responded to our questions about their evaluations and the anonymous reviewers of our submission. We gratefully acknowledge funding support from the Freedom 2 Connect Foundation, Intel, the National Science Foundation (grants 0424422, 1237265, 1223717, and 1518918), the Open Technology Fund, the US Department of State Bureau of Democracy, Human Rights, and Labor. The opinions in this paper are those of the authors and do not necessarily reflect the opinions of any funding sponsor or the United States Government.

References

[1] A. Houmansadr, C. Brubaker, and V. Shmatikov, "The parrot is dead: Observing unobservable network communications," in *2013 IEEE Symp. on Security and Privacy*, ser. SP '13. IEEE Computer Society, 2013, pp. 65–79.

[2] The Tor Project, "BridgeDB," https://bridges.torproject.org/.

[3] P. Lincoln, I. Mason, P. Porras, V. Yegneswaran, Z. Weinberg, J. Massar, W. Simpson, P. Vixie, and D. Boneh, "Bootstrapping communications into an anti-censorship system," in *Free and Open Communications on the Internet*. USENIX, 2012.

[4] N. Feamster, M. Balazinska, G. Harfst, H. Balakrishnan, and D. R. Karger, "Infranet: Circumventing web censorship and surveillance," in *USENIX Security Symp.*, 2002, pp. 247–262.

[5] T. Ruffing, J. Schneider, and A. Kate, "Identity-based steganography and its applications to censorship resistance," in *Hot Topics in Privacy Enhancing Technologies*. Springer, 2013.

[6] L. Invernizzi, C. Kruegel, and G. Vigna, "Message In A Bottle: Sailing past censorship," in *Annual Computer Security Applications Conf.* ACM, 2013.

[7] C. Connolly, P. Lincoln, I. Mason, and V. Yegneswaran, "TRIST: Circumventing censorship with transcoding-resistant image steganography," in *Free and Open Communications on the Internet*. USENIX, 2014.

[8] Ludde, uau, The_8472, Parg, and Nolar, "Message stream encryption," https://wiki.vuze.com/w/Message_Stream_Encryption, Feb. 2006, accessed Aug. 10, 2015.

[9] S. Ho, "Feed Over Email," https://code.google.com/p/foe-project/.

[10] IT-Consulting Ulf Borcherdt und Bjoern Glaessner GbR, "Mailmyweb," https://www.mailmyweb.com/index.php/en/.

[11] S. Cao, L. He, Z. Li, and Y. Yang, "SkyF2F: Censorship resistant via skype overlay network," in *Intl. Conf. on Information Engineering*. IEEE, 2009, pp. 350–354.

[12] S. Burnett, N. Feamster, and S. Vempala, "Chipping away at censorship firewalls with user-generated content," in *USENIX Security Symp.* USENIX, 2010.

[13] A. Houmansadr, G. T. K. Nguyen, M. Caesar, and N. Borisov, "Cirripede: Circumvention infrastructure using router redirection with plausible deniability," in *Computer and Communications Security*. ACM, 2011, pp. 187–200.

[14] J. Karlin, D. Ellard, A. W. Jackson, C. E. Jones, G. Lauer, D. P. Mankins, and W. T. Strayer, "Decoy routing: Toward unblockable Internet communication," in *Free and Open Communications on the Internet*. USENIX, 2011.

[15] E. Wustrow, S. Wolchok, I. Goldberg, and J. A. Halderman, "Telex: Anticensorship in the network infrastructure," in *USENIX Security Symp.*, 2011.

[16] Q. Wang, X. Gong, G. T. K. Nguyen, A. Houmansadr, and N. Borisov, "CensorSpoofer: Asymmetric communication using IP spoofing for censorship-resistant web browsing," in *Computer and Communications Security*. ACM, 2012.

[17] D. Fifield, N. Hardison, J. Ellithorpe, E. Stark, D. Boneh, R. Dingledine, and P. Porras, "Evading censorship with browser-based proxies," in *12th Intl. Conf. on Privacy Enhancing Technologies*, ser. PETS'12. Springer-Verlag, 2012, pp. 239–258.

[18] H. Mohajeri Moghaddam, B. Li, M. Derakhshani, and I. Goldberg, "SkypeMorph: Protocol obfuscation for Tor bridges," in *2012 ACM conf. on Computer and communications security*, 2012, pp. 97–108.

[19] Z. Weinberg, J. Wang, V. Yegneswaran, L. Briesemeister, S. Cheung, F. Wang, and D. Boneh, "StegoTorus: A camouflage proxy for the Tor anonymity system," in *2012 ACM conf. on Computer and communications security*. ACM, 2012, pp. 109–120.

[20] G. Kadianakis and N. Mathewson, "obfs2 (the twobfuscator)," Jan. 2011, https://gitweb.torproject.org/pluggable-transports/obfsproxy.git/tree/doc/obfs2/obfs2-protocol-spec.txt.

[21] ——, "obfs3 (the threebfuscator)," Jan. 2013, https://gitweb.torproject.org/pluggable-transports/obfsproxy.git/tree/doc/obfs3/obfs3-protocol-spec.txt.

[22] Y. Angel and P. Winter, "obfs4 (the obfourscator)," May 2014, https://gitweb.torproject.org/pluggable-transports/obfs4.git/tree/doc/obfs4-spec.txt.

[23] P. Winter, T. Pulls, and J. Fuss, "ScrambleSuit: A polymorphic network protocol to circumvent censorship," in *Wksp. on Privacy in the Electronic Society*. ACM, 2013, uRL: http://www.cs.kau.se/philwint/pdf/wpes2013.pdf.

[24] B. Wiley, "Dust: A blocking-resistant Internet transport protocol," Available at http://blanu.net/Dust.pdf, 2011.

[25] GoAgent, "GoAgent," https://github.com/goagent/goagent.

[26] D. Fifield, C. Lan, R. Hynes, P. Wegmann, and V. Paxson, "Blocking-resistant communication through domain fronting," *Privacy Enhancing Technologies*, vol. 1, no. 2, 2015.

[27] D. Fifield, G. Nakibly, and D. Boneh, "OSS: Using online scanning services for censorship circumvention," in *Privacy Enhancing Technologies Symp.* Springer, 2013.

[28] K. P. Dyer, S. E. Coull, T. Ristenpart, and T. Shrimpton, "Protocol misidentification made easy with format-transforming encryption," in *2013 ACM SIGSAC Conf. on Computer & Communications Security*. ACM, 2013, pp. 61–72.

[29] K. P. Dyer, S. E. Coull, and T. Shrimpton, "Marionette: A programmable network traffic obfuscation system," in *24th USENIX Security Symposium (USENIX Security 15)*, Aug. 2015.

[30] A. Houmansadr, T. J. Riedl, N. Borisov, and A. C. Singer, "I want my voice to be heard: IP over Voice-over-IP for unobservable censorship circumvention." in *NDSS*, 2013.

[31] W. Zhou, A. Houmansadr, M. Caesar, and N. Borisov, "SWEET: Serving the web by exploiting email tunnels," in *Hot Topics in Privacy Enhancing Technologies*. Springer, 2013.

[32] D. Nobori and Y. Shinjo, "VPN Gate: A volunteer-organized public VPN relay system with blocking resistance for bypassing government censorship firewalls," in *Networked Systems Design and Implementation*. USENIX, 2014.

[33] B. Jones, S. Burnett, N. Feamster, S. Donovan, S. Grover, S. Gunasekaran, and K. Habak, "Facade: High-throughput, deniable censorship circumvention using web search," in *Free and Open Communications on the Internet*. USENIX, 2014.

[34] S. Li, M. Schliep, and N. Hopper, "Facet: Streaming over videoconferencing for censorship circumvention," in *WPES*, 2014.

[35] E. Wustrow, C. M. Swanson, and J. A. Halderman, "TapDance: End-to-middle anticensorship without flow blocking," in *USENIX Security Symp.* USENIX, 2014.

[36] C. Brubaker, A. Houmansadr, and V. Shmatikov, "CloudTransport: Using cloud storage for censorship-resistant networking," in *Privacy Enhancing Technologies Symp.* Springer, 2014.

[37] University of Washington, "uProxy," https://www.uproxy.org/.

[38] J. Marshall, "CGIProxy," http://www.jmarshall.com/tools/cgiproxy/.

[39] Ultrareach Internet Corporation, "Ultrasurf," http://ultrasurf.us/.

[40] Dynamic Internet Technology, Inc., "Freegate," http://dit-inc.us/freegate.html.

[41] Psiphon Inc, "Psiphon 3 circumvention system README," https://bitbucket.org/psiphon/psiphon-circumvention-system.

[42] Brave New Software Project, Inc, "Lantern," https://getlantern.org/.

[43] D. Octavian, "bit-smuggler," https://github.com/danoctavian/bit-smuggler.

[44] Global Information Freedom, Inc., "GTunnel," http://www.internetfreedom.org/GTunnel.html.

[45] AnchorFree, "Hotspot Shield," http://www.hotspotshield.com/.

[46] JAP Team, "JAP," https://anon.inf.tu-dresden.de/index_en.html.

[47] resolution Reichert Network Solutions GmbH, "Your Freedom," https://www.your-freedom.net/.

[48] B. Hahn, R. Nithyanand, P. Gill, and R. Johnson, "Games without frontiers: Investigating video games as a covert channel," in *IEEE European Symp. on Security and Privacy (Euro S&P)*, 2016, pp. 63–77, arXiv report available: http://arxiv.org/abs/1503.05904.

[49] P. Vines and T. Kohno, "Rook: Using video games as a low-bandwidth censorship resistant communication platform," https://homes.cs.washington.edu/~yoshi/papers/tech-report-rook.pdf, 2015.

[50] J. Holowczak and A. Houmansadr, "CacheBrowser: Bypassing Chinese censorship without proxies using cached content," in *Computer and Communications Security*. ACM, 2015.

[51] Simurgh Proxy, Inc., "Green Simurgh," http://simurghesabz.net/.

[52] D. Ellard, A. Jackson, C. Jones, V. U. Manfredi, T. Strayer, B. Thapa, and M. V. Welie, "Rebound: Decoy routing on asymmetric routes via error messages," in *Local Computer Networks*. IEEE, 2015.

[53] Y. Wang, P. Ji, B. Ye, P. Wang, R. Luo, and H. Yang, "GoHop: Personal VPN to defend from censorship," in *Intl. Conf. on Advanced Communication Technology*. IEEE, 2014.

[54] C. Callanan, H. Dries-Ziekenheiner, A. Escudero-Pascual, and R. Guerra, "Leaping over the firewall: A review of censorship circumvention tools," Freedom House, Tech. Rep., 2011.

[55] A. Escudero-Pascual, "Circumvention is not privacy!" Jul. 2010.

[56] H. Roberts, E. Zuckerman, and J. Palfrey, "2011 circumvention tool evaluation," Berkman Center for Internet and Society, Tech. Rep., Aug. 2011.

[57] R. Dingledine, "Ten things to look for in a circumvention tool," https://www.torproject.org/press/presskit/2010-09-16-circumvention-features.pdf, Jul. 2010.

[58] T. Elahi and I. Goldberg, "CORDON–a taxonomy of Internet censorship resistance strategies," Centre for Applied Cryptographic Research (CACR), University of Waterloo, Tech. Rep. 2012-33, 2012.

[59] T. Elahi, C. M. Swanson, and I. Goldberg, "Slipping past the cordon: A systematization of Internet censorship resistance," Centre for Applied Cryptographic Research (CACR), University of Waterloo, Tech. Rep. 2015-10, Aug. 2015.

[60] S. Khattak, L. Simon, and S. J. Murdoch, "Systemization of pluggable transports for censorship resistance," ArXiv CoRR, Tech. Rep. 1412.7448, 2014.

[61] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: The second-generation onion router," in *13th USENIX Security Symp.*, Aug. 2004.

[62] Anonymous, "Torproject.org blocked by GFW in China: Sooner or later?" Tor Blog, Jun. 2008, https://blog.torproject.org/blog/torprojectorg-blocked-gfw-china-sooner-or-later.

[63] A. Lewman, "Tor partially blocked in China," Tor Blog, Sep. 2009, https://blog.torproject.org/blog/tor-partially-blocked-china, Accessed Oct. 29, 2014.

[64] R. Dingledine and J. Appelbaum, "How governments have tried to block Tor," 28th Chaos Communication Congress, Dec. 2012.

[65] R. Dingledine and N. Mathewson, "Design of a blocking-resistant anonymity system," The Tor Project, Tech. Rep., 2006.

[66] R. Dingledine, "Please run a bridge relay! (was Re: Tor 0.2.0.13-alpha is out)," https://lists.torproject.org/pipermail/tor-talk/2007-December/003854.html, Dec. 2007.

[67] A. Lewman, "China blocking Tor: Round two," https://blog.torproject.org/blog/china-blocking-tor-round-two, Mar. 2010, accessed Oct. 29, 2014.

[68] G. Kadianakis, "GFW probes based on Tor's SSL cipher list," Tor Trac ticket, https://bugs.torproject.org/4744.

[69] N. Mathewson, "TLS history," https://trac.torproject.org/projects/tor/wiki/org/projects/Tor/TLSHistory, May 2012.

[70] T. Wilde, "Great Firewall Tor Probing Circa 09 DEC 2011," https://gist.github.com/twilde/da3c7a9af01d74cd7de7, Jan. 2012, accessed Oct. 29, 2014.

[71] R. Ensafi, D. Fifield, P. Winter, N. Feamster, N. Weaver, and V. Paxson, "Examining how the Great Firewall discovers hidden circumvention servers," in *Internet Measurement Conf.* ACM, 2015, pp. 445–458.

[72] S. Köpsell and U. Hillig, "How to achieve blocking resistance for existing systems enabling anonymous web surfing," in *2004 ACM Wksp. on Privacy in the Electronic Society*, ser. WPES '04. ACM, 2004, pp. 47–58.

[73] D. Robinson, H. Yu, and A. An, "Collateral freedom: A snapshot of Chinese Internet users circumventing censorship," Open Internet Tools Project Report, Apr. 2013.

[74] C. S. Leberknight, M. Chiang, H. V. Poor, and F. Wong, "A taxonomy of Internet censorship and anti-censorship," http://www.princeton.edu/~chiangm/anticensorship.pdf, Dec. 2010, accessed Nov. 18, 2014.

[75] C. S. Leberknight, M. Chiang, and F. M. F. Wong, "A taxonomy of censors and anti-censors part II: Anti-censorship technologies," *Int. J. E-Polit.*, vol. 3, no. 4, pp. 20–35, Oct. 2012.

[76] A. Pfitzmann and M. Hansen, "A terminology for talking about privacy by data minimization: Anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management," 2010.

[77] J. Geddes, M. Schuchard, and N. Hopper, "Cover your ACKs: Pitfalls of covert channel censorship circumvention," in *Computer and Communications Security*. ACM, 2013.

[78] L. Wang, K. P. Dyer, A. Akella, T. Ristenpart, and T. Shrimpton, "Seeing through network-protocol obfuscation," in *22nd ACM SIGSAC Conf. on Computer and Communications Security*, ser. CCS '15. ACM, 2015, pp. 57–69.

[79] 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 (FOCI)*. USENIX, 2015.

[80] J. Zittrain and B. G. Edelman, "Internet filtering in China," *IEEE Internet Computing*, vol. 7, no. 2, pp. 70–77, Mar. 2003.

[81] S. Khattak, M. Javed, P. D. Anderson, and V. Paxson, "Towards illuminating a censorship monitor's model to facilitate evasion," in *The 3rd USENIX Wksp. on Free and Open Communications on the Internet*. USENIX, 2013.

[82] S. Aryan, H. Aryan, and J. A. Halderman, "Internet censorship in Iran: A first look," *Free and Open Communications on the Internet, Washington, DC, USA*, 2013.

[83] Z. Nabi, "The anatomy of web censorship in Pakistan," in *Presented as part of the 3rd USENIX Wksp. on Free and Open Communications on the Internet*. USENIX, 2013.

[84] P. Winter and S. Lindskog, "How the Great Firewall of China is blocking Tor," in *Free and Open Communications on the Internet*. USENIX, 2012.

[85] R. Clayton, S. J. Murdoch, and R. N. M. Watson, "Ignoring the Great Firewall of China," in *Privacy Enhancing Technologies*. Springer, 2006, pp. 20–35.

[86] R. Clayton, "Failures in a hybrid content blocking system," in *Privacy Enhancing Technologies*. Springer, 2006, pp. 78–92.

[87] J. R. Crandall, D. Zinn, M. Byrd, E. Barr, and R. East, "ConceptDoppler: A weather tracker for Internet censorship," in *Computer and Communications Security*. ACM, 2007, pp. 352–365.

[88] C. Anderson, "Dimming the Internet: Detecting throttling as a mechanism of censorship in Iran," *arXiv preprint arXiv:1306.4361*, 2013.

[89] X. Xu, Z. M. Mao, and J. A. Halderman, "Internet censorship in China: Where does the filtering occur?" in *Passive and Active Measurement Conf.* Springer, 2011, pp. 133–142.

[90] J.-P. Verkamp and M. Gupta, "Inferring mechanics of web censorship around the world," in *Free and Open Communications on the Internet*. USENIX, 2012.

[91] A. Sfakianakis, E. Athanasopoulos, and S. Ioannidis, "CensMon: A web censorship monitor," in *Free and Open Communications on the Internet*. USENIX, 2011.

[92] J. C. Park and J. R. Crandall, "Empirical study of a national-scale distributed intrusion detection system: Backbone-level filtering of HTML responses in China," in *Distributed Computing Systems*. IEEE, 2010, pp. 315–326.

[93] D. Anderson, "Splinternet behind the Great Firewall of China," *Queue*, vol. 10, no. 11, p. 40, 2012.

[94] Anonymous, "Towards a comprehensive picture of the Great Firewall's DNS censorship," in *Free and Open Communications on the Internet*. USENIX, 2014.

[95] Y. Akdeniz, "Report of the OSCE representative on freedom of the media on Turkey and Internet censorship," http://www.osce.org/fom/41091?download=true.

[96] C. Anderson, P. Winter, and Roya, "Global network interference detection over the RIPE Atlas network," in *4th USENIX Wksp. on Free and Open Communications on the Internet (FOCI 14)*. USENIX Association, Aug. 2014.

[97] M. Marquis-Boire, J. Dalek, and S. McKune, "Planet Blue Coat: Mapping global censorship and surveillance tools," https://citizenlab.org/2013/01/planet-blue-coat-mapping-global-censorship-and-surveillance-tools/, Jan. 2013.

[98] A. Chaabane, T. Chen, M. Cunche, E. D. Cristofaro, A. Friedman, and M. A. Kaafar, "Censorship in the wild: Analyzing Internet filtering in Syria," in *Internet Measurement Conf.* ACM, 2014.

[99] A. Dainotti, C. Squarcella, E. Aben, K. C. Claffy, M. Chiesa, M. Russo, and A. Pescapé, "Analysis of country-wide Internet outages caused by censorship," in *Internet Measurement Conf.* ACM, 2011, pp. 1–18.

[100] S. Wolfgarten, "Investigating large-scale Internet content filtering," http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.133.5778&rep=rep1&type=pdf, Aug. 2006.

[101] OpenNet Initiative, "Internet filtering in China in 2004-2005: A country study," https://opennet.net/sites/opennet.net/files/ONI_China_Country_Study.pdf, 2005.

[102] ——, "Internet Filtering in Iran," https://opennet.net/sites/opennet.net/files/ONI_Iran_2009.pdf, 2009.

[103] Citizen Lab, "O Pakistan, we stand on guard for thee: An analysis of Canada-based Netsweeper's role in Pakistan's censorship regime," https://citizenlab.org/2013/06/o-pakistan/, Jun. 2013.

[104] 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 Conf.* ACM, 2013.

[105] R. Ensafi, P. Winter, A. Mueen, and J. R. Crandall, "Analyzing the Great Firewall of China over space and time," *Privacy Enhancing Technologies*, vol. 1, no. 1, 2015.

[106] M. Dornseif, "Government mandated blocking of foreign Web content," in *17. DFN-Arbeitstagung über Kommunikationsnetze*, ser. Lecture Notes in Informatics, J. Von Knop, W. Haverkamp, and E. Jessen, Eds., 2003, pp. 617–648.

[107] G. Lowe, P. Winters, and M. L. Marcus, "The great DNS wall of China," New York University, Tech. Rep., 2007.

[108] P. Gill, M. Crete-Nishihata, J. Dalek, S. Goldberg, A. Senft, and G. Wiseman, "Characterizing web censorship worldwide: Another look at the OpenNet Initiative data," *Transactions on the Web*, vol. 9, no. 1, pp. 4:1–4:29, Jan. 2015.

[109] P. Winter, "Towards a censorship analyser for Tor," in *The 3rd USENIX Wksp. on Free and Open Communications on the Internet*. USENIX, 2013.

[110] S. Burnett and N. Feamster, "Encore: Lightweight measurement of web censorship with cross-origin requests," *ACM SIGCOMM Computer Communication Review*, vol. 45, no. 4, pp. 653–667, 2015.

[111] D. Fifield and L. Tsai, "Censors' delay in blocking circumvention proxies," in *6th USENIX Wksp. on Free and Open Communications on the Internet (FOCI 16)*. USENIX Association, 2016.

[112] R. Ensafi, P. Winter, A. Mueen, and J. R. Crandall, "Large-scale spatiotemporal characterization of inconsistencies in the world's largest firewall," *arXiv preprint arXiv:1410.0735*, 2014.

[113] OpenNet Initiative, "Pakistan," 2010, https://opennet.net/sites/opennet.net/files/ONI_Pakistan_2010.pdf.

[114] Citizen Lab, "Behind Blue Coat: Investigations of commercial filtering in Syria and Burma," https://citizenlab.org/2011/11/behind-blue-coat/, Nov. 2011.

[115] OpenNet Initiative, "Internet Filtering in Iran in 2004–2005," https://opennet.net/sites/opennet.net/files/ONI_Country_Study_Iran.pdf, 2005.

[116] M. Bevand, "My experience with the Great Firewall of China," Zorinaq: mrb's blog, Jan. 2016, http://blog.zorinaq.com/?e=81.

[117] G. Esfandiari, "Iran admits throttling internet to 'preserve calm' during election," Radio Free Europe/Radio Liberty, Jun. 2013, http://www.rferl.org/content/iran-internet-disruptions-election/25028696.html.

[118] hrimfaxi, "Bridge easily detected by GFW," Tor Trac ticket, https://bugs.torproject.org/4185.

[119] P. Winter, "GFW actively probes obfs2 bridges," Tor Trac ticket, Mar. 2013, https://bugs.torproject.org/8591.

[120] G. Kadianakis, "Ethiopia blocks Tor based on ServerHello," Tor Trac ticket, https://bugs.torproject.org/6045.

[121] G. Kadianakis, P. Winter, I. A. Lovecruft, and Anonymous, "Censorship by country: Iran," OONI Censorship Wiki, https://trac.torproject.org/projects/tor/wiki/doc/OONI/censorshipwiki/CensorshipByCountry/Iran.

[122] A. Lewman, "Update on Internet censorship in Iran," Tor Blog, https://blog.torproject.org/blog/update-internet-censorship-iran.

[123] R. Dingledine, "Hack up stunnel to test a transport that uses a vanilla SSL handshake," Tor Trac ticket, https://bugs.torproject.org/4248.

[124] ——, "Iran blocks Tor; Tor releases same-day fix," Tor Blog, https://blog.torproject.org/blog/iran-blocks-tor-tor-releases-same-day-fix.

[125] P. Winter, "How is Iran blocking Tor?" Tor Trac ticket, https://bugs.torproject.org/7141.

[126] A. Lewman, "Iran partially blocks encrypted network traffic," Tor Blog, Feb. 2012, https://blog.torproject.org/blog/iran-partially-blocks-encrypted-network-traffic.

[127] Iran Media, "Persian cyberspace report: internet blackouts across Iran; BBC journalists interrogated, family members imprisoned," blog post, Feb. 2012, http://iranmediaresearch.com/en/blog/101/12/02/09/840.

[128] R. Dingledine, "SSL handshake filtered when MAX_SSL_KEY_LIFETIME_ADVERTISED is 365 days," Tor Trac ticket, Mar. 2013, https://bugs.torproject.org/8443.

[129] Small Media, "Iranian Internet infrastructure and policy report," white paper, Apr. 2013, http://smallmedia.org.uk/InfoFlowReportAPRIL.pdf.

[130] C. Anderson, "Vanilla Tor connectivity issues in Iran – directory authorities blocked?" Tor Trac ticket, Jul. 2014, https://bugs.torproject.org/12727.

[131] R. Sandvik, "Kazakhstan uses DPI to block Tor," https://bugs.torproject.org/6140, Jun. 2012, accessed Oct. 29, 2014.

[132] P. Winter, "Censorship by country: The Philippines," OONI Censorship Wiki, https://trac.torproject.org/projects/tor/wiki/doc/OONI/censorshipwiki/CensorshipByCountry/The_Philippines.

[133] ——, "The Philippines are blocking Tor?" Tor Trac ticket, Jun. 2012, https://bugs.torproject.org/6258.

[134] ——, "Censorship by country: Syria," OONI Censorship Wiki, https://trac.torproject.org/projects/tor/wiki/doc/OONI/censorshipwiki/CensorshipByCountry/Syria.

[135] R. Sandvik, "UAE uses DPI to block Tor," Tor Trac ticket, Jun. 2012, https://bugs.torproject.org/6246.

[136] ——, "Ethiopia introduces deep packet inspection," https://blog.torproject.org/blog/ethiopia-introduces-deep-packet-inspection, May 2012, accessed Oct. 29, 2014.

[137] C. Anderson, "Dimming the Internet: Detecting throttling as a mechanism of censorship in Iran," University of Pennsylvania, Tech. Rep., 2013.

[138] J. Appelbaum, "Recent events in Egypt," Tor Blog, Jan. 2011, https://blog.torproject.org/blog/recent-events-egypt.

[139] H. Post, "Libya Internet shut down amid protests, later restored (update)," http://www.huffingtonpost.com/2011/02/18/libya-internet-shut-down-_n_825473.html, Feb. 2011, accessed Oct. 29, 2014.

[140] M. Chulov, "Syria shuts off internet access across the country," The Guardian, Nov. 2012, http://www.theguardian.com/world/2012/nov/29/syria-blocks-internet Accessed Oct. 29, 2014.

[141] P. Boehler, "US-funded Lantern program allows Chinese to dodge Great Firewall and view banned websites," South China Morning Post, Dec. 2013, http://www.scmp.com/news/china-insider/article/1372661/use-lantern-software-means-view-banned-websites-grows-china.

[142] ——, "Anti-firewall tool Lantern infiltrated by Chinese censors," South China Morning Post, Dec. 2013, http://www.scmp.com/news/china-insider/article/1378201/anti-firewall-tool-lantern-infiltrated-chinese-censors.

[143] D. Pauli, "Massive DDoS racks up $30,000-a-day Amazon bill for China activists," The Register, Mar. 2015, http://www.theregister.co.uk/2015/03/20/greatfire_chinese_activists_under_ddos/.

[144] C. Smith, "We are under attack," GreatFire blog, Mar. 2015, https://en.greatfire.org/blog/2015/mar/we-are-under-attack.

[145] Fox-IT, "DigiNotar certificate authority breach," Sep. 2011, https://www.rijksoverheid.nl/ministeries/bzk/documenten/rapporten/2011/09/05/diginotar-public-report-version-1.

[146] J. Appelbaum, "The DigiNotar debacle, and what you should do about it," Tor Blog, Aug. 2011, https://blog.torproject.org/blog/diginotar-debacle-and-what-you-should-do-about-it.

[147] M. Johnson, "China, GitHub and the man-in-the-middle," GreatFire blog, Jan. 2013, https://en.greatfire.org/blog/2013/jan/china-github-and-man-middle.

[148] M. Marquis-Boire, "Iranian anti-censorship software 'Simurgh' circulated with malicious backdoor," Research Brief, The Citizen Lab, May 2012, https://citizenlab.org/2012/05/iranian-anti-censorship-software-simurgh-circulated-with-malicious-backdoor-2/.

[149] J. Holowczak and A. Houmansadr, "CacheBrowser: Bypassing Chinese censorship without proxies using cached content," in *22nd ACM SIGSAC Conf. on Computer and Communications Security*. ACM, 2015, pp. 70–83.

[150] N. Feamster, M. Balazinska, W. Wang, H. Balakrishnan, and D. Karger, "Thwarting web censorship with untrusted messenger discovery," in *Privacy Enhancing Technologies*, ser. LICS, R. Dingledine, Ed. Springer Berlin Heidelberg, 2003, vol. 2760, pp. 125–140.

[151] R. Smits, D. Jain, S. Pidcock, I. Goldberg, and U. Hengartner, "BridgeSPA: Improving Tor bridges with single packet authorization,"

in *10th Annual ACM Wksp. on Privacy in the Electronic Society*, ser. WPES '11. ACM, 2011, pp. 93–102.
[152] Q. Wang, Z. Lin, N. Borisov, and N. Hopper, "rbridge: User reputation based tor bridge distribution with privacy preservation." in *NDSS*. The Internet Society, 2013.
[153] The Tor Project, "Tor's combined bug tracker and wiki website," https://trac.torproject.org/, accessed Nov. 17, 2014.
[154] S. J. Murdoch and G. Kadianakis, "Pluggable transports roadmap," The Tor Project, Tech. Rep. 2012-03-003, 2012.

Appendix

Adaptability to blocking: Assesses the system's resilience to unexpected blocking events and new kinds of blocking.
Application support: Assesses the system's usefulness for a wide variety of applications (e.g. web browsing, chat, email).
Availability of documentation: Assesses the quality of user and developer documentation.
Availability of infrastructure: This criterion considers the availability of the infrastructure used by an approach for users and the feasibility of deployment.
Byte overhead: How many extra bytes are introduced by the tool.
CPU usage by users: Assesses the percentage of CPU cycles consumed by the client or server part of the system.
Clean uninstall: Assesses whether the client software leaves traces after being uninstalled.
Client performance: Assesses an approach's client program efficiency in terms of CPU and memory usage.
Concurrent connection count: They measured the number of simultaneous connections to a server.
Connection length: Measures the duration of flows (e.g., TCP connections) and evaluates whether the duration can be a distinguisher. (If a connection is suspiciously long, for example.)
Cost of external services: Estimates the cost of external services, e.g., cloud services, CDNs, that are required to deploy an approach.
Decentralization of trust: Evaluates the degree to which trust is centralized; i.e., all trust is placed in a single server/company, or spread out among many parties.
Deniability under computer inspection: Whether the circumvention tool users can plausibly deny using it even when their computers are inspected.
Developed by experts: Assesses whether the developers of the tool are known security experts.
Diversity of users: Assesses the diversity of types of users of the system.
Does not store user information: Does not log user information.
End-user protection: Assesses whether an approach protects user's privacy from intermediate and end nodes.
Fraction of clients that can utilize the network: Assesses the number of clients (source hosts) in the Internet that would be able to join a system.
Free/low cost: Cost in USD to use the system.
Geographic diversity of proxies: Examines the diversity of proxies in terms of geographic location.
Goodput: The amount of useful throughput the tool enables.
Has a GUI: Assesses whether the client software has a graphical user interface.
Hide user information from end host: The tool hides information about the user from the end host (destination) to which the user connects to provide some degree of anonymity.
Ignore invalid connections (DoS): Whether an approach ignores invalid connections to avoid denial of service attacks.
Independent deployment: Can deploy the circumvention approach without needing the help of third-parties, such as friendly ISPs.
Indirect connection to forwarder: The circumventor's computer connects indirectly to the circumvention network's forwarders through an innocuous server.
Infrastructure cost: Assesses the cost of infrastructure required to deploy an approach in real world.
Inter-packet timing: Measures the distribution of packet timing (interpacket times or packet rates) to assess whether it is unlike that of a blocked protocol, or like that of an allowed protocol.
Latency: Assesses the round-trip time for a request.
Limit service to each user ID (DoS): Avoid DoS by limiting the amount of service the approach will provide to each user.
Localization: Assesses whether the software and documentation are localized to relevant languages.
Matching allowed n-gram distribution: Considers the distribution of consecutive strings of symbols, for example bytes. This includes 1-grams (e.g., distribution of single byte values).
Memory usage by users: Assesses the memory requirements to run the system.
Network performance: Assesses the system's performance in terms of goodput, latency and overhead.
Network stack fingerprinting: Assesses the fingerprintability of the network stack, for example comparing the TCP options of two different hosts. Relevant when one host spoofs packets on behalf of another.
No installation: Using the tool does not require installing special software.
No usage limitation: Assesses whether an approach artificially limits who can use it and for which service.
Number of HTTP requests/responses: Measures the total number of HTTP request-response pairs per TCP connection.
Number of errors per webpage: This criterion is specific to link-rewriting web proxies like CGIProxy that actually have to interpret HTML and JavaScript and change links so they point back into the proxy and not to their original location.
Number of proxies: The number of proxies usable with the tool.
Number of requests needed to retrieve data: Assesses the number of requests that a requester must make to retrieve hidden messages.
Number of unique connections: Discusses the number unique of IP addresses that connect to the system on a daily basis.
Number of users: The number of users the tool has.
Open source: The tool's source code is open.
Openness of design: Assesses whether the source code available (client and server) and whether the design public or relies on security through obscurity.
Packet size distribution: Measures the distribution of packet lengths to assess wehether it is unlike that of a blocked protocol, or like that of an allowed protocol.
Portability: Assesses the system's portability to different operating systems and devices.
Protocol misclassification rate: Assesses the misclassification rate of the protocol classifiers to see how well the tool can evade the classifiers.
Rate of proxy churn: Measures or estimates the rate at which new proxies appear and old proxies go away.
Registration performance: Some systems need to apply a special distinguisher or mark to traffic destined for circumvention. For example, end-to-middle proxying systems need to tag flows at the client side and recognize them at the station. This criterion considers the performance of the registration method.
Resistance to active probing: Active probing attacks involve the censor initiating connections to hosts to determine whether the host runs a given circumvention protocol, typically then blocking the host's IP address upon finding that it does. A system is resistant to active probing if an adversary cannot discover the use of the system using this technique.
Resistance to blocking: A system resists blocking if it is hard to block the protocol or IP address of the infrastructure that the approach uses, even given a method of identifying it. For example, if blocking would cause substantial collateral damage.
Resistance to insider attacks: Considers whether the system continues to work even if the censor joins the circumvention network and attempts to disrupt it.
Resistance to security attacks: This criterion considers different measures that a paper uses to avoid security attacks such as man-in-the-middle, denial of service, malicious proxy, key reuse and replay attack.
Resistance to traffic analysis: An approach is resistant to traffic analysis if an adversary cannot statically use properties of the traffic generated by the approach to detect it. (Some of the metrics used for this goal can also be used for active probing, but they are not inherently active.)
Resistance to traffic manipulation: Evaluates the system's resistance to modification of packets, or injecting or dropping packets. This criterion is concerned only with manipulation of client-initiated flows.
Respond to probes like something else: When probed, respond similar to how some allowed server would respond so that a censor deciding to block such responses will incur false positives.
Self promotion: Evaluates whether an approach or tool promotes itself in a way that is likely to attract harmful attention (from the media or from the censor, for example).
Serial connection count: Counted the number of connections made in a row to a server.
Server obfuscation: Keeping the server used as a forwarder by the circumvention network hidden from the censor.
Small download file: The size of the tool's client program file is small.
Software updates: Assesses the availability of software updates.
Speed of downloading a webpage: Assesses the time required to download a webpage. This is really a combination of goodput and latency, but it is specifically applied so often that we made it its own criterion.

Deployed Tools and Deployed & Papers:

	[2]	[8]	[9]	[10]	[20]	[21]	[22]	[25]	[37]	[38]	[39]	[40]	[41]	[42]	[44]	[45]	[47]	[51]	[23]	[26]	[28]	[32]	[46]	[53]
Network performance									☐								☐		☐	☐	☐	☐		☐
Fraction of clients that can utilize the network																								
Throughput																	☑		☑			☑		☑
Goodput																			☑		☑	☑		☑
Number of requests needed to retrieve data																								
Time overhead																								☑
Byte overhead																			☑	☑	☑			☑
Latency																								
Speed of downloading a webpage																			☑		☑			☑
Registration performance																								
Client performance																					☐			
Memory usage by users																					☑			
CPU usage by users																								
Startup time																								
Infrastructure cost																				☐				
Cost of external services																				☑				
Usability				☐					☐	☐	☐	☐	☐	☐	☐	☐	☐	☐		☐			☐	
Number of errors per webpage																								
Small download file				☑						☑	☑	☑	☑	☑	☑			☑						
Free/low cost											☑	☑	☑	☑	☑	☑	☑	☑					☑	
No installation				☑						☑	☑	☑	☑	☑										
Clean uninstall																								
No usage limitation																								
Software updates															☑									
Availability of documentation																								
Localization									☐						☑									
Portability															☑									
Application support																								
Has a GUI															☑									
Usage																	☐		☐	☐	☐	☐	☐	☐
Number of unique connections																						☑		
Number of users																	☑			☑		☑		
Test deployment																			☑	☑	☑	☑	☑	☑
Openness of design	☐	☐	☐		☐	☐	☐	☐	☐	☐			☐	☐					☐	☐	☐	☐	☐	☐
Open source	☑	☑	☑		☑	☑	☑	☑	☑	☑			☑	☑					☑	☑	☑	☑	☑	☑
Sustainable network and development																								
Developed by experts																								

Academic Papers Proposing Approaches:

	[3]	[4]	[5]	[6]	[7]	[11]	[12]	[13]	[14]	[15]	[16]	[17]	[18]	[19]	[24]	[27]	[29]	[30]	[31]	[33]	[34]	[35]	[36]	[43]	[48]	[49]	[50]	[52]
Network performance		☐		☐		☐	☐	☐		☐	☐	☐	☐	☐		☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Fraction of clients that can utilize the network								☑																				
Throughput								☑		☑		☑	☑											☑		☑		☑
Goodput						☑					☑		☑	☑		☑	☑	☑			☑				☑	☑		
Number of requests needed to retrieve data		☑																	☑									
Time overhead				☑			☑			☑							☑											
Byte overhead		☑					☑						☑							☑			☑					
Latency				☑										☑				☐	☑	☐	☐		☑	☑				
Speed of downloading a webpage											☑								☑			☑	☑				☑	☑
Registration performance								☑		☑												☑						
Client performance								☐													☐	☐						☐
Memory usage by users								☑													☑							
CPU usage by users								☑									☑				☑	☑						☑
Startup time																												
Infrastructure cost				☐		☐													☐				☐				☐	
Cost of external services				☑		☑													☑				☑				☐	
Usability	☐					☐						☐							☐		☐				☐		☐	
Number of errors per webpage																												
Small download file																									☑			
Free/low cost																												
No installation						☑						☑									☑							
Clean uninstall																												
No usage limitation																												
Software updates																												
Availability of documentation																												
Localization																												
Portability																												
Application support																												
Has a GUI																												
Usage												☐																
Number of unique connections																												
Number of users																												
Test deployment												☑																
Openness of design		☐		☐			☐			☐		☐	☐	☐	☐	☐	☐			☐	☐	☐		☐	☐			☐
Open source		☑		☑			☑			☑		☑	☑	☑	☑	☑	☑			☑	☑	☑		☑	☑			☑
Sustainable network and development																												
Developed by experts																												

Evaluation Papers:

	[54]	[55]	[56]	[57]
Network performance	☐	☐	☐	☐
Fraction of clients that can utilize the network				
Throughput				
Goodput		☐		☐
Number of requests needed to retrieve data				
Time overhead				
Byte overhead				
Latency		☐		☐
Speed of downloading a webpage			☐	
Registration performance				
Client performance				
Memory usage by users				
CPU usage by users				
Startup time	☐			
Infrastructure cost				
Cost of external services				
Usability	☐	☐	☐	☐
Number of errors per webpage			☐	
Small download file	☐	☐		☐
Free/low cost	☐		☐	
No installation	☐	☐		☐
Clean uninstall	☐	☐		
No usage limitation	☐	☐		☐
Software updates		☐		☐
Availability of documentation	☐	☐		☐
Localization	☐	☐		☐
Portability	☐	☐		☐
Application support	☐	☐		
Has a GUI	☐			
Usage		☐		☐
Number of unique connections				
Number of users				
Test deployment		☐		☐
Openness of design	☐	☐	☐	☐
Open source	☐	☐		
Sustainable network and development	☐	☐		☐
Developed by experts	☐	☐		

TABLE VII

GOALS AND METRICS NOT RELATED TO THE CENSOR BY APPROACH/PAPER. Metrics are listed under the goals for which papers have used them.

Stability of decoy hosts: Examines how long a decoy host is available to carry on a conversation.
Startup time: Measures how quickly client software starts up.
Sustainable network and development: Whether the system has funds and other resources to continue operating for the long term.
TLS characteristics: Prevents detection by TLS characteristics, like TLS nonce, clienthello or serverhello messages.
Test deployment: An approach proposed in an academic paper that is deployed in the real world and used by users.
Throughput: The amount of throughput/bandwidth the tool enables.
Time overhead: How much extra time it takes to use the tool.
Time to create an adaptation: The amount of time it takes some programmer to create a new adaptation of the protocol.
Total TCP connection: The total number of TCP connections per session does not stick out.
Total payload length: The total payload length produced by the tool does not stick out.
Usability: Assesses the additional effort that the circumvention tool client user must expand to use the system.
Usage: Assesses real world usage of an approach.
Use TLS for confidentiality: Whether an approach uses TLS to provide confidentiality.
Use TLS for integrity: Whether an approach uses TLS to provide integrity.
Use UDP with reliability: Whether an approach uses UDP with reliability.
Use a popular protocol: Whether an approach sends traffic using a popular protocol, such as the Skype protocol, to force the censor to either block a popular protocol or identify the circumventing usage of the protocol from normal usage.
Use authenticated key exchange (MITM): Whether an approach uses authenticated key exchange.
Use authentication: Whether a client needs authentication to connect to the server.
Use block cipher (key reuse): Whether an approach uses block cipher to resist key reuse attack.
Use certificate pinning (MITM): Whether an approach uses certificate pinning to avoid MiTM.
Use client puzzle (DoS): Require clients to solve a puzzle to prevent DoS.
Use encryption for confidentiality: Whether an approach uses encryption for confidentiality (and/or integrity).
Use encryption to resist traffic analysis: Whether an approach uses encryption to resist traffic analysis.
Use error correcting codes: Whether an approach uses error correcting codes.
Use many access points: Whether an approach uses too many hosts to make it hard for a censor to block all of them.
Use network infrastructure: Whether an approach uses infrastructure within a network, e.g., router, to avoid address blocking.
Use popular hosts: Whether an approach uses popular hosts, such as Skype nodes and CDNs, to resist address blocking.
Use random port: Use a random port number for communications.
Use shared secret (MITM): Whether an approach uses shared secret to resist man-in-the-middle attacks.
Use strong third-party service (DoS): A censor would have to overcome not just the circumvention deployment, but some third-party that hosts the deployment.
Use timestamp (replay): Whether an approach uses timestamp to resist replay attack.
Use trustworthy proxy: By using a trustworthy proxy as the forwarder, the approach avoids the risks of a malicious proxy (as long as the proxy remains trustworthy).
Veracity of claims: Evaluates whether the claims of about an approach by the its provider match reality.