# SoK: Making Sense of Censorship Resistance Systems

Sheharbano Khattak*, Tariq Elahi*, Laurent Simon, Colleen M. Swanson, Steven J. Murdoch, and Ian Goldberg

# SoK: Making Sense of Censorship Resistance Systems

**Abstract:** An increasing number of countries implement Internet censorship at different scales and for a variety of reasons. Several censorship resistance systems (CRSs) have emerged to help bypass such blocks. The diversity of the censor's attack landscape has led to an arms race, leading to a dramatic speed of evolution of CRSs. The inherent complexity of CRSs and the breadth of work in this area makes it hard to contextualize the censor's capabilities and censorship resistance strategies. To address these challenges, we conducted a comprehensive survey of CRSs—deployed tools as well as those discussed in academic literature—to systematize censorship resistance systems by their threat model and corresponding defenses. To this end, we first sketch a comprehensive attack model to set out the censor's capabilities, coupled with discussion on the scope of censorship, and the dynamics that influence the censor's decision. Next, we present an evaluation framework to systematize censorship resistance systems by their security, privacy, performance and deployability properties, and show how these systems map to the attack model. We do this for each of the functional phases that we identify for censorship resistance systems: *communication establishment*, which involves distribution and retrieval of information necessary for a client to join the censorship resistance system; and *conversation*, where actual exchange of information takes place. Our evaluation leads us to identify gaps in the literature, question the assumptions at play, and explore possible mitigations.

DOI 10.1515/popets-2016-0028
Received 2016-02-29; revised 2016-06-02; accepted 2016-06-02.

## 1 Introduction

The importance of online communication for political freedom, social change and commerce continues to grow; for example, recent history suggests that the events of the Arab Spring were in part spurred by the ability of revolutionaries to mobilize and organize the population through the use of social networking tools [1, 2]. Consequently, various actors (particularly nation states) have resorted to censor communications which are considered undesirable, with more than 60 countries around the world engaging in some form of censorship [3]. To evade such blocks, numerous *censorship resistance systems (CRSs)* have been developed with the aim of providing unfettered access to information, and to protect the integrity and availability of published documents. Due to the diversity of censorship mechanisms across different jurisdictions and their evolution over time, there is no one approach which is optimally efficient and resistant to all censors. Consequently, an arms race has developed resulting in the evolution of censorship resistance systems to have dramatically sped up. This has resulted in a lack of broad perspective on capabilities of the censor, approaches to censorship resistance, and the overarching trends and gaps in the field.

While a couple of surveys on this topic have emerged in recent years, these appear to trade-off between being comprehensive [4] and rigorous [5], or have different goals (e.g. analyzing how threat models employed in CRS evaluations relate to the behaviour of real censors as documented in field reports and bug tickets [6]). We systematize knowledge of censorship techniques and censorship resistance systems, with the goal to capture both the breadth and depth of this area. Our goal is to leverage this view to find common threads in CRSs, how to evaluate these, how do these relate to the censor's attack capabilities, and identify a list of research gaps that should be considered in future research. Our study is motivated by two observations. First, CRSs tend to employ *disparate threat models*, which makes it hard to assess the scope of circumvention offered by the system in isolation as well as in comparison with others. Secondly, CRSs have *diverse goals* which they meet by employing various metrics; it is difficult to discern how these metrics fit into broad CRS functionality, and to what effect.

We conduct a comprehensive survey of 73 censorship resistance systems, including deployed systems and those described in academic literature. We consolidate the threat models employed in these systems, to develop a single, comprehensive model of censor capabilities (Section 2). We augment this model with an abstract model of censorship that explains the extrinsic factors that may deter the censor from blocking even when it has the technical resources to do so. Turn-

***Corresponding Author: Sheharbano Khattak:** University of Cambridge, Sheharbano.Khattak@cl.cam.ac.uk
***Corresponding Author: Tariq Elahi:** KU Leuven, tariq@kuleuven.be
**Laurent Simon:** University of Cambridge, Laurent.Simon@cl.cam.ac.uk
**Colleen M. Swanson:** University of California, Davis, cmswan@ucdavis.edu
**Steven J. Murdoch:** University College London, s.murdoch@ucl.ac.uk
**Ian Goldberg:** University of Waterloo, iang@cs.uwaterloo.ca

ing to censorship resistance, we first define a censorship resistance system as full-blown software involving interaction between various components, and having two-step functionality (Section 3). Next from our survey of CRSs, we extract CRS schemes based on recurring themes and underlying concepts. We then establish a set of definitions to describe CRSs in terms of their security, privacy, performance and deployability properties (Section 4). Finally, we apply this framework to evaluate CRS schemes by their functional phases, and discuss strengths and limitations of various schemes (Sections 5 and 6). We conclude by laying out a set of overarching research gaps and challenges to inform future efforts (Section 7). To summarize our key contributions, we (*i*) present a comprehensive censorship attack model; (*ii*) provide a definition of a censorship resistance system in terms of its components and functionality; (*iii*) establish a CRS evaluation framework based on a common set of well-defined security, privacy, performance and deployability properties; (*iv*) identify broad schemes of censorship resistance that succinctly represent the underlying concepts; (*v*) conduct comparative evaluation of censorship resistance schemes; and (*vi*) discuss research gaps and challenges for the community to consider in future research endeavours.

# 2 Internet Censorship

A censorship resistance systems (CRS) is meant for information exchange between a user and a publisher that are connected by a communication channel in the face of a censor who actively attempts to prevent this communication. A censor's goal is to disrupt information exchange over a target system, by attacking the information itself (through corruption, insertion of false information, deletion or modification), or by impairing access to, or publication of, information (Figure 1).

## 2.1 Censorship Distinguishers

A censor's decision is aided by *distinguishers* which are composed of feature and value pairs. A *feature* is an attribute, such as an IP address, protocol signature, or packet size distribution, with an associated *value* that can be a singleton, list, or a range. In general, values are specified by a distribution on the set of all possible values that the feature can take. Where this distribution is sufficiently unique, feature-value pairs can be used as a distinguisher to detect prohibited activities. For example, a prevalence of 586-byte packet lengths forms a distinguisher the censor can utilize to identify Tor traffic.

Distinguishers are high- or low-quality depending on if they admit low or high error rates, respectively. Furthermore,

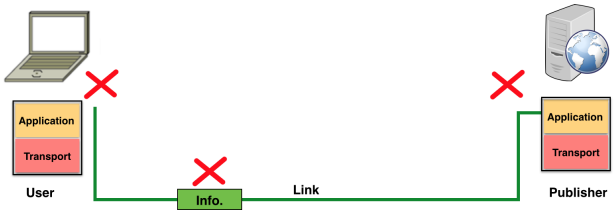


**Fig. 1.** A system meant for information exchange involves a *user* that retrieves *information* from a *publisher* over a link that connects the two. The censor's goal is to disrupt information exchange either by corrupting it, or by hindering its access or publication (indicated by red crosses in the figure).

they can be low- or high-cost depending on if small or large amounts of resources are required to utilize them. For the censor, high-quality, low-cost distinguishers are ideal. The primary source of distinguishers is network traffic between the user and publisher, where feature-value pairs may correspond to headers and payloads of protocols at various network layers, e.g. source and destination addresses in IP headers, or destination ports and sequence numbers in the TCP header, or TLS record content type in the TLS header. Packet payloads, if unencrypted, can also reveal forbidden keywords; the censor may act on such distinguishers without fear of collateral damage since there is little uncertainty about the labelling of these flows. Additionally, the censor can use the distinguishers discussed above to develop models of allowed traffic, and disrupt flows that deviate from normal. The censor can also collect traffic statistics to determine distinguishers based on flow characteristics, such as packet length and timing distributions.

## 2.2 Scope of Censorship

Censors vary widely with respect to their motivation, effectiveness, and technical sophistication. A wide range of entities, from individuals to corporations to state-level actors, may function as a censor. The extent to which a censor can effectively disrupt a system is a consequence of that censor's resources and constraints. In particular, the censor's technical resources, capabilities, and goals are informed by its *sphere of influence* and *sphere of visibility*. The sphere of influence is the degree of *active* control the censor has over the flow of information and behaviour of individuals and/or larger entities. The sphere of visibility is the degree of *passive* visibility a censor has over the flow of information on its own networks and those of other operators.

The spheres of influence and visibility are dictated by *physical*, *political*, or *economic* dynamics. Limitations due to geography are an example of physical constraints, while relevant legal doctrine, or international agreements and under-

standings, which may limit the types of behaviour in which a censor is legally allowed or willing to engage, are examples of political limitations. Economic constraints assume the censor operates within some specified budget which affects the technical sophistication and accuracy of the censorship apparatus it can field.

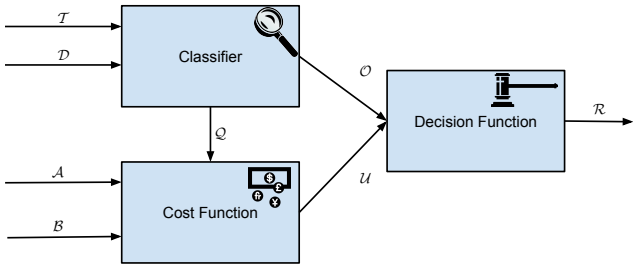


**Fig. 2.** Censorship apparatus model. The classifier takes as inputs the set of network traffic to be analyzed, $\mathcal{T}$, and the set of distinguishers, $\mathcal{D}$, and outputs a set $\mathcal{O}$ of traffic labels and the associated FNR and FPR, denoted by $\mathcal{Q}$. The cost function takes as inputs $\mathcal{Q}$, together with the censor's tolerance for collateral damage (FPR) and information leakage (FNR), denoted by $\mathcal{A}$ and $\mathcal{B}$, respectively, and outputs a utility function, $\mathcal{U}$. $\mathcal{B}$, and $\mathcal{Q}$. The decision function takes $\mathcal{O}$ and $\mathcal{U}$ as inputs and outputs a response $\mathcal{R}$.

## 2.3 An Abstract Model of Censorship

At an abstract level, the censorship apparatus is composed of classifier and cost functions that feed into a decision function (Figure 2). Censorship activity can be categorized into two distinct phases: *fingerprinting*, and *direct censorship*.

In the first phase (fingerprinting), the censor identifies and then uses a set of distinguishers $\mathcal{D}$ that can be used to flag prohibited network activity. For example, the censor may employ regular expressions to detect flows corresponding to a blocked publisher. The classifier takes $\mathcal{D}$ and the set of network traffic to be analyzed, $\mathcal{T}$, as inputs, and outputs offending traffic flows within some acceptable margin of error due to misclassification. The censor's error rates are the *information leakage*, or *false negative rate (FNR)*, which is the rate at which offending traffic is mislabeled as legitimate, and the *false positive rate (FPR)*, which is the rate at which legitimate traffic is mislabeled as offending, thus causing *collateral damage*. We assume that the censor wants to minimize the FNR and FPR within its given set of constraints.

In the second phase (direct censorship), the censor responds to flagged network flows, relying on a utility function that accounts for the censor's costs and tolerance for errors. For example, the censor may choose to block flagged network flows by sending TCP reset packets to both the user and publisher to force the connection to terminate.

## 2.4 Censor's Attack Model

We now outline the attack surface available to the censor (illustrated in Figure 3). This model expands the concepts of fingerprinting and direct censorship from the abstract model of censorship described in Section 2.3, grouping censorship activities by where these take place (user, publisher, or the link that connects these). Our attack model (and later systematization in Section 4) is based on survey of 73 censorship resistance systems, including deployed tools and academic papers: we consolidated the threat models sketched in all the surveyed systems to a single attack model described below.

### 2.4.1 Fingerprinting

**Fingerprint Destinations.** A flow can be associated with a protocol based on distinguishers derived from the connection tuple. Destination port is a typical target of censorship (e.g. 80 for HTTP). Flows addressed to IP addresses, hosts and domain names known to be associated with a blocked system can be disrupted by implication. Flow fingerprinting of this kind can form part of a multi-stage censorship policy, possibly followed by a blocking step. Clayton examines the hybrid two-stage censorship system CleanFeed deployed by British ISP, BT. In the first stage, it redirects suspicious traffic (based on destination IP and port) to an HTTP proxy. In the next stage, it performs content filtering on the redirected traffic and returns an error message if requested content is on the block list [7].

**Fingerprint Content.** Flows can be fingerprinted by checking for the presence of protocol-specific strings, blacklisted keywords, domain names and HTTP hosts. A number of DPI boxes can perform regex-based traffic classification [8–11], however it remains unclear what are the true costs of performing deep packet inspection (DPI) at scale [12, 13]. Alternatively, flows can be fingerprinted based on some property of the content being carried. For example, if the censor does not allow encrypted content, it can use high entropy as a distinguisher to flag encrypted flows [14].

**Fingerprint Flow Properties.** The censor can fingerprint a protocol by creating its statistical model based on flow-based distinguishers such as packet length, and timing-related features (inter-arrival times, burstiness etc.). Once a model has been derived, the censor can fingerprint flows based on their resemblance or deviation from this model [15, 16]. Wiley [17] used Bayesian models created from sample traffic

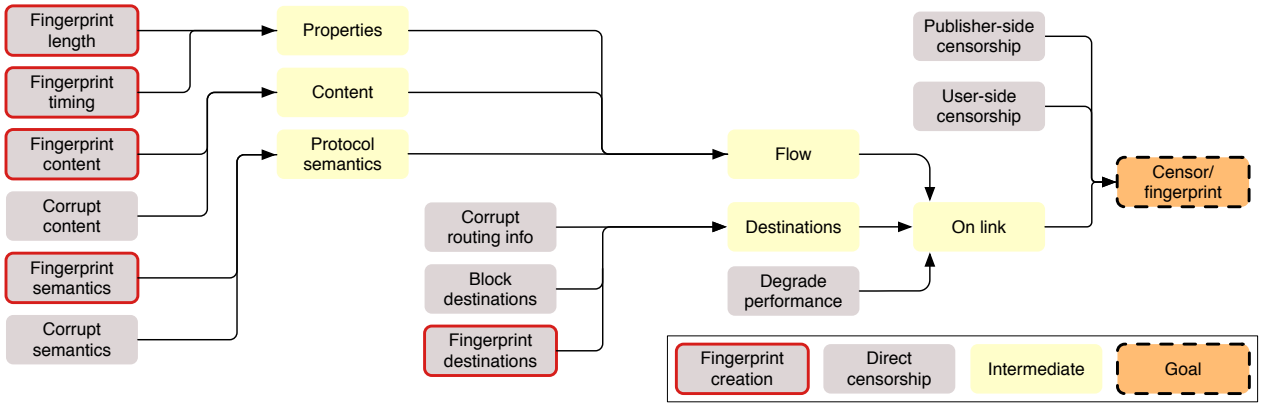


**Fig. 3.** Censor's attack model, showing both direct censorship (information corruption, or disabling access or publication) and fingerprinting (to develop and improve features for direct censorship). This figure does not include the relatively intangible attacks of coercion, denial of service, and installation of censorship software on machines.

to fingerprint obfuscated protocols (Dust [18], SSL and obfsopenssh [19]) based on flow features, and found that across these protocols length and timing detectors achieved accuracy of 16% and 89% respectively over entire packet streams, while the entropy detector was 94% accurate using only the first packet. A host's transport layer behavior (e.g. the number of outgoing connections) can be used for application classification. Flow properties can also be used to fingerprint the website a user is visiting even if the flow is encrypted [20–23].

**Fingerprint Protocol Semantics.** The censor can fingerprint flows based on protocol behaviour triggered through different kinds of active manipulation, i.e. by dropping, injecting, modifying and delaying packets. The censor's goal is to leverage knowledge of a protocol's semantic properties to elicit behaviour of a known protocol. Alternatively, a censor can perform several fingerprinting cycles to elicit the information on which to base subsequent blocking decision. In 2011, Wilde [24] investigated how China blocked Tor bridges and found that unpublished Tor bridges are first scanned and then blocked by the Great Firewall of China (GFW). Wilde's analysis showed that bridges were blocked in the following fashion: (*i*) When a Tor client within China connects to a Tor bridge or relay, GFW's DPI box flags the flow as a potential Tor flow, (*ii*) random Chinese IP addresses then connect to the bridge and try to establish a Tor connection; if it succeeds, the bridge IP/port combination is blocked.

### 2.4.2 Direct Censorship

**User Side Censorship.** The censor can directly or discretely (facilitated by malware or insider attacks) install *censorship software* on user's machine. This software can then disrupt user's access to information in various ways, for example by disallowing unapproved software installation, disrupting functionality of Internet searches by returning pruned results, and displaying warnings to the user to dissuade them from attempting to seek, distribute, or use censored content. This kind of censorship may lead to corruption of information as well as access disruption. China's Green Dam, a filtering software product purported to prevent children from harmful Internet content, was mandated to be installed on all new Chinese computers in 2009 [25]. The software was found to be far more intrusive than officially portrayed, blocking access to a large blacklist of websites in diverse categories, and monitored and disrupted operation of various programs if found to be engaging in censored activity. TOM-Skype, a joint venture between a Chinese telephony company TOM Online and Skype Limited, is a Voice-over-IP (VoIP) chat client program that uses a list of keywords to censor chat messages in either direction [26].

The censor can *coerce* users trying to access blocked content. The censor can set up malicious resources, such as proxy nodes or fraudulent documents that counteract the publisher's goals, to attract unwary users. For example, adversarial guard relays are known to exist on the Tor network and can be used to compromise Tor's client-destination unlinkability property. See Elahi *et al.* [27] and several follow-on works [28–30] for an in-depth analysis. China regularly regulates the online expression of its citizens, using an army of thousands of workers to monitor all forms of public communication and to identify dissidents [31], who may then be targeted for punishment [32].

**Publisher Side Censorship.** The censor can install *censorship software* on the publisher or employ a manual censorship process to corrupt the information being published or disrupt the publication process. A number of studies investigate Chinese government's censorship of posts on the na-

tional microblogging site Sina Weibo. Bamman *et al.* analyze three months of Weibo data and find that 16% of politically-driven content is deleted [33]. Zhu *et al.* note that Weibo's user-generated content is mainly removed during the hour following the post with about 30% of removals occurring within 30 minutes and about 90% within 24 hours [34]. Another study observes posts from politically active Weibo users over 44 days and finds that censorship varies across topics, with the highest deletion rate culminating at 82%. They further note the use of *morphs*–adapted variants of words to avoid keyword-based censorship. Weiboscope, a data collection, image aggregation and visualization tool, makes censored Sina Weibo posts by a set of Chinese microbloggers publicly available [35].

Within its sphere of influence the censor can use legal or extralegal *coercion* to shut down a publisher. History shows that this can been successful, e.g., a popular anonymous email service was pressured by the U.S. government to reveal users' private information [36]. The censor can coerce publishers, e.g. through threats of imprisonment, into retracting publications and otherwise chill their speech.

For destinations outside the censor's sphere of influence, the censor can mount a network attack, e.g., China launched an effective *distributed denial of service attack (DDoS)* against Github, to target censorship resistance content hosted there [37]. Other attacks based on infiltration of resource pools can cause similar denial of service.

**Degrade Performance.** The censor can manipulate characteristics of the link between the user and publisher, for example by introducing delays, low connection time-out values and other constraints. These techniques are especially useful if the censor does not have reliable distinguishers, and can be very effective if the target CRS traffic is brittle and not resilient to errors, but legitimate traffic is not similarly degraded. Compared to more drastic measures like severing network flows, degradation is a soft form of censorship that diminishes access to information while at the same time affording deniability to the censor.

Anderson uses a set of diagnostics data (such as network congestion, packet loss, latency) to study the use of throttling of Internet connectivity in Iran between January 2010 and 2013 [38]. He uncovers two extended periods with a 77% and 69% decrease in download throughput respectively; as well as eight to nine shorter periods. These often coincide with holidays, protest events, international political turmoils and important anniversaries, and are sometimes corroborated by overt filtering of online services or jamming of international broadcast television.

**Block Destinations.** Censorship can leverage distinguishers pertaining to a connection tuple (source IP address, source port, destination IP address and destination port), or hosts and domain names to disable information access or publication. The block can continue for a short period of time to create a chilling effect and encourage self-censorship on part of the users. One study notes that the Great Firewall of China (GFW) blocks communication from a user's IP address to a destination IP address and port combination for 90 seconds after observing 'objectionable' activity over that flow [39]. GFW has been reported to drop packets originating from Tor bridges based on both source IP address and source port to minimize collateral damage [40].

**Corrupt Routing Information.** The censor can disrupt access by corrupting information that helps in finding destinations on the Internet. This can be done by changing routing entries on an intermediate censor-controlled router, or by manipulating information that supports the routing process, e.g. BGP hijacking and DNS manipulation. Border Gateway Protocol (BGP) is the de facto protocol for inter-AS routing. The censor can block a network's connectivity to the Internet by withdrawing previously advertised network prefixes or re-advertising them with different properties (rogue BGP route advertisements). A number of countries have attempted to effect complete or partial Internet outages in recent years by withdrawing their networks in the Internet's global routing table (Egypt [41], Libya [42], Sudan [43], Myanmar [44]). DNS is another vital service that maps names given to different Internet resources to IP addresses. The hierarchical distributed nature of DNS makes it vulnerable to censorship. Typical forms of DNS manipulation involve redirecting DNS queries for blacklisted domain names to a censor-controlled IP address (DNS redirection or poisoning), a non-existent IP address (DNS blackholing) or by simply dropping DNS responses for blacklisted domains. China's injection of forged DNS responses to queries for blocked domain names is well known, and causes large scale collateral damage by applying the same censorship policy to outside traffic that traverses Chinese links [45].

**Corrupt Flow Content.** The censor can compromise information or disrupt access by corrupting transport layer payload of a flow. For example, a censor can inject HTTP 404 Not Found message in response to requests for censored content and drop the original response, or modify HTML page in the body of an HTTP response.

**Corrupt Protocol Semantics.** The censor can corrupt information or disrupt access by manipulating protocol semantics. A censor can leverage knowledge of protocol specification to induce disruption on a flow (for example, injecting forged TCP reset packets into a flow will cause both endpoints to tear down the connection).

Having gained understanding of various aspects of censorship, in the next section we turn to censorship resistance systems (CRSs). Note that there is not much that a CRS can

offer in terms of block evasion if the censor has installed malicious software on user or publisher machines. We therefore consider these attacks out-of-scope, and do not discuss these through the rest of this paper.

# 3 Censorship Resistance

A censorship resistance system (CRS) thwarts the censor's attempts to corrupt information, or its access or publication. The CRS achieves this goal by overcoming the censor's sphere of influence and sphere of visibility while maintaining an acceptable level of security and performance for its clients. CRS users include whistleblowers, content publishers, citizens wishing to organize, and many more, and any one CRS may be employed simultaneously for multiple *use cases*. Each use case may require different properties of the CRS. For example, a whistleblower needs to have their identity protected and be able to get their message to an outsider without raising suspicions that this is occurring; a content publisher wants that no amount of coercion, of themselves or the hosts of their content, could impact the availability of their content; and citizens wishing to organize would want unblockable access to their self-organization tool.

## 3.1 Censorship Resistance Systems: Components and Functionality

A censorship resistance system is full-blown software that involves interaction between various components to achieve unblockable communication between user and publisher (Figure 4). The user first installs the *CRS client* software on her machine which provides the desired CRS functionality. At a high level, we can break down CRS functionality into two distinct phases: *Communication Establishment* and *Conversation*.

**Communication Establishment.** Communication Establishment includes various steps that the CRS client performs starting from obtaining CRS credentials from the *dissemination server* to be able to access and use the *CRS server*. The goal here is that legitimate CRS users should be able to learn CRS credentials easily, while the censor should not be able to harvest efficiently. Additionally, the communication exchanges between the CRS client, and the dissemination and CRS servers should be resistant to fingerprinting.

As a preliminary step, the user may employ an *out-of-band link* to gather bootstrapping information (such as a secret key or token). This step is not mandatory and only required by some systems. The out-of-band link is used to communicate CRS-operational secrets that are assumed to somehow arrive with absolute security with respect to the censor. The key is that there is a limitation inherent to the link that does not allow it to be used for primary CRS communications, which would also obviate the need to use a CRS in the first place. An example of such a link is a person from outside the censor's sphere of visibility bringing addresses of CRS proxies on a USB stick through the airport and hand delivering it to the user.

Next the CRS client uses the dissemination link to connect to the dissemination server to retrieve information about the CRS. Some of this information may be public, such as domain name mappings to IP addresses, routing information, and other general Internet communications details. CRS-specific information is usually also required, including addressing information of proxy servers or locations of censored content. In some cases, CRS-specific information is restricted to prevent the censor from learning or tampering with it. *CRS credentials* are requested and served over the dissemination link, which can take many forms, such as ordinary email to a publicly available address, a lookup requested by the CRS client to a directory serving a file, or an anonymized encrypted connection to a hidden database. Next the CRS client uses the information previously obtained to connect to the CRS server over the user-to-CRS link. Some systems may skip the connection between CRS client and dissemination server, and directly connect to the CRS server. The connection between the CRS client and the dissemination server, and that between the CRS client and the CRS server, can optionally be facilitated by various intermediate *participants*. Usually these are proxies that relay traffic between the CRS client, and the dissemination and CRS servers.

**Conversation.** In the Conversation phase, the CRS client has successfully connected to the CRS server and is ready to exchange information. The goal here is to frustrate the censor's attempts to block the user-to-CRS link or tamper with the information being carried, or fingerprint distinguishers to subsequently use for blocking. The CRS client connects to the CRS server, typically outside the censor's sphere of influence, which in turn connects to the publisher. The CRS server acts like a proxy that sits between the user and publisher, relaying traffic back and forth over an unblockable channel. The communication between the CRS client and CRS server can optionally be supported by CRS facilitator(s). The publisher is the covert page, person, or platform to which the CRS client ultimately wants access. Examples include a dissident blog (page), a video call with a friend (person), and tweeting about the location of the next anti-government rally on Twitter (platform). Alternatively, the publisher could also be the stepping stone to another blocked system. For example, a number of countries block Tor by blocking access to its entry relays (bridge nodes). Users in these countries often employ CRS to establish an unblockable connection with the bridge nodes,

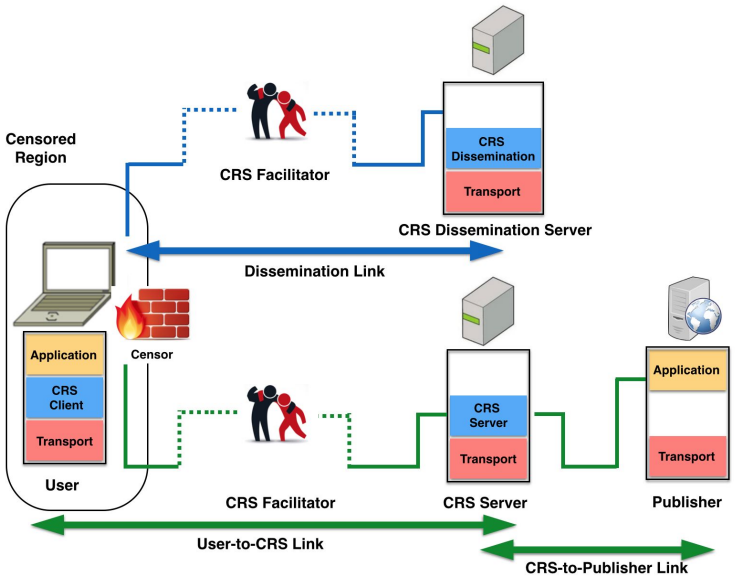


**Fig. 4.** The Censorship Resistance System (CRS) provides users unfettered access to information despite censorship. It comprises of *CRS client* software installed on the user which obtains information (CRS credentials) about how to access the *CRS server* from *dissemination server* over the *dissemination link*. The CRS client then uses these credentials to connect to the CRS server over the *user-to-CRS link*. Optionally there could be one or more *CRS facilitators* on the dissemination link and the user-to-CRS link which support the CRS's operation. The CRS server connects to the publisher over the *CRS-to-publisher link*, effectively acting as a proxy that enables unblockable communication between the user and publisher. It is common for CRSs to handle the CRS-to-publisher link separately than the other two links due to different security and performance properties

and subsequently bootstrap into the Tor network. In this example, the bridge node is the publisher from the perspective of the CRS client.

It is common for CRSs to treat the dissemination link and the user-to-CRS link separately than the CRS-to-publisher link as these lend themselves to different design, implementation, and software distribution practices. Most CRSs provide circumvention on the user-to-CRS link due to its flashpoint status in the censorship arms race: censorship on this link is less intrusive and more convenient for the censor as most of the communication infrastructure is within its sphere of influence, and fingerprinting is more effective as bulk exchange of information takes place on this link. In contrast, being typically outside the censor's sphere of influence, the CRS-to-publisher link may simply provide access to the publisher, without offering any additional security properties and could be simply implemented as a HTTP or SOCKS proxy, or as a VPN. Alternatively, the CRS-to-publisher link may connect to an anonymity system like Tor [46] to offer privacy properties in addition to unblockable access to information.

**Example.** To tie things together, we provide the example of a CRS, ScrambleSuit [47], illustrating the phases of Communication Establishment and Conversation, and the interaction between various components. In the Communication Establishment phase in ScrambleSuit, the CRS client retrieves a short-lived ticket from the CRS over a low-bandwidth out-of-band channel. The CRS client redeems the ticket from the dissemination server for mutual authentication. After successful authentication, the dissemination server gives the client a ticket for the next connection, so that the CRS client does not have to use the out-of-band channel to retrieve tickets for subsequent connections. The CRS client includes this ticket in its connection request to the CRS server. The CRS server only responds to a connection request if it contains a valid ticket, thus obfuscating its existence from the censor that probes (and subsequently fingerprints) IP addresses that run the CRS server. In the Conversation phase, the CRS client and CRS server transform their traffic into random-looking bytes to thwart content-based fingerprinting, and remove flow fingerprints by randomizing packet lengths and flow timing.

# 4 Systematization Methodology

We now describe our methodology to evaluate censorship resistance systems. We conducted a comprehensive survey of CRSs–totalling 73 systems–both deployed and proposed in academic publications until February 2016. We selected academic publications that appeared in well-respected venues in security research, and for deployed tools we turned to references of the surveyed academic papers, and results of search on Google Scholar using relevant keywords. We include the full list of CRSs surveyed in Appendix A.

Our evaluation of CRSs spans four dimensions: security, privacy, performance and deployability. We break down each of these dimensions into specific properties. As a large number of CRSs have emerged over the years, we base our evaluation on *schemes* representing recurring themes and underlying concepts in the censorship resistance landscape, instead of individual systems. We believe that evaluating CRSs by common schemes rather than individual systems is a useful abstraction to visualize their strengths, limitations and opportunities. Next we select a representative CRS from the group of all systems that employ the given strategy, and evaluate it along the four dimensions of security, privacy, performance and deployability. We note that our selection of the representative CRS is based on how well it captures the given strategy; a CRS may well employ multiple strategies in a layered design. It is possible that some properties of the representative CRS do not exclusively represent a given scheme; we indicate in our discussion where this is the case.

We apply the evaluation methodology sketched above to each of the two phases of CRS functionality, Communication Establishment and Conversation. While most properties along the dimensions of security, privacy, performance and deployability are common across both Communication Establishment and Conversation, we state where this is not the case. Our methodology may be described as a semi-structured approach to evaluation of censorship resistance systems. Most properties have binary values (*has*, *does not have*), while some also have an intermediate value (*partially has*).

The evaluation process involved all six authors, each acting as a domain expert, and included the following steps. (*i*) *Survey and categorization:* First, two groups of two authors studied all the 73 CRSs identified. This exercise generated two lists of common schemes for censorship resistance across all the CRSs surveyed, and a representative system for each of these schemes. The two lists were consolidated into one and then refined through discussion amongst the four contributing authors. (*ii*) *Developing evaluation framework:* Two authors developed the framework to evaluate CRSs, which was iteratively refined through discussion with the other two authors. (*iii*) *Evaluation:* Once the evaluation framework, CRS strategies and representative CRSs had been identified, one author evaluated all the representative systems, and then divided all the evaluated CRSs into three sets (where each set contained systems corresponding to a diverse mix of CRS strategies) and invited the other three authors to independently verify the evaluation of these systems. Differences in evaluation were resolved through iterative discussions and incorporated in the evaluation. (*iv*) Finally, to ensure inter-rater reliability the other two, thus far uninvolved, authors verified the soundness of the evaluation process and scores assigned.

Another important dimension of CRS evaluation alongside security, privacy, performance and deployability, is usability. In this paper, however, we restrict our scope to the first four of these and defer usability to future work. Usability evaluation of CRSs is not well understood, and forms an emerging research area.

## 4.1 Security Properties

Censorship resistance systems incorporate a number of security properties; to prevent or slow the censor from learning high-quality distinguishers; to make it hard for the censor to distinguish CRS traffic from other, allowed network flows; and to ensure CRS availability even in the case of successful detection e.g. by taking advantage of the censor's resource limitations by raising the overall cost of processing network traffic.

**Unobservability.** The censor cannot detect prohibited communication or the use of CRS itself based on content or flow-based signatures, or destinations associated with the CRS or those known to serve prohibited content. The following are three common techniques.

- **Content Obfuscation:** Communication exchanges between the CRS client and other CRS components do not contain unique static strings or string patterns that the censor can associate with prohibited material or the CRS itself.
- **Flow Obfuscation:** Communication exchanges between the CRS client and other CRS components do not contain unique stochastic patterns (such as packet sizes and timing) that the censor can associate with prohibited material or the CRS itself.
- **Destination Obfuscation:** The identity and network location (e.g. IP addresses, domain names and hosts) of the dissemination server, CRS server, and key facilitators is hidden.

**Unblockability.** The censor, even after having identified prohibited content or usage of the CRS, is either unable or unwilling to block communications, often due to collateral damage. Two common techniques follow:

- **Outside Censor's Influence:** This involves placing critical CRS components susceptible to attack beyond the censor's sphere of influence (and sometimes sphere of visibility). In this case, even if the censor can identify the CRS component as a target, the censor may still be unable to launch an effective attack.
- **Increase Censor's Cost to Block:** The censor's blocking decision depends on a number of factors, such as the accuracy of distinguishers, and blocking mechanism employed. In particular, the censor's policy must make con-

sideration for the acceptable false positive rate as these have political and economic ramifications [48]. Mechanisms under this category obfuscate CRS credentials or infrastructure in such a way that blocking it incurs unacceptable collateral damage.

**Availability.** Does the CRS incorporate mitigation against Denial of Service attacks on its components, either directly or by leveraging a DoS-resistant participant? In general, there is a lack of robust techniques to completely neutralize DoS attacks, with general mitigation strategies involving over-provisioning of bandwidth and IP addresses.

**Communication Integrity.** Access to information and the information itself is robust in the presence of a censor that tampers with information, or actively manipulates channel properties, such as injecting, modifying or dropping packets, or changing routing assumptions.

- **Message Integrity:** Does the CRS verify integrity of data in the presence of an active censor that can tamper with it while in transit or stored on the publisher?
  **Server/Publisher Authentication.** Does the CRS incorporate authentication of the dissemination and CRS servers (Communication Establishment phase), and the publisher (Conversation phase).
- **Packet Loss Support:** Is the CRS resilient against packet drops induced by the censor? (While packet drops can impact Communication Establishment as well as Conversation, we consider it only in the latter case where the threat is more pressing due to the volume of information exchanged).
- **Out-of-Order Packets Support:** Can the CRS handle packets that arrive out-of-order due to the censor injecting random delays to traffic? (Applies to Conversation phase only).

## 4.2 Privacy Properties

This includes anonymity and deniability for CRS components to mitigate coercion by the censor. In a CRS that provides coercion resistance to its components, the censor's threats of force are of limited impact, such as when the CRS component is outside the censor's sphere of influence, or the system is designed in a way that makes it technically impossible to comply with the censor's demands.

**User Anonymity.** Can the CRS client anonymously retrieve information from the dissemination server (Communication Establishment) and the publisher (Conversation)? Here the threat is from a censor that can enumerate (and subsequently coerce) users by observing connections to the dissemination server and the publisher.

**Server/Publisher Anonymity.** Can the dissemination server (Communication Establishment) and publisher (Conversation) disseminate information to users without revealing their identity? Here the threat is from a censor that identifies (and subsequently coerces) servers and publishers of information by observing where CRS clients connect to (for example, by masquerading as a CRS client), or where prohibited information is fetched from (for example, through passive analysis of data on the wire).

**User Deniability.** This means that the censor cannot reasonably confirm if the user intentionally accessed prohibited information or used the CRS, and therefore cannot implicate users for lack of sufficient evidence. User deniability can be enforced by transforming traffic to an obfuscated form when it leaves user machine, for example through encryption or steganography.

**Server/Publisher Deniability.** This means that the censor cannot reasonably confirm if the dissemination server or publisher intentionally served prohibited information or participated in the CRS, and therefore lacks sufficient evidence to implicate the two. This can be enforced by obfuscating responses of the dissemination server and publisher, for example through encryption or steganography.

**Participant Deniability.** Parties that support the CRS in its operations should be able to deny intentional participation in the CRS, or helping disseminate prohibited information, thus preventing the censor from blocking them or meting out punishment.

## 4.3 Performance Properties

All censorship resistance systems have certain performance characteristics: in some cases these are the side effect of the CRS scheme employed with little room for improvement, while in other cases these are due to inadequate design choices of the representative system and can be ameliorated.

**Latency.** For Communication Establishment, it means how long does it take from when the CRS client initiates Communication Establishment to when it is ready to start Conversation, compared to the baseline approach of directly connecting to the CRS server. In the context of Conversation, latency is the delay introduced by the CRS, compared to the baseline approach of downloading a document over a standard protocol like HTTP without employing the CRS. A number of factors can contribute to CRS communications latency including artificial inter-arrival times between packets, packet padding and the additional CRS protocol header.

**Goodput.** This refers to the useful bandwidth available for the information originally requested by the CRS client, and amounts to the total bandwidth minus the CRS's operational

overhead. A high-goodput CRS implies that the CRS is high-throughput and has low operational overhead resulting in high amounts of bandwidth available for the CRS client to retrieve information. A low-goodput CRS means that either the CRS is low-throughput to begin with or that the CRS is high-throughput but has high overhead resulting in little bandwidth for information requested by the CRS client. (Applies to Conversation phase only).

**Stability.** Do the performance characteristics of the CRS, such as communication latency and goodput, remain consistent? For each scheme, we identify factors (e.g. reliance on external conditions beyond CRS control), and the degree to which these can cause performance to fluctuate, and rate accordingly.

**Scalability.** How well does the CRS scale as the number of CRS clients that want to communicate with the dissemination server, or access information using the CRS server, increases? For each scheme, we identify challenges and the degree to which these can affect scalability, and rate accordingly.

**Computational Overhead.** This refers to the degree of additional computational resources incurred by the dissemination server (compared to the baseline approach of directly connecting to the CRS server) and the CRS server (compared to the baseline approach of directly downloading a web page without employing the CRS) due to CRS usage. In Communication Establishment, this is typically proportional to the cryptographic overhead introduced by the authentication scheme in use by the dissemination server.

**Storage Overhead.** This refers to the degree of additional storage required by the dissemination server and the CRS server due to CRS usage compared to the baselines described for computational overhead.

### 4.4 Deployability Properties

The utility of a CRS depends not only on its security, privacy and performance properties, but also on how amenable it is to be deployed and used in the real world. This depends on a number of factors ranging from cost to implement and maintain the CRS, to the degree of cooperation expected from participants.

**Synchronicity.** Does the system require all components relevant to the Communication Establishment and Conversation phase to be online at the same time?

**Network Agnostic.** Do the Communication Establishment and Conversation phases make specific network layer assumptions (for example, specific routing requirements, or assumptions about packet fragmentation and TTL values) to function correctly and as intended?

**Coverage.** This refers to the degree of Internet access (such as fraction of the Web, and Internet services and protocols) enabled by the CRS. We rate a CRS to have high coverage if it can be used to access any content, medium coverage if there are restrictions on content type or length, and low coverage if both content type and length are restricted. (Applies to Conversation phase only).

**Participation.** A large number of CRSs depend on cooperation from participants. We use various metrics to assess the quality and flexibility of cooperation the CRS requires from participants.

- **Quantitative Incentivization:** Is the incentive structure to encourage participants to help the Conversation based on tangible rewards, monetary or in kind (in contrast to qualitative incentives, such as goodwill and public relations)?
- **Distributed Participation:** This represents the degree to which participation is diffused among cooperative entities: a low value corresponds to a single organized entity, such as a corporation, a company, or an institution; medium value represents multiple organized bodies; while a high value indicates individual volunteers participating in personal capacity. This property has implications for CRS stability and scalability if participants opt out of the CRS due to policy changes.
- **Voluntary Participation:** Is the participant aware that the CRS has employed it for censorship resistance, or are they unwitting?
- **Conditional Participation:** Is the participation premised on specific conditions, such as popularity, reputation and location (in contrast to open participation)?
- **Deterministic Cost:** Does the CRS provide estimated cost to participants, or alternatively allow them to control the degree of participation in the CRS?
- **Security Delegation:** Are the security properties offered by the CRS compromised due to subverted participant(s)?
- **Privacy Delegation:** Are the privacy properties offered by the CRS compromised due to subverted participant(s)?

## 5 Communication Establishment

We describe schemes employed by CRSs to enable the CRS client to obtain CRS credentials, while preventing the censor from detecting and blocking Communication Establishment, or harvesting and fingerprinting CRS credentials.

### 5.1 High Churn Access

This scheme relies on the selection of CRS credentials that change regularly so that these cannot be preemptively blocked

by the censor. Since the censor has yet to discover distinguisher values, there is a window of opportunity where misclassification can occur.

## 5.2 Rate-Limited Access

To prevent the censor from harvesting CRS credentials as a legitimate CRS participant, the CRS may limit the rate at which CRS credentials can be queried. This is usually done by requiring proofs of work from participants, partitioning the value space over time slices and based on CRS participant attribute(s), or including reputation-based mechanisms.

**Proof of Work.** To prevent the censor from harvesting CRS information at scale, some CRSs employ proof of work approaches, such as captchas and other puzzles. The assumption is that these are too expensive for wide-scale harvesting by a resource-bounded censor.

**Time Partitioning.** This involves partitioning CRS credentials over time slices, e.g. by choosing a large pool of unpredictable values and then using and retiring them within a short time frame. As a result, the censor needs to continuously allocate resources to keep up-to-date with CRS values.

**Keyspace Partitioning.** This refers to partitioning CRS information over attribute(s) specific to users, e.g. client IP address. This way, a user only learns a restricted set of CRS credentials over the value space.

## 5.3 Active Probing Resistance Schemes

A censor may probe suspected dissemination and CRS servers to confirm if they participate in the CRS. To mitigate this threat, the CRS introduces a sequence of steps during the Communication Establishment phase; these steps are feasible for a single CRS user to follow, but hard for a censor to perform at scale. The servers can obfuscate their association with the CRS from an unauthenticated user by pretending to be offline altogether (*obfuscating aliveness*), or by providing an innocuous response or no response at all CRS (*obfuscating service*).

**Obfuscating Aliveness.** A dissemination server may not respond to connection requests from a CRS client until they complete an expected sequence of steps, e.g. by requiring clients to embed a secret token in the request, or encoding a valid request through a series of packets to a number of ports in a specific order.

**Obfuscating Service.** This differs from the previous case in that the dissemination server responds to connection requests from a CRS client (thus revealing its aliveness), but refuses to speak the CRS protocol until the CRS client completes an expected sequence of steps.

## 5.4 Trust-based Access

To only service requests from legitimate CRS clients, the server may associate an element of trust with CRS users based on previous behaviour, and the server only responds to requests from CRS clients with satisfactory reputation. In some systems, the reputation of CRS users is derived from their social network graph.

## 5.5 Discussion

In Table 1, we present our evaluation of censorship resistance schemes related to the Communication Establishment phase of CRS functionality. The rows correspond to the security, privacy, performance and deployability properties described in Section 4. The columns represent our evaluation of CRS schemes (using a representative CRS) along these properties; we state in our discussion where properties of the representative CRS do not exclusively represent a given scheme. The last row of the table provides a breakdown of the deployment status of all the CRSs for a scheme (a complete list of citations for these systems can be found in Table 3 in Appendix A). Most of the efforts in the area of censorship resistance have concentrated on Conversation: we found only 11 systems for Communication Establishment (compared to 62 for Conversation), of which about half have tools available. Tschantz *et al.* note that the research focus on Conversation is orthogonal to real censorship attacks that concentrate on Communication Establishment [6]. We now present the results from our evaluation, highlighting common trends and discussing the strengths and limitations of the CRS schemes. We emphasize that the effectiveness of a given CRS depends on the context in which it has been employed; the goal of our evaluation is to characterize the suitability of CRS schemes to different use cases and censor capabilities.

We observe that schemes where the server responds after validating users (active probing resistance, proof of work) typically offer *content obfuscation*, e.g. by ensuring that messages containing puzzles, tokens or similar credentials do not have content-based signatures. A naive active probing resistance system that includes a fixed-length token in the request, is vulnerable to *flow fingerprinting*, as the censor can detect connections that always begin by sending a fixed number of bytes. Such length-based signatures can be removed by using pseudo-random padding [47]. With respect to performance, the user validation step in these schemes comes at the cost of increased *time to join the system*, higher *computational needs*, possibly coupled with additional *storage requirements* if the server stores identifiers for static matching instead of verifying identifiers at run-time. As disparate interactions are required

**Table 1.** Evaluation of censorship resistance schemes related to Communication Establishment phase of CRS functionality. Notation for binary values: ✓ has property, ✗ does not have property. Notation for non-binary values: ● has property, ◒ partially has property, ○ does not have property. – means the property does not apply to the given scheme. The last row provides a breakdown of deployment status of *all* the systems surveyed for the given scheme; a full list of the corresponding citations is provided in Appendix A.

| | | High Churn Access | Rate Limited | | | Active Probing | | Trust-Based Access |
|---|---|---|---|---|---|---|---|---|
| | | | Proof of Life/Work | Time Partitioning | Keyspace Partitioning | Obfuscating Aliveness | Obfuscating Service | |
| | **System** | **Flashproxy [49]** | **Defiance [50]** | **Tor Bridges [51]** | **Keyspace-Hopping [52]** | **SilentKnock [53]** | **ScrambleSuit [47]** | **Proximax [54]** |
| **Security** | **Unobservability** | | | | | | | |
| | Content Obfuscation | ✗ | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ |
| | Flow Obfuscation | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ | ✗ |
| | Destination Obfuscation | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| | **Unblockability** | | | | | | | |
| | Outside Censor Influence | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ |
| | Increase Censor Cost to Block | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| | **Availability** | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| | **Communication Integrity** | | | | | | | |
| | Server Authentication | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ | ✗ |
| | Message Integrity | ✗ | ✓ | ✓ | ✗ | ✓ | ✓ | ✗ |
| **Privacy** | User Anonymity | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ |
| | Publisher/Server Anonymity | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ |
| | User Deniability | ✗ | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ |
| | Publisher/Server Deniability | ✗ | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ |
| | Participant Deniability | ✗ | ✓ | ✓ | ✓ | – | – | ✗ |
| **Performance** | Low Latency | ● | ○ | ● | ● | ◒ | ◒ | ● |
| | Stability | ○ | ○ | ● | ◒ | ● | ● | ○ |
| | Scalability | ○ | ○ | ○ | ○ | ● | ◒ | ○ |
| | Low Computation | ● | ○ | ● | ● | ◒ | ◒ | ● |
| | Low Storage Overhead | ● | ◒ | ● | ● | ● | ● | ● |
| **Deployability** | **Synchronicity** | ◒ | ◒ | ● | ● | ● | ● | ◒ |
| | **Network Agnosticism** | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| | **Participation** | | | | | | | |
| | Quantitative Incentivization | ✗ | ✗ | ✗ | ✗ | – | – | ✗ |
| | Distributed Participation | ● | ● | ● | ● | – | – | ● |
| | Voluntary Participation | ✓ | ✓ | ✓ | ✓ | – | – | ✓ |
| | Conditional Participation | ✗ | ✗ | ✓ | ✗ | – | – | ✓ |
| | Deterministic Cost | ✓ | ✗ | ✓ | ✗ | – | – | – |
| | Security Delegation | ✓ | ✓ | ✓ | ✓ | – | – | ✓ |
| | Privacy Delegation | ✓ | ✓ | ✓ | ✓ | – | – | ✓ |
| **Status** | **Academic & Tool available** | **2** | **0** | **0** | **0** | **2** | **1** | **0** |
| | **Academic paper only** | **0** | **2** | **0** | **1** | **0** | **0** | **1** |
| | **Tool only** | **0** | **0** | **1** | **0** | **0** | **0** | **1** |

between the CRS client and different CRS servers and participants, these can be at least partially conducted *asynchronously*, usually within some time window. Schemes that obfuscate service need more consideration with respect to *load balancing*; for each request a TCP connection is still established, since validation takes place at the application layer.

Most trust-based and high churn access schemes do not *obfuscate content* and such functionality must be built on top of the scheme. The same is true of *flow obfuscation*: communication establishment involves just few brief interactions while statistical classifiers, especially those based on inter-packet arrival times, work best on high volume data. Schemes that involve straight forward access to CRS servers with some prudence on part of the server to hand out values with discretion (e.g. high churn access, time partitioning, keyspace hopping, proxy and trust-based schemes) tend to have low *latency*. Typically there is a single interaction between the CRS client and the CRS server directly or through intermediate forwarding participants, consequently the client, server and all participants need to be *online at the same time*.

A particular consideration for high churn access systems is that new values must constantly be available. However, if these values are opportunistically derived from volunteer participants, the system is neither *stable* nor *scalable*; it is not clear how long a value will remain available, and how many values will be available as the number of users increases. A possible mitigation is to have a notion of participant quality (for example, in terms of available bandwidth and uptime) so that requests are more intelligently distributed under increased demand (*conditional participation*). We note that conditional participation enables the CRS to adequately plan issues concerning system scalability and stability. In a broader context, we note that most participant-based schemes recruit individual entities who volunteer to help based on *qualitative incentivization*, usually goodwill. All these schemes fail to offer privacy and security if one or more participants are controlled by the censor. Any participant can potentially compromise security and privacy properties of the CRS: in such an event, decentralized systems fare better in terms of damage control.

A shortcoming of rate-limited access and active probing resistance schemes is that the censor can sometimes deploy more resources than anticipated (the Sybil attack). The censor can deploy or control a large enough pool of resources to harvest the CRS value space to effectively neutralize the benefits of these schemes (e.g. to neutralize proof of work schemes, the censor can invest in computational power to solve multiple puzzles in parallel [55]). Perhaps this explains why proof of work schemes only exist in literature.

In high churn access the CRS value space harvested by the censor is quickly outdated; effectively, distinguishers harvested by the censor are unstable and inconsistent, thus necessitating frequent updates. A challenge for high churn access schemes is how to manage the value space so that new values are constantly available, especially when the value space is contingent on volunteers whom the CRS cannot directly control. Trust-based schemes are vulnerable to malicious participants. To gain access to a CRS employing trust-based access, the censor must impersonate as a credible user, for example by stealing credentials of an existing user, or earning good reputation over a period of time, both of which do not scale well. Moreover, most schemes track CRS user reputation even after initial authorization, and malicious behaviour can lead to ejection. However, few subverted CRS users can potentially deplete and block the entire value space. This can be mitigated by rate-limiting the information available per CRS user.

Most schemes do not incorporate *authentication* of dissemination and CRS servers. This is a problem as the censor can masquerade as a server and enumerate and coerce users. To mitigate this threat, the CRS can maintain a centralized authority which the CRS client can query independently to establish authenticity of a server (e.g. Tor directory services [56]). Alternatively, the censor can masquerade as a server and disrupt CRS accessibility by distributing bogus CRS credentials. This threat is particularly relevant when the CRS design includes volunteer servers which cannot be authenticated. Possible mitigation includes for the CRS to digitally sign the information distributed by servers, so that the CRS users can verify *integrity of information* obtained from untrusted servers (e.g. Defiance NET payloads). For the same reason, most active probing schemes offer message integrity so that the censor cannot tamper with the information that validates the CRS user to the server. On the other hand, we note that most schemes (high churn access, keyspace hopping, trust-based access) that manage control to unauthenticated proxies do not offer message integrity, and this has to be handled by another application on top of it.

With respect to *privacy*, none of the schemes offer *user anonymity*; if the censor successfully fingerprints credentials of CRS servers, it can identify users that connect to these. However, it may prove difficult to implicate users at this point because the user has only attempted to use the CRS, but not actually used it to access blocked content. The same applies to *server anonymity*; the censor can masquerade as a legitimate CRS user and identify the servers that the CRS client contacts. Most schemes offer *user and server deniability* by using encryption and/or steganography, which somewhat alleviates the lack of anonymity. Trust-based schemes are an exception; these erect stringent criteria for CRS participation but do not adequately address user deniability if one or more members have been subverted. Schemes that rely on volunteer participants have an obvious incentive to offer *participant deniability*. Some schemes waive this property and instead as-

sume that, being outside the censor's sphere of influence, the participant will be immune to coercion attempts.

# 6 Conversation

In this section, we describe schemes employed by CRSs to evade detection and blocking in the Conversation phase. We note that the identified schemes enable unobservable and unblockable access to information (access-centric) and/or may incorporate additional measures to store information for increased availability and coercion resistance (publication-centric). We observe this distinction in our discussion for clarity, however, we note that it is possible for a CRS to employ schemes from both groups.

## 6.1 Access-Centric Schemes

Schemes in this category protect access to information over the user-to-CRS link (Figure 4) by safeguarding security and privacy of relevant CRS components, and by protecting information in transit from corruption.

**Mimicry.** This scheme transforms traffic to look like whitelisted communication, such that the transformed traffic resembles the syntax or content of an allowed protocol. Mimicry can be of known protocol, or randomness, and can be at the level of content as well as flow.

*Content Mimicry* evades censorship by imitating the syntax of an innocuous protocol (e.g. HTTP) or content (e.g. HTML). Typically the mimicry is based on widely-deployed protocols or popular content thus complicating a censor's task by (*i*) increasing the censor's work load as there is more volume of traffic to inspect, and (*ii*) increasing the collateral damage associated with wholesale protocol blocking. Another approach is to make traffic content look like an unknown protocol, either by imitating randomness or arbitrarily deviating from a known blocked one. This idea is motivated by the general assumption that the censor implements blacklisting of known protocols and is unwilling to incur high collateral damage associated with whitelisting.

*Flow Mimicry* is similar to content mimicry except that the CRS imitates flow-level characteristics (e.g. packet length and inter-arrival timings) of an unblocked protocol, or randomizes flow-level characteristics to remove statistical fingerprints.

**Tunnelling.** In contrast to mimicry where the CRS pretends to be an unblocked protocol and may manage to only partially mimic the target protocol, in this scheme CRS traffic is tunnelled through an unblocked application thus avoiding the problems inherent to mimicry.

**Covert Channel.** involves hiding CRS communication in a cover medium, creating a covert channel within the cover which transmits CRS traffic in a manner not part of its original design (e.g. hiding HTTP requests within a cover image). As a result, CRS traffic is not only hard to detect, but also deniable for both users and publishers. Steganographic techniques are useful here.

**Traffic Manipulation.** This scheme exploits the limitations of the Censor's traffic analysis model and shapes CRS traffic such that the censor is unable to fingerprint it. Effectively, this scheme renders even cleartext traffic unobservable which is particularly useful for countries that prohibit encrypted traffic.

**Destination Obfuscation.** To prevent the censor from observing and blocking the key destinations the CRS client directly connects to, the client can instead relay CRS traffic through one or more intermediate nodes to obfuscate CRS destinations.

The CRS may employ a *proxy* as CRS facilitator to relay traffic between the CRS client and the CRS server. Usually, the proxies change frequently to avoid getting blocked. The CRS traffic can possibly be relayed through a number of proxies for improved privacy.

In *decoy routing*, clients covertly signal a cooperating intermediate router to deflect their traffic purportedly en route to an unblocked destination (to evade the censor) to a blocked one. The deflecting routers must be located on the forward network path from the client to the unblocked destination.

## 6.2 Publication-Centric Schemes

Schemes in this category protect information over the CRS-to-publisher link (Figure 4). These systems allow publishers to push information to the CRS which is stored among multiple CRS servers, and served to CRS users upon request. Consequently, the main goal of publication-centric schemes is to ensure availability and integrity of information, while offering deniability to CRS servers, and preferably to CRS users and publishers too. It is common for these schemes to refer to published information as *documents*, hence we interchangeably refer to information as documents.

**Content Redundancy.** refers to storing content redundantly on a large number of CRS servers, typically placed in different jurisdictions making it hard for the censor to remove prohibited content from all the servers.

**Distributed Content Storage.** breaks the content into smaller chunks and distributes these among a number of CRS servers so that no server has the full document. To reconstruct the document, the corresponding chunks are retrieved from the CRS servers where these are stored.

**Table 2.** Evaluation of censorship resistance schemes related to Conversation phase of CRS functionality. Notation for binary values: ✓ has property, ✗ does not have property. Notation for non-binary values: ● has property, ◒ partially has property, ○ does not have property. – means the property does not apply to the given scheme. The last row provides a breakdown of deployment status of *all* the systems surveyed for the given scheme; a full list of the corresponding citations is provided in Appendix A.

| | | Access-Centric Schemes | | | | | | Publication-Centric Schemes | |
|---|---|---|---|---|---|---|---|---|---|
| | | Content/Flow Obfuscation | | | | Destination Obfuscation | | Content Redundancy | Distributed Storage |
| | | Mimicry | Tunnelling | Covert Channel | Traffic Manip. | Proxy | Decoy Routing | | |
| | System | SkypeMorph [57] | Freewave [58] | Collage [59] | Khattak *et al.* [60] | Tor [46] | Cirripede [61] | Freenet [62] | Tangler [63] |
| Security | **Unobservability** | | | | | | | | |
| | Content Obfuscation | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ | ✗ |
| | Flow Obfuscation | ✓ | ✗ | – | ✗ | ✗ | ✓ | ✗ | ✗ |
| | Destination Obfuscation | ✗ | ✓ | ✓ | ✗ | ✓ | ✓ | ✓ | ✓ |
| | **Unblockability** | | | | | | | | |
| | Outside Censor Influence | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✗ | ✓ |
| | Increase Censor Cost to Block | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| | **Availability** | ✗ | ✗ | ✓ | ✗ | ✗ | ✗ | ✓ | ✓ |
| | **Communication Integrity** | | | | | | | | |
| | Publisher Authentication | ✓ | ✗ | ✓ | ✗ | ✗ | ✗ | ✓ | ✓ |
| | Message Integrity | ✓ | ✗ | ✗ | ✗ | ✓ | ✓ | ✓ | ✓ |
| | Packet Drop Resistance | ✗ | ✗ | – | ✗ | ✗ | ✓ | – | – |
| | Out-of-Order Resistance | ✓ | ✓ | – | ✗ | ✓ | ✓ | – | – |
| Privacy | User Anonymity | ✗ | ✓ | ✓ | ✗ | ✓ | ✓ | ✓ | ✓ |
| | Publisher Anonymity | ✗ | ✓ | ✓ | ✗ | ✓ | ✓ | ✓ | ✓ |
| | User Deniability | ✓ | ✓ | ✓ | ✗ | ✓ | ✓ | ✗ | ✓ |
| | Publisher Deniability | ✓ | ✗ | ✓ | ✗ | ✗ | ✗ | ✗ | ✓ |
| | Participant Deniability | ✓ | ✓ | ✓ | – | ✓ | ✓ | ✗ | ✓ |
| Performance | Low Latency | ◒ | ◒ | ○ | ◒ | ◒ | ● | ● | ◒ |
| | High Goodput | ◒ | ◒ | ○ | ● | ● | ● | – | – |
| | Stability | ● | ● | ○ | ◒ | ● | ● | ◒ | ● |
| | Scalability | ◒ | ● | ○ | ○ | ◒ | ◒ | ◒ | ◒ |
| | Low Computation | ◒ | ◒ | ◒ | ● | ● | ◒ | ● | ◒ |
| | Low Storage Overhead | ● | ● | ○ | ● | ● | ● | ● | ● |
| Deployability | **Synchronicity** | ✓ | ✓ | ✗ | ✓ | ✓ | ✓ | ✗ | ✗ |
| | **Network Agnosticism** | ✓ | ✓ | – | ✗ | ✓ | ✗ | ✓ | ✓ |
| | **Coverage** | ● | ● | ○ | ● | ● | ● | ◒ | ◒ |
| | **Participation** | | | | | | | | |
| | Quantitative Incentivization | ✗ | ✗ | ✗ | – | ✗ | ✗ | ✗ | ✓ |
| | Distributed Participation | ○ | ◒ | ◒ | – | ● | ◒ | ● | ● |
| | Voluntary Participation | ✗ | ✗ | ✗ | – | ✓ | ✓ | ✓ | ✓ |
| | Conditional Participation | ✓ | ✓ | ✓ | – | ✓ | ✗ | ✓ | ✓ |
| | Deterministic Cost | ✗ | ✗ | ✗ | – | ✓ | ✗ | ✗ | ✓ |
| | Security Delegation | ✓ | ✓ | ✓ | – | ✓ | ✓ | ✓ | ✓ |
| | Privacy Delegation | ✓ | ✓ | ✓ | – | ✓ | ✓ | ✓ | ✓ |
| Status | **Academic & Tool available** | 2 | 0 | 1 | 2 | 4 | 0 | 1 | 0 |
| | **Academic paper only** | 8 | 7 | 7 | 2 | 4 | 5 | 4 | 1 |
| | **Tool only** | 5 | 2 | 0 | 1 | 10 | 0 | 0 | 0 |

## 6.3 Discussion

In Table 2, we present our evaluation of censorship resistance schemes related to the Conversation phase of CRS functionality. The rows correspond to the security, privacy, performance and deployability properties described in Section 4; and the columns represent evaluation of CRS schemes identified above, along these properties. As our evaluation is based on representative systems for various CRS schemes, we state in our discussion where properties of the representative CRS do not exclusively represent a given scheme. The last row of the table shows a breakdown of the deployment status of all the CRSs for a given type of scheme (a complete list of citations for these systems can be found in Table 4 in Appendix A). We surveyed 62 systems, of which 28 have end user tools available—most of which are mimicry and proxy-based schemes. We now discuss common trends, as well as strengths and limitations of the CRS schemes based on our evaluation. Our goal is to characterize the suitability of the CRS schemes to different use cases and censor capabilities.

We observe that nearly all access-centric schemes offer *content obfuscation*, but only mimicry schemes *obfuscate traffic flows*. Some mimicry schemes morph traffic to resemble the content or flow-level characteristics of a cover protocol. This approach is inherently imperfect as cover protocols are generally complex, with disparities stemming from incomplete or incorrect cover protocol imitation, such as failure to handle errors in a consistent manner. The censor can leverage such disparities to identify evasive traffic through active manipulation [64–66]. Other schemes morph traffic such that its content and flow-based features look random. Effectively the censor, being unable to classify such traffic as any known protocol, cannot block. However, if the censor only allows whitelisted protocols then it can flag random looking traffic as anomalous and block it. Though most mimicry-based systems offer user deniability, they typically lack in destination obfuscation and most privacy properties.

Tunnelling ameliorates the limitations of mimicking cover protocols to some degree (such as *destination obfuscation*, and resistance to *dropped packets*), however, the censor may be able to take advantage of inconsistencies in channel usage or content. The CRS traffic may still have flow-based features which distinguish it from the cover protocol [64]. Further, the CRS may rely on channel characteristics in a different manner from the cover protocol; if the cover protocol is more robust to network degradation, for example, the censor can manipulate the network to disrupt CRS traffic while not affecting legitimate cover protocol traffic [67]. Tunnelling systems typically leverage popular third-party platforms to increase *collateral damage* associated with blocking the CRS. As such platforms are provisioned for Internet-scale performance, tunnelling-based systems also inherit high *availability* and *scalability*. Both mimicry and tunnelling schemes incur additional protocol overhead, resulting in decreased *goodput*, and additional *latency* in extracting CRS traffic from the cover (e.g. demodulating voice data received over a cover VoIP application to recover tunnelled information). Despite its strengths, tunnelling schemes have largely received attention in academic literature, with only two tools available.

In our survey, proxy-based schemes have the highest number of end user tools available. Both proxy-based schemes and decoy routing relay traffic through intermediate participants. In the former case, traffic can be redirected through multiple proxies, which can lead to increase in *latency*. A crucial component of the decoy routing strategy is that by building circumvention into the Internet infrastructure, the need for communication establishment is obviated: the CRS client includes the credentials needed to join the CRS using covert channel inside a request for an unblocked overt destination, which is intercepted by a decoy router on path to the overt destination and deflected to a proxy that facilitates communication with a blocked destination. In addition to *destination obfuscation*, the use of TLS leverages *content obfuscation*, good *performance* and resistance to *active manipulation* (we find in our evaluation that most decoy routing systems are resilient against attacks involving dropped packets). However, the requirement for decoy routers to be deployed in cooperative ISPs is based on the assumption that it is impossible for the censor to "route around" cooperating ISPs without significant *collateral damage*: if this assumption is invalid, however, the censor can avoid or otherwise blackhole the route and nullify this scheme [68]. We note that none of these systems have been deployed; the requirement to be supported by real-world ISPs poses a significant deployment challenge, and *scalability* is dependent on the number and location of supporting ISPs as decoy routers should be deployed widely enough to be on path to a large number of overt destinations.

Traffic manipulation schemes use low level network tricks to cause the censor to misinterpret traffic flows and effectively *fail to detect blocked content*. As traffic manipulation schemes make certain assumptions about censorship apparatus, it is vulnerable to attacks involving *out-of-order packets*. It is hard to offer a good degree of *performance* if the censor's policy changes frequently. For the same reason, this scheme is not *scalable*: the CRS has to be tuned and updated according to individual censorship policies.

Covert channel-based systems have been popular in academic literature, but there is only one such tool available. This scheme provides *unobservable communication*, while maintaining a high degree of *privacy* for all CRS components. Like tunnelling-based systems, the robustness of the scheme depends on how closely the CRS traffic blends into content of

the cover medium, and conforms to semantics of the cover protocol. Some cover media are vulnerable to specific attacks: timing-based covert channels are sensitive to *dropped and out of order packets*; while covert channels built into header values of network protocols (e.g. timestamp, initial sequence numbers, padding values and different flags) can attract attention for being anomalous due to abnormal usage, or the covert channel can be destroyed if the censor normalizes fields that applications typically do not use. Attaining good *performance* is a challenge for covert channel-based schemes; as CRS traffic has to be encoded inside cover traffic, the amount of information that the cover traffic can carry (*goodput*) is limited. Therefore, the *coverage* of these schemes is typically restricted to static short messages. Some recent systems [69, 70] that utilize online games as the cover application manage to achieve lower *latency*, but the issue with low goodput remains, and additional processing to extract CRS traffic from the cover medium adds to latency. Some systems might have high *storage* requirements if the server has to maintain a collection of cover media to embed CRS traffic.

There exist only six publication-centric systems, which appeared in early 2000s, with only two cases where a tool is available; focus has shifted to access-centric schemes, with properties traditionally associated with secure publication being offered by content delivery networks. Publication-centric schemes have the goal to increase *availability* of the information they store, while offering publishers and participants *deniability*. Existing approaches to protect documents from removal include replication of documents across multiple participants, typically individual volunteers with *participation* potentially *conditioned* on the bandwidth or storage they are willing to offer. Another approach to increase document availability is to create dependence between multiple documents by intertwining them with each other. In the former case, the censor has to invest additional resources to remove a document, while in the latter case removing a document results in collateral damage as other intertwined documents are also deleted. While most publication-centric schemes provide *publisher authentication, document integrity and privacy properties*, a major limitation is that the protocol messages are observable which makes it possible for the censor to disrupt the CRS. By design, publication-centric schemes tend to be *asynchronous*, and provide partial *coverage*; allowing access to static documents, typically with no restriction on size. *Stability* of these schemes depends on the time it takes to find requested information.

# 7 Open Areas and Research Challenges

We provided a comparative evaluation of existing censorship resistance approaches, to characterize their suitability to different censorship models (security and privacy properties), use cases (performance properties), and deployability scenarios. In a broader context, this evaluation has highlighted a number of open areas and challenges, as described below.

## 7.1 Modular System Design

Our evaluation of various CRS schemes (Sections 5.5 and 6.3) suggests that it may be possible to increase coverage of desirable security and privacy properties by combining complementary CRSs. In many situations having several schemes in operation is more effective than using any one [71] (for example, Tor dealt with TLS handshake fingerprinting of January and September 2011 by modifying its protocol to protect against the vulnerable attack path).

However, this is not straight forward as most censorship resistance systems have been designed to be stand-alone systems with tightly integrated functionality. Even designs that share a common base, e.g. the Tor network and the various *pluggable transports* (CRS systems that can interface with Tor in a plug-and-play fashion [72]), suffer from this problem. Although the Tor community is actively trying to address this problem with the pluggable transport framework [73], the desired level of modularity and composability within pluggable transports themselves are not currently in place [5, 74]. Jumpbox alleviates, but does not solve, the problem at the network interface layer, by providing a standard interface for encapsulating CRS traffic to look like regular web traffic [75].

There has been some effort in the context of Tor to combine pluggable transports, however, this has happened not in a black-box way, but through the sharing of source code. LibFTE is in use by Tor (in its fteproxy Pluggable Transport form) and a number of other projects [76]. Similarly, Meek [77] which was originally developed for Tor now also exists in a fork by Psiphon [78] with minor adaptations. Fog uses multiple proxies to chain Pluggable Transports in a black box fashion [79]. This approach is not suitable for practical deployment due to a number of limitations. For example, not all combinations of pluggable transports make sense: the chain obfs3 [80] (flow fingerprinting resistance) followed by Flashproxy [49] (IP address filtering resistance) offers more comprehensive resistance, but the reverse, i.e. Flashproxy followed by obfs3 breaks the former's network layer assumptions. Khat-

tak *et al.* provide a coarse framework to build access-centric CRSs out of reusable components [5].

Another challenge is that CRS designs are typically forced to make a trade-off between security and performance. Combining CRSs to create a hybrid is likely to come at a performance cost. For example, a CRS that tunnels traffic through a popular content provider can be combined with a CRS that redirects traffic through multiple proxies to provide anonymity may provide greater security and privacy, but performance will be poor due to the tunnelling protocol's network overhead, combined with multi-hop routing employed by the second CRS. However, there might be cases where the hybrid's performance profile remains the same, such as the combination of a publication-centric CRS (e.g. Tangler [63]) with an access-centric scheme that uses covert channel to store CRS content on popular platforms (e.g. Collage [59]): the CRS user can use covert channel to send its request to the CRS server, which can then serve the requested information the usual way. Internally, the CRS can replicate information over multiple CRS servers for high availability and allow publishers to deniably post information to the CRS. This kind of hybrid is effective for use cases that must already allow for performance compromises, while prioritizing security properties.

## 7.2 Revisiting Common Assumptions

Our study of the literature (Tables 1 and 2) suggests that nearly all CRSs assume that increasing cost to block (collateral damage) and placing servers and facilitators outside the censor's sphere of influence will lead to unblockability. However, as we note below, these assumptions do not always hold.

While existing CRS schemes are contingent on and try to maximize collateral damage incurred by the censor in terms of false positives, they do not evaluate the impact of false negatives on censor's behaviour, nor identify parameters that might affect it. Filling this gap is an important next step in illuminating the dynamics of the censorship resistance game and providing feedback on best practices for CRS design. Indeed, we are beginning to see activity in this vein in recent work by Tschantz *et al.* [48] and Elahi *et al.* [81]. The latter specifically pursues the last two areas of research, by applying game-theoretic analysis to censorship resistance and investigating the impact of information leakage, collateral damage, and accuracy of the censorship apparatus on the censor's behaviour.

Another common CRS strategy is to resist the censor's fingerprinting efforts by mitigating perceived low-cost distinguishers. However, the capabilities and willingness of the censor to engage in sophisticated traffic analysis is unclear, and the cost of detecting complex high-cost distinguishers is not well understood. While it is difficult to find out the true capabilities of a censor, it is still useful to assess the cost to censor in employing lower- and higher-hanging distinguishers. While the literature does analyze detection attacks, these analyses are usually implementation specific, and as such the results are of limited scope and do not tell us if they are applicable to relate to a realistic censor. A recent study notes that the threat models employed by most existing CRSs are disconnected from how real censors operate, e.g. real censors tend to avoid attacks involving packet dropping to avoid the collateral damage of blocking allowed connections [6]. Such studies help explain why mimicry and tunnelling-based schemes continue to be effective in practice, despite theoretical attacks that demonstrate their vulnerability to active manipulation [64, 65] (Section 6). Similarly, a number of systems protect against a censor capable of fingerprinting flow properties and content; evaluation of such systems will greatly benefit from a repository of traffic capture files. Systems that perform traffic shaping use various protocols in a 'correct' manner and require labelled datasets against which to validate their schemes. Adversary Lab [82] has done some preliminary work on developing a standard environment to evaluate systems resistant to flow fingerprinting by subjecting them to a range of adversaries. There is a need for Internet-scale studies of distinguisher effectiveness, which may provide clues about which distinguishers are of the highest utility to the censor and the biggest threats to the CRS.

All CRSs rely on some CRS components being located outside the censor's sphere of influence. However, recent revelations, in particular those by Edward Snowden [83], call these design choices into question. The alarming reach of "Five Eyes"—a program of cooperation and surveillance data sharing between the governments of Australia, Canada, New Zealand, the United Kingdom, and the United States—necessitates a reevaluation of the basic assumptions most CRSs make with respect to the censor's sphere of influence and visibility. Systems are typically not designed to withstand global passive adversaries, for instance, and designs often count on distribution across diverse jurisdictions to make this assumption realistic. Given that a significant number of government entities cooperate and have far-reaching network capabilities, systems designed with these assumptions may be more brittle than previously believed. While it is unclear just how large the spheres of visibility and influence are for any given censor, their reach is likely far larger than anticipated by current CRS designs.

## 7.3 Security Gaps

Our analysis in Sections 5.5 and 6.3 reveals certain gaps in the security provided by CRS schemes. First, there are no ef-

fective countermeasures against the censor poisoning CRS information (*corrupt routing information* in Section 2.4). Any public information used by both CRS and non-CRS activities, however, has at least an implicit level of defense, since it is not without risk for the censor to poison such information.

Second, most CRSs are not completely immune to denial-of-service attacks on key CRS components and resources [84]. The CRS implicitly depends on capabilities of participant(s) or the network connectivity provider hosting dissemination and CRS servers to prevent such attacks. However, this is a broader security issue; denial-of-service attacks do not yet have a robust solution, outside of over-provisioning of bandwidth and IP addresses.

Third, corrupted content and malicious CRS participants are not adequately defended against and is an area of active research [28, 64]. At present, the primary work to address these problems is in the Tor ecosystem. The poisoning of the public relay information, such as IP addresses and port numbers, is mitigated using digital signature schemes, but protecting the authenticity and confidentiality (from the censor) of bridge addresses is an outstanding problem. There have been many denial-of-service attacks, both theoretical and actual, on the Tor network; these have been addressed on a case-by-case basis through programmatic changes. Finally, the Tor network actively attempts to detect suspicious relays through various network-level tests, but this is done in an ad-hoc fashion.

## 7.4 Considerations for Participation

We note that most schemes in Communication Establishment (Table 1), and some schemes in Conversation (Table 2) recruit individual entities who volunteer to help based on *qualitative incentivization*, usually goodwill. We note that most schemes treat participation as a binary decision, where participants lack the ability to control the degree of cooperation (e.g. bandwidth, number of connections, or users). Indeed, in most schemes it is not even clear what is the cost of participation. These issues may act as a deterrent to wide-scale adoption. A possible mitigation is to enforce a lower threshold on participation and give participants the flexibility to switch between higher values and the threshold. For example, some systems, such as Tor [46] and Tangler [63], demand a minimum amount of bandwidth or storage from participants, which they can configure to higher values if desired.

Our study highlights an increasing trend among CRSs to leverage existing, popular commercial services as participants; this is particularly true of mimicry, tunnelling and covert channel-based schemes which represent 32 of the total 73 CRSs surveyed. (Table 2). We note that such implementations meet CRS design goals through happy, and possibly temporary, coincidence. As a result, there are extrinsic properties we must account for when evaluating such CRSs, which may be undesirable even in the absence of the censor's active interference. First, such a design has dependencies on external parties who may not be invested in the CRS's success. Second, the properties for which the CRS leverages the commercial party do not exist by its own design, implying that there may be unexpected states that render the CRS ineffective, or the desired functionality may simply disappear with an update or for commercial reasons. For example, Skype used to have super nodes (special nodes with high bandwidth and stability, that relayed traffic between Skype users), but these were phased out in 2013 for scalability reasons. This affected mimicry-based CRSs (e.g.SkypeMorph [57]) that leveraged Skype's super nodes to obfuscate IP addresses of CRS users, thereby providing privacy.

A primary motivation for CRSs to entwine their activities with popular commercial parties is the assumption that the censor is unable to accurately extract CRS activity from the commercial party, and will refrain from wholesale blocking of the commercial party due to high collateral damage. Effectively the CRS causes the commercial party to act as a concentration point for maximizing collateral damage incurred by the censor. Counterintuitively, the censor may actually benefit from this concentration, since the attack surfaces and CRS security failures are also concentrated and well defined, e.g. CloudTransport [85] uses only traffic to Amazon's cloud storage and hence potentially easier to contain.

A related issue is that the CRS can be rendered ineffective if the censor develops local alternatives for commercial parties leveraged by the CRS and forces users in its sphere of influence to use them. Finally, commercial participants are often operated by single corporations; the censor can strike back at the CRS by attacking the platform and/or the entity that operates it. The strike can be in the form of actual network-level attacks [86] that cause the operator to reconsider or decry [87] its role as a host to CRS activity. Indeed, there is uncertainty around exactly how the operating entity may respond. In the worst case it may even act as an informant to the censor and monitor CRS activity.

# 8 Related Work

The literature contains a number of surveys, taxonomies, and reports that analyze censorship resistance, their varying goals, scopes, and technical depth. While the present study extends previous work in many ways, we believe its value lies in capturing breadth of the entire field of censorship resistance, while also providing sufficient technical details.

Elahi and Goldberg sketch the censor's attack model, and present a taxonomy of censorship resistance strategies for different types of censors (e.g. ISP, government) with the decision-making process based on their resources, capabilities, limitations, and utility [4]. Our work extends this work by providing more technical depth and rigorous systematization methodology.

Khattak *et al.* focus exclusively on pluggable transports [72], a framework to allow access-centric CRSs to flexibly plug into a larger system like Tor [5]. They represent functionality of a CRS that protects the link between CRS client and CRS server as a layered stack, and discuss threats and mitigations relevant to each layer. Their model of pluggable transports mirrors our own in that they are also concerned with disruption of access to information; however, our scope includes the entire CRS landscape including publishers and CRS participants, and issues surrounding communication establishment.

In a more recent work, Tschantz *et al.* conduct an extensive survey of evaluation criteria used by CRSs, and compare these to the behaviour of real censors as reported in field reports and popular bug tickets [6]. They find a significant disconnect between the threat models employed in theoretical evaluations, and how censors operate in practice. While their enumeration of criteria used by CRSs has some overlap with the security, privacy, performance, and deployability properties we use in our systematization, the study has different goals to ours: we evaluate underlying CRS approaches, while they exhaustively enumerate evaluation criteria employed by CRSs to identify trends, and how these relate to real world censors.

The above work is most closely related to our study. We now discuss other studies that are generally relevant to ours. Köpsell and Hillig present a classification of blocking techniques based on the communication layer involved (TCP/IP), the content of communication (e.g. images, web) and metadata of the communication (e.g. IP addresses of participants, time of the communication, protocols involved) [88]. Leberknight *et al.* survey the social, political, and technical aspects that underpin censorship. They also propose metrics that quantify their efficacy, i.e. scale, cost, and granularity [89]. Perng *et al.* classify circumvention systems based on the technical primitives and principles they build upon [90]. Tschantz *et al.* argue that the evaluation of circumvention tools should be based on economic models of censorship [48]. Gardner presents a non-technical document on freedom-supporting technologies, describing their maintenance and funding ecosystem, user demographics, their impact and the factors governing their development and usage [91].

# 9 Conclusion

As the importance of the Internet to effectively engage in civil society continues to grow, so does the desire of various actors to control the flow of information, and censor communications which are considered undesirable. A number of censorship resistance systems (CRSs) have emerged to help bypass such blocks. The interplay between these two actors with contrasting goals has given rise to an arms race, as a result of which the field has grown increasingly complex.

In this work we presented a comprehensive censor's attack model and established evaluation framework for measuring security, privacy, performance and deployability of CRSs. We provided a comparative evaluation of existing censorship resistance approaches, while emphasizing their strengths and limitations. In a broader context, our study has highlighted a number of open areas and challenges: *(i)* combining CRSs may increase coverage of desirable security and privacy properties, but doing so in a meaningful way without breaking other operational assumptions is challenging; *(ii)* common CRS assumptions regarding the censor's sphere of influence, and blocking cost to censor should be reevaluated in the light of evolving sociopolitical dynamics; *(iii)* there are outstanding security issues that CRSs cannot yet effectively mitigate, such as denial of service attacks, corruption of information that supports CRS (e.g. DNS poisoning), and malicious CRS participants; *(iv)* recruitment of volunteer participants should be at least partially regularized for better system stability and scalability; furthermore, the growing trend for CRSs to rely on powerful, commercial participants to increase collateral damage associated with blocking can potentially act as a single point of failure from a security and privacy standpoint. We hope that our work on contextualizing the area of censorship resistance and the research gaps identified will inform future research endeavours.

**Acknowledgement:** We appreciate the valuable feedback we received from the anonymous reviewers. Steven J. Murdoch and Sheharbano Khattak were supported by The Royal Society [grant number UF110392]; Engineering and Physical Sciences Research Council [grant number EP/L003406/1]. Tariq Elahi was supported by NSERC, Research Council KU Leuven: C16/15/058, and the European Commission through H2020-DS-2014-653497 PANORAMIX. Laurent Simon was supported by Samsung SERI, grant RG67002 and Thales e-security, grant NRG0EFKM. Colleen Swanson was supported by NSF CNS 1228828 and NSF CNS 1314885. Ian Goldberg thanks NSERC for grant RGPIN-341529.

# References


[1] R. M. Bond, C. J. Fariss, J. J. Jones, A. D. Kramer, C. Marlow, J. E. Settle, and J. H. Fowler, "A 61-Million-Person Experiment in Social Influence and Political Mobilization," *Nature*, vol. 489, no. 7415, pp. 295–298, 2012.

[2] N. Eltantawy and J. Wiest, "The Arab Spring | Social Media in the Egyptian Revolution: Reconsidering Resource Mobilization Theory," *International Journal of Communication*, vol. 5, no. 0, 2011.

[3] Freedom House, "Freedom in the World 2016." https://freedomhouse.org/report/freedom-world/freedom-world-2016, 2016.

[4] T. Elahi, C. M. Swanson, and I. Goldberg, "Slipping Past the Cordon: A Systematization of Internet Censorship Resistance." CACR Tech Report 2015-10, Aug. 2015.

[5] S. Khattak, L. Simon, and S. J. Murdoch, "Systemization of Pluggable Transports for Censorship Resistance." http://arxiv.org/pdf/1412.7448v1.pdf, 2014.

[6] M. C. Tschantz, S. Afroz, D. Fifield, and V. Paxson, "SoK: Towards Grounding Censorship Circumvention in Empiricism," in *Proceedings of the IEEE Symposium on Security and Privacy*, 2016.

[7] R. Clayton, "Failures in a Hybrid Content Blocking System," in *Privacy Enhancing Technologies*, (Cambridge, England), pp. 78–92, Springer, 2006.

[8] I7-filter, "http://research.dyn.com/2013/08/myanmar-internet/."

[9] Bro, "https://www.bro.org."

[10] Snort, "https://www.snort.org."

[11] nDPI, "http://www.ntop.org/products/ndpi/."

[12] A. Dainotti, A. Pescape, and K. Claffy, "Issues and Future Directions in Traffic Classification," *IEEE Network*, vol. 26, pp. 35–40, Jan 2012.

[13] R. Sommer and V. Paxson, "Outside the Closed World: On Using Machine Learning For Network Intrusion Detection," in *In Proceedings of the IEEE Symposium on Security and Privacy*, 2010.

[14] P. Dorfinger, G. Panholzer, and W. John, "Entropy Estimation for Real-Time Encrypted Traffic Identification," in *Traffic Monitoring and Analysis*, vol. 6613 of *Lecture Notes in Computer Science*, pp. 164–171, Springer Berlin Heidelberg, 2011.

[15] L. Bernaille and R. Teixeira, "Early Recognition of Encrypted Applications," in *Proceedings of the 8th International Conference on Passive and Active Network Measurement*, PAM'07, (Berlin, Heidelberg), pp. 165–175, Springer-Verlag, 2007.

[16] C. V. Wright, F. Monrose, and G. M. Masson, "On Inferring Application Protocol Behaviors in Encrypted Network Traffic," *J. Mach. Learn. Res.*, vol. 7, pp. 2745–2769, Dec. 2006.

[17] B. Wiley, "Blocking-Resistant Protocol Classification Using Bayesian Model Selection," tech. rep., University of Texas at Austin, 2011.

[18] B. Wiley, "Dust: A Blocking-Resistant Internet Transport Protocol," tech. rep., School of Information, University of Texas at Austin, 2011.

[19] B. Leidl, "obfuscated-openssh." https://github.com/brl/obfuscated-openssh/blob/master/README.obfuscation, 2009.

[20] A. Panchenko, L. Niessen, A. Zinnen, and T. Engel, "Website Fingerprinting in Onion Routing Based Anonymization Networks," in *Proceedings of the Workshop on Privacy in the Electronic Society (WPES)*, ACM, Oct. 2011.

[21] Q. Sun, D. R. Simon, Y.-M. Wang, W. Russell, V. N. Padmanabhan, and L. Qiu, "Statistical Identification of Encrypted Web Browsing Traffic," in *Proceedings of the IEEE Symposium on Security and Privacy*, May 2002.

[22] A. Hintz, "Fingerprinting Websites Using Traffic Analysis," in *Proceedings of Privacy Enhancing Technologies workshop (PET)*, Springer-Verlag, LNCS 2482, Apr. 2002.

[23] G. D. Bissias, M. Liberatore, and B. N. Levine, "Privacy Vulnerabilities in Encrypted HTTP Streams," in *Proceedings of Privacy Enhancing Technologies workshop (PET)*, pp. 1–11, May 2005.

[24] T. Wilde, "Great Firewall Tor Probing." https://gist.github.com/da3c7a9af01d74cd7de7, Nov. 2015.

[25] ONI, "China's Green Dam: The Implications of Government Control Encroaching on the Home PC." https://opennet.net/chinas-green-dam-the-implications-government-control-encroaching-home-pc.

[26] J. Knockel, J. R. Crandall, and J. Saia, "Three Researchers, Five Conjectures: An Empirical Analysis of TOM-Skype Censorship and Surveillance," in *Free and Open Communications on the Internet*, (San Francisco, CA, USA), USENIX, 2011.

[27] T. Elahi, K. Bauer, M. AlSabah, R. Dingledine, and I. Goldberg, "Changing of the Guards: A Framework for Understanding and Improving Entry Guard Selection in Tor," in *Proceedings of the ACM Workshop on Privacy in the Electronic Society*, pp. 43–54, ACM, 2012.

[28] A. Biryukov, I. Pustogarov, and R.-P. Weinmann, "Trawling for Tor Hidden Services: Detection, Measurement, Deanonymization," in *Proceedings of the IEEE Symposium on Security and Privacy*, May 2013.

[29] A. Johnson, C. Wacek, R. Jansen, M. Sherr, and P. Syverson, "Users Get Routed: Traffic Correlation on Tor by Realistic Adversaries," in *Proceedings of the 20th ACM conference on Computer and Communications Security*, Nov. 2013.

[30] R. Dingledine, N. Hopper, G. Kadianakis, and N. Mathewson, "One Fast Guard for Life (or 9 Months)," in *7th Workshop on Hot Topics in Privacy Enhancing Technologies*, 2014.

[31] T. Zhu, D. Phipps, A. Pridgen, J. R. Crandall, and D. S. Wallach, "The Velocity of Censorship: High-Fidelity Detection of Microblog Post Deletions," in *Proceedings of the 22nd USENIX Security Symposium*, USENIX, 2013.

[32] R. Rife, "Opinion: The Chilling Reality of China's Cyberwar on Free Speech." *CNN (U.S. Edition)*, http://www.cnn.com/2015/03/24/opinions/china-internet-dissent-roseann-rife/, Mar. 2015.

[33] D. Bamman and B. O'Connor and N. Smith, "Censorship and deletion practices in Chinese social media." http://journals.uic.edu/ojs/index.php/fm/article/view/3943/3169, Nov. 2015.

[34] T. Zhu, D. Phipps, A. Pridgen, J. R. Crandall, and D. S. Wallach, "The Velocity of Censorship: High-fidelity Detection of Microblog Post Deletions," in *Proceedings of the 22Nd*

*USENIX Conference on Security*, (Berkeley, CA, USA), pp. 227–240, USENIX Association, 2013.

[35] Journalism and Media Studies Centre, "Weiboscope." http://weiboscope.jmsc.hku.hk, Nov. 2015.

[36] K. Fisher, "The Death of SuprNova.org." *Ars Technica*, http://arstechnica.com/staff/2005/12/2153/, Dec. 2005.

[37] D. Goodin, "Massive denial-of-service attack on GitHub tied to Chinese government." *Ars Technica*, http://arstechnica.com/security/2015/03/massive-denial-of-service-attack-on-github-tied-to-chinese-government/, Mar. 2015.

[38] C. Anderson, "Dimming the Internet: Detecting Throttling as a Mechanism of Censorship in Iran," tech. rep., University of Pennsylvania, 2013.

[39] "HTTP URL/keyword detection in depth." http://gfwrev.blogspot.jp/2010/03/http-url.html, 2010.

[40] P. Winter and S. Lindskog, "How the Great Firewall of China is Blocking Tor," in *Free and Open Communications on the Internet*, (Bellevue, WA, USA), USENIX, 2012.

[41] Renesys, "http://research.dyn.com/2011/01/egypt-leaves-the-internet/," Jan. 2011.

[42] Renesys, "http://research.dyn.com/2011/08/the-battle-for-tripolis-intern/," Aug. 2011.

[43] Renesys, "http://research.dyn.com/2013/09/internet-blackout-sudan/," Sep. 2013.

[44] Renesys, "http://research.dyn.com/2013/08/myanmar-internet/," Aug. 2013.

[45] Anonymous, "The Collateral Damage of Internet Censorship by DNS Injection," *SIGCOMM Comput. Commun. Rev.*, vol. 42, pp. 21–27, June 2012.

[46] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: The Second-Generation Onion Router," in *Proceedings of the 13th USENIX Security Symposium*, Aug. 2004.

[47] P. Winter, T. Pulls, and J. Fuss, "ScrambleSuit: A Polymorphic Network Protocol to Circumvent Censorship," in *Proceedings of the Workshop on Privacy in the Electronic Society*, ACM, Nov. 2013.

[48] M. C. Tschantz, S. Afroz, V. Paxson, and J. D. Tygar, "On Modeling the Costs of Censorship." *arXiv preprint,* http://arxiv.org/pdf/1409.3211v1.pdf, 2014.

[49] D. Fifield, N. Hardison, J. Ellithorpe, E. Stark, R. Dingledine, P. Porras, and D. Boneh, "Evading Censorship with Browser-Based Proxies," in *Privacy Enhancing Technologies Symposium*, (Vigo, Spain), pp. 239–258, Springer, 2012.

[50] P. Lincoln, I. Mason, P. Porras, V. Yegneswaran, Z. Weinberg, J. Massar, W. A. Simpson, P. Vixie, and D. Boneh, "Bootstrapping Communications into an Anti-Censorship System," in *Proceedings of the USENIX Workshop on Free and Open Communications on the Internet*, Aug. 2012.

[51] The Tor Project, "Tor Bridges Specification." https://gitweb.torproject.org/torspec.git/tree/attic/bridges-spec.txt, May 2009.

[52] N. Feamster, M. Balazinska, W. Wang, H. Balakrishnan, and D. Karger, "Thwarting web censorship with untrusted messenger discovery," in *Privacy Enhancing Technologies*, pp. 125–140, Springer, 2003.

[53] E. Y. Vasserman, N. Hopper, and J. Tyra, "Silent knock : practical, provably undetectable authentication.," *Int. J. Inf. Sec.*, vol. 8, no. 2, pp. 121–135, 2009.

[54] D. McCoy, J. A. Morales, and K. Levchenko, "Proximax: A Measurement Based System for Proxies Dissemination," *Financial Cryptography and Data Security*, vol. 5, no. 9, p. 10, 2011.

[55] B. Laurie and R. Clayton, "Proof-of-Work Proves Not to Work," in *In Third Annual Workshop on Economics and Information Security (WEIS)*, (Minneapolis, MN), May 2004.

[56] R. Dingledine and N. Mathewson, "Tor Directory Protocol, Version 3." https://gitweb.torproject.org/torspec.git/tree/dir-spec.txt, Jan. 2006.

[57] H. Mohajeri Moghaddam, B. Li, M. Derakhshani, and I. Goldberg, "SkypeMorph: Protocol Obfuscation for Tor Bridges," in *Proceedings of the 19th ACM conference on Computer and Communications Security*, Oct. 2012.

[58] A. Houmansadr, T. Riedl, N. Borisov, and A. Singer, "IP over Voice-over-IP for Censorship Circumvention." *arXiv preprint,* https://arxiv.org/pdf/1207.2683v2.pdf, 2012.

[59] S. Burnett, N. Feamster, and S. Vempala, "Chipping Away at Censorship Firewalls with User-Generated Content," in *Proceedings of the 19th USENIX Security Symposium*, 2010.

[60] S. Khattak, M. Javed, P. D. Anderson, and V. Paxson, "Towards Illuminating a Censorship Monitor's Model to Facilitate Evasion," in *Free and Open Communications on the Internet*, (Washington, DC, USA), USENIX, 2013.

[61] A. Houmansadr, G. T. K. Nguyen, M. Caesar, and N. Borisov, "Cirripede: Circumvention Infrastructure using Router Redirection with Plausible Deniability," in *Proceedings of the 18th ACM conference on Computer and Communications Security*, Oct. 2011.

[62] I. Clarke, O. Sandberg, B. Wiley, and T. W. Hong, "Freenet: A Distributed Anonymous Information Storage and Retrieval System," in *Proceedings of Designing Privacy Enhancing Technologies: Workshop on Design Issues in Anonymity and Unobservability*, pp. 46–66, Jul. 2000.

[63] M. Waldman and D. Mazières, "Tangler: A Censorship-Resistant Publishing System based on Document Entanglements," in *Proceedings of the 8th ACM Conference on Computer and Communications Security*, pp. 126–135, Nov. 2001.

[64] J. Geddes, M. Schuchard, and N. Hopper, "Cover Your ACKs: Pitfalls of Covert Channel Censorship Circumvention," in *Proceedings of the 20th ACM conference on Computer and Communications Security*, 2013.

[65] A. Houmansadr, C. Brubaker, and V. Shmatikov, "The Parrot is Dead: Observing Unobservable Network Communication," in *Proceedings of the 2013 IEEE Symposium on Security and Privacy*, May 2013.

[66] S. Li, M. Schliep, and N. Hopper, "Facet: Streaming over Videoconferencing for Censorship Circumvention," in *Proceedings of the Workshop on Privacy in the Electronic Society*, Nov. 2014.

[67] Y. Torbati, "Iranians Face New Internet Curbs Before Presidential Election." *Reuters*, http://www.reuters.com/article/2013/05/21/net-us-iran-election-internet-idUSBRE94K0ID20130521, May 2013.

[68] M. Schuchard, J. Geddes, C. Thompson, and N. Hopper, "Routing Around Decoys," in *Computer and Communications Security*, ACM, 2012.

[69] P. Vines and T. Kohno, "Rook: Using Video Games as a Low-Bandwidth Censorship Resistant Communication Platform." http://homes.cs.washington.edu/~yoshi/papers/tech-report-rook.pdf, 2015.

[70] B. Hahn, R. Nithyanand, P. Gill, and R. Johnson, "Games Without Frontiers: Investigating Video Games as a Covert Channel." *arXiv preprint,* http://arxiv.org/pdf/1503.05904v2.pdf, 2015.

[71] T. Elahi, J. A. Doucette, H. Hosseini, S. J. Murdoch, and I. Goldberg, "A Framework for the Game-theoretic Analysis of Censorship Resistance," *Proceedings on Privacy Enhancing Technologies*, vol. 2016, no. 4, 2016.

[72] The Tor Project, "Pluggable Transports." https://www.torproject.org/docs/pluggable-transports.html.en, Jan. 2012.

[73] ifinity0, "Composing Pluggable Transports." *Pseudonymously*, https://github.com/infinity0/tor-//notes/blob/master/pt-compose.rst, Oct. 2013.

[74] ifinity0, "Complete Specification for Generalised PT Composition." *Pseudonymously*, https://trac.torproject.org/projects/tor/ticket/10061, Oct. 2013.

[75] J. Massar, I. Mason, L. Briesemeister, and V. Yegneswaran, "JumpBox–A Seamless Browser Proxy for Tor Pluggable Transports," *Security and Privacy in Communication Networks. Springer*, p. 116, 2014.

[76] D. Luchaup, K. P. Dyer, S. Jha, T. Ristenpart, and T. Shrimpton, "LibFTE: A Toolkit for Constructing Practical, Format-Abiding Encryption Schemes," in *USENIX Security Symposium*, USENIX, 2014.

[77] D. Fifield, "meek." *Tor Blog*, https://trac.torproject.org/projects/tor/wiki/doc/meek, Jan. 2014.

[78] Psiphon Inc., "Psiphon." https://psiphon.ca/en/index.html.

[79] The Tor Project, "Fog." https://gitweb.torproject.org/pluggable-transports/fog.git.

[80] The Tor Project, "obfs3." https://gitweb.torproject.org/pluggable-transports/obfsproxy.git/tree/doc/obfs3/obfs3-protocol-spec.txt.

[81] T. Elahi, J. Doucette, H. Hosseini, S. Murdoch, and I. Goldberg, "A Framework for the Game-theoretic Analysis of Censorship Resistance," Tech. Rep. 2015-11, CACR, 2015.

[82] "Adversary Lab." https://github.com/blanu/AdversaryLab/.

[83] Paul Farrell, "History of 5-Eyes âĂŞ Explainer." *The Guardian*, http://www.theguardian.com/world/2013/dec/02/history-of-5-eyes-explainer, Dec. 2013.

[84] R. Dingledine, "How to Handle Millions of New Tor Clients." https://blog.torproject.org/blog/how-to-handle-millions-new-tor-clients, Sep. 2013.

[85] C. MiscBrubaker, A. Houmansadr, and V. Shmatikov, "CloudTransport: Using Cloud Storage for Censorship-Resistant Networking," in *Privacy Enhancing Technologies Symposium*, Springer, 2014.

[86] B. Marczak, N. Weaver, J. Dalek, R. Ensafi, D. Fifield, S. McKune, A. Rey, J. Scott-Railton, R. Deibert, and V. Paxson, "China's Great Cannon." https://citizenlab.org/2015/04/chinas-great-cannon/, 2015.

[87] E. Dou and A. Barr, "U.S. Cloud Providers Face Backlash From China's Censors." *The Wall Street Journal*, http://www.wsj.com/articles/u-s-cloud-providers-face-backlash-from-chinas-censors-1426541126, 2015.

[88] S. Köpsell and U. Hillig, "How to Achieve Blocking Resistance for Existing Systems Enabling Anonymous Web Surfing," in *Proceedings of the Workshop on Privacy in the Electronic Society*, Oct. 2004.

[89] C. S. Leberknight, M. Chiang, H. V. Poor, and F. Wong, "A Taxonomy of Internet Censorship and Anti-censorship." https://www.princeton.edu/~chiangm/anticensorship.pdf, 2012.

[90] G. Perng, M. K. Reiter, and C. Wang, "Censorship Resistance Revisited," in *Proceedings of Information Hiding Workshop*, pp. 62–76, Jun. 2005.

[91] S. Gardner, "Freedom-supporting technologies: their origins and current state." https://docs.google.com/document/d/1Pa566Vnx9MEuV9gltXOZkypQ3wXyzVyslg6xHuebqqU/edit#, 2016.

[92] R. Dingledine and N. Mathewson, "Design of a Blocking-Resistant Anonymity System," Tech. Rep. 2006-1, The Tor Project, Nov. 2006.

[93] 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.

[94] R. Smits, D. Jain, S. Pidcock, I. Goldberg, and U. Hengartner, "BridgeSPA: Improving Tor Bridges with Single Packet Authorization," in *Proceedings of the 10th Annual ACM Workshop on Privacy in the Electronic Society*, (New York, NY, USA), pp. 93–102, ACM, 2011.

[95] uproxy, "uProxy." https://www.uproxy.org/.

[96] 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 *Proceedings of the 19th ACM conference on Computer and Communications Security*, Oct. 2012.

[97] I. Goldberg and D. Wagner, "TAZ Servers and the Rewebber Network: Enabling Anonymous Publishing on the World Wide Web," *First Monday*, vol. 3, Aug. 1998.

[98] foe-project, "https://code.google.com/p/foe-project/."

[99] L. Invernizzi, C. Kruegel, and G. Vigna, "Message in a Bottle: Sailing Past Censorship," in *Privacy Enhancing Technologies Symposium*, 2012.

[100] Y. Wang, P. Ji, B. Ye, P. Wang, R. Luo, and H. Yang, "GoHop: Personal VPN to Defend from Censorship," in *International Conference on Advanced Communication Technology*, IEEE, 2014.

[101] E. Wustrow, S. Wolchok, I. Goldberg, and J. A. Halderman, "Telex: Anticensorship in the Network Infrastructure," in *Proceedings of the 20th USENIX Security Symposium*, Aug. 2011.

[102] R. Anderson, "The Eternity Service," in *Proceedings of Pragocrypt*, 1996.

[103] MailMyWeb, "http://www.mailmyweb.com."

[104] N. Feamster, M. Balazinska, G. Harfst, H. Balakrishnan, and D. R. Karger, "Infranet: Circumventing Web Censorship and Surveillance.," in *USENIX Security Symposium*, pp. 247–262, USENIX, 2002.

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

[106] D. Fifield, G. Nakibly, and D. Boneh, "OSS: Using Online Scanning Services for Censorship Circumvention," in *Proceedings of the 13th Privacy Enhancing Technologies Sym-*

*posium*, Jul. 2013.

[107] E. Wustrow, C. M. Swanson, and J. A. Halderman, "TapDance: End-to-Middle Anticensorship Without Flow Blocking," in *Proceedings of the 23rd USENIX conference on Security Symposium*, pp. 159–174, USENIX Association, 2014.

[108] W. Zhoun, A. Houmansadr, M. Caesar, and N. Borisov, "SWEET: Serving the Web by Exploiting Email Tunnels," *HotPETS*, 2013.

[109] "Scholar Zhang: Intrusion detection evasion and black box mechanism research of the Great Firewall of China." https://code.google.com/p/scholarzhang/, 2010.

[110] "west-chamber-season-2." https://code.google.com/p/west-chamber-season-2/, 2010.

[111] "west-chamber-season-3." https://github.com/liruqi/west-chamber-season-3/, 2011.

[112] 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.

[113] 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*, (San Francisco, CA, USA), USENIX, 2011.

[114] M. Waldman, A. Rubin, and L. Cranor, "Publius: A Robust, Tamper-Evident, Censorship-Resistant and Source-Anonymous Web Publishing System," in *Proceedings of the 9th USENIX Security Symposium*, pp. 59–72, Aug. 2000.

[115] J. Holowczak and A. Houmansadr, "Cachebrowser: Bypassing chinese censorship without proxies using cached content," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pp. 70–83, ACM, 2015.

[116] D. Ellard, C. Jones, V. Manfredi, W. T. Strayer, B. Thapa, M. Van Welie, and A. Jackson, "Rebound: Decoy routing on asymmetric routes via error messages," in *Local Computer Networks (LCN), 2015 IEEE 40th Conference on*, pp. 91–99, IEEE, 2015.

[117] A. Serjantov, "Anonymizing Censorship Resistant Systems," in *Proceedings of the 1st International Peer To Peer Systems Workshop*, Mar. 2002.

[118] W. Mazurczyk, P. Szaga, and K. Szczypiorski, "Using Transcoding for Hidden Communication in IP Telephony." *arXiv preprint,* http://arxiv.org/pdf/1111.1250.pdf, 2011.

[119] 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.

[120] K. P. Dyer, S. E. Coull, T. Ristenpart, and T. Shrimpton, "Protocol Misidentification Made Easy with Format-Transforming Encryption," in *Proceedings of the 20th ACM conference on Computer and Communications Security (CCS 2013)*, Nov. 2013.

[121] bit-smuggler, "tunnel traffic through a genuine bittorrent connection." https://github.com/danoctavian/bit-smuggler.

[122] 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 *Proceedings of the USENIX Workshop on Free and Open Communications on the Internet*, USENIX, 2014.

[123] K. P. Dyer, S. E. Coull, and T. Shrimpton, "Marionette: A Programmable Network Traffic Obfuscation System," in *Proceedings of the 24th USENIX Security Symposium*, pp. 367–382, USENIX Association, Aug. 2015.

[124] S. Cao, L. He, Z. Li, and Y. Yang, "SkyF2F: Censorship Resistant via Skype Overlay Network," in *Information Engineering, 2009. ICIE'09. WASE International Conference on*, vol. 1, pp. 350–354, IEEE, 2009.

[125] T. Ruffing, J. Schneider, and A. Kate, "Identity-Based Steganography and Its Applications to Censorship Resistance." 6th Workshop on Hot Topics in Privacy Enhancing Technologies (HotPETs), 2013.

[126] The Tor Project, "Tor: Hidden Service Protocol." https://www.torproject.org/docs/hidden-services.html.en.

[127] yourfreedom, "Your Freedom - VPN, tunneling, anonymization, anti-censorship. Windows/Mac/Linux/Android." https://www.your-freedom.net/.

[128] J. Salowey, H. Zhou, P. Eronen, and H. Tschofenig., "MessageStreamEncryption." http://wiki.vuze.com/w/Message_Stream_Encryption, 2006.

[129] anchorfree, "Get Behind The Shield." http://www.anchorfree.com/.

[130] GTunnel, "Garden Networks For Information Freedom." http://gardennetworks.org/products.

[131] The Tor Project, "obfs2." https://gitweb.torproject.org/pluggable-transports/obfsproxy.git/tree/doc/obfs2/obfs2-protocol-spec.txt.

[132] JAP, "JAP Anonymity & Privacy." https://anon.inf.tu-dresden.de/index.html.

[133] Lantern, "Open Internet for Everyone." https://getlantern.org/.

[134] Yawning, "obfs4." *Pseudonymously*, https://github.com/Yawning/obfs4/blob/5bdc376e2abaf5ac87816b763f5b26e314ee9536/doc/obfs4-spec.txt.

[135] 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 *Proceedings of the 19th ACM conference on Computer and Communications Security*, Oct. 2012.

[136] cgiproxy, "HTTP/FTP Proxy in a CGI Script." https://www.jmarshall.com/tools/cgiproxy/.

[137] Ultrasurf, "http://ultrasurf.us."

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

# A Surveyed Censorship Resistance Systems

**Table 3.** Surveyed systems relevant to different schemes of Communication Establishment; † Academic paper, ⋆ Deployed.

| High Churn Access | Rate Limited | | | Active Probing | | Trust-Based Access |
|---|---|---|---|---|---|---|
| | Proof of Life/Work | Time Partitioning | Keyspace Partitioning | Obfuscating Aliveness | Obfuscating Service | |
| Flashproxy [49]†⋆ | Defiance [50]† | Tor Bridges [92]⋆ | Keyspace-Hopping [52]† | SilentKnock [53]†⋆ | ScrambleSuit [47]†⋆ | Proximax [54]† |
| VPN-Gate [93]†⋆ | Köpsell *et al.* [88]† | | | BridgeSPA [94]†⋆ | | uProxy [95]⋆ |

**Table 4.** Surveyed systems relevant to different schemes of Conversation; † Academic paper, ⋆ Deployed.

| Access-Centric Schemes | | | | | | Publication-Centric Schemes | |
|---|---|---|---|---|---|---|---|
| Content/Flow Obfuscation | | | | Destination Obfuscation | | Content Redundancy | Distributed Storage |
| Mimicry | Tunnelling | Covert Channel | Traffic Manip. | Proxy | Decoy Routing | | |
| StegoTorus [96]† | Freewave [58]† | Collage [59]† | Khattak *et al.* [60]† | Flashproxy [49]†⋆ | Cirripede [61]† | Rewebber [97]† | Tangler [63]† |
| FOE [98]† | Facet [66]† | MIAB [99]† | GoHop [100]†⋆ | VPN-Gate [93]†⋆ | Telex [101]† | Eternity [102]† | |
| MailMyWeb [103]⋆ | JumpBox [75]† | Infranet [104]†⋆ | Clayton *et al.* [105]† | OSS [106]† | TapDance [107]† | Freenet [62]†⋆ | |
| SWEET [108]† | CloudTransport [85]† | Castle [70]† | Zhang *et al.* [109–111]⋆ | Domain Fronting [77, 112]†⋆ | Curveball [113]† | Publius [114]† | |
| SkypeMorph [57]† | Castle [70]† | Rook [69]† | CacheBrowser [115]†⋆ | Freewave [58]† | Rebound [116]† | Serjantov [117]† | |
| TransTeg [118]† | Rook [69]† | TRIST [119]† | | Rewebber [97]† | | | |
| FTE [120]†⋆ | Bit-Smuggler [121]⋆ | Facade [122]† | | Tor [46]†⋆ | | | |
| Marionette [123]† | SkyF2F [124]† | IBS [125]† | | Tor Hidden Services [126]⋆ | | | |
| ScrambleSuit [47]†⋆ | YourFreedom [127]⋆ | | | YourFreedom [127]⋆ | | | |
| MSE [128]⋆ | | | | AnchorFree [129]⋆ | | | |
| Dust [18]† | | | | GTunnel [130]⋆ | | | |
| obfs2 [131]⋆ | | | | JAP [132]⋆ | | | |
| obfs3 [80]⋆ | | | | Lantern [133]⋆ | | | |
| obfs4 [134]⋆ | | | | uProxy [95]⋆ | | | |
| CensorSpoofer [135]† | | | | CGIProxy [136]⋆ | | | |
| | | | | Ultrasurf [137]⋆ | | | |
| | | | | Freegate [138]⋆ | | | |
| | | | | CensorSpoofer [135]† | | | |