Censorship Resistant Web Applications

Will Scott

A dissertation
submitted in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy

University of Washington

2016

Reading Committee:

Thomas Anderson, Chair

Arvind Krishnamurthy, Chair

Tadayoshi Kohno

Program Authorized to Offer Degree:
UW Computer Science & Engineering

University of Washington

Abstract

Censorship Resistant Web Applications

Will Scott

Co-Chairs of the Supervisory Committee:
Professor Thomas Anderson
UW CSE

Associate Professor Arvind Krishnamurthy
UW CSE

Of the countless individuals around the world confronted by Internet censorship every day, only a small fraction are successful in reaching their destinations. This is a failing of the censorship circumvention community. The widespread nature of censorship reflects our inability to understand, work around, or fix the problem as a whole.

This dissertation presents novel approaches to measuring, circumventing, and resisting Internet censorship without relying on users. In particular, we present (a) Satellite, a system for remotely monitoring the state of DNS-based censorship. Satellite is able to document new episodes of censorship and shed new light on already documented cases. (b) uProxy, a system for in-browser circumvention based on social trust and private routes for individual users. Through circumvention, we deploy an easy-to-use system that is already in use by tens of thousands of users. (c) Activist, a library allowing web publishers to circumvent many forms of censorship without any user involvement. Through resistance, we have established an alternative approach to censorship circumvention. A particular focus is the implementation of these systems; all three exist as public, free, open-source code.

TABLE OF CONTENTS

LIST OF FIGURES

GLOSSARY

AS: Autonomous System. A unique identifier assigned to each distinct entity providing Internet service (The University of Washington is AS 73).

CDN: Content Distribution Network. A globally distributed system maintaining multiple copies of web content to provide low-latency access to users in diverse locations.

DNS: Domain Name System. The process by which a domain, like `google.com` is converted into a IP address of an Internet host.

DNSSEC: DNS Security. An extension to DNS allowing responses to be signed or authenticated by the owner of the domain.

DPI: Deep Packet Inspection. A technique which routes Internet packets based on their contents, rather than headers.

HTTP: Hyper-Text Transfer Protocol. The protocol used by web browsers and other applications to request resources from remote servers.

ICP: Internet Content Provider. A license required for web publishers in China.

IETF: Internet Engineering Task Force. The governing body which maintains standards documents specifying behavior of Internet protocols.

IP: Internet Protocol. The encapsulation of traffic on the Internet and accompanying addressing system.

NETWORK INTERFERENCE: Network Interference describes unexpected blocking, interruption, or manipulation of traffic by a router in the network. Unlike censorship, it does not ascribe intention to those actions.

OBFUSCATION: A variety of techniques to obscure traffic, either by making it look 'random', or by disguising it to look like another type of traffic.

OONI: Open Observatory of Network Interference. An open-source distributed measurement platform associated with The Tor Project.

PTR: DNS Pointer record type. The record type used for a reverse lookup from an Internet-reachable host to an associated name.

RENDEZVOUS: A system to assist in establishing a connection between two users.

STEGANOGRAPHY: A variety of techniques to hide messages within other communications. In contrast to Obfuscation, steganography is typically performed within a legitimate communication channel.

SOCKS: "Socket Secure" is a protocol for indirecting TCP traffic through a proxy server. It does not guarantee authenticity or confidentiality of traffic, despite the use of the word secure. It is however one of the most widely used protocols for proxying traffic.

SRI: Sub-Resource Integrity. The concept that when a web-site includes a resource from a different domain, it needs to trust both the network and the other domain to serve the expected resource. Recent work has started to define ways for the requester to validate the integrity of such resources.

STUN: Session Traversal Utilities for NAT. A UDP protocol by which a client can discover where it appears to be to an external observer, in order to receive incoming communication from others [166].

TLS: Transport Layer Security. The Internet Standard for encrypting HTTP streams as HTTPS.

TOR: The Onion Router. A software system providing online anonymity through multiple steps of indirection [55].

RST: A TCP packet with the Reset flag set. This indicates that something is wrong with the stream of data, and the connection needs to start over if the other host wants to continue. Sending a RST packet is a frequent mechanism used by censors to stop unwanted streams of communication.

VOIP: Voice Over IP. Refers to several different protocols for transmitting voice (and sometimes video) in real time over the Internet.

W3C: World Wide Web Consortium. The governing body specifying the APIs and rendering standards across web browsers.

WEBRTC: Web Real Time Communication. WebRTC refers to both the browser API and network protocol for communication between two client web browsers.

XHR: XmlHTTPRequest. The Web API allowing a web application to request data from a server.

XSS: Cross Site Scripting. A security attack where a script from one domain is executed in the context of another. The attack in its canonical form allows an attacker to steal the credentials of users who visit a maliciously crafted link.

ACKNOWLEDGMENTS

This thesis would not have been possible without Arvind Krishnamurthy and Thomas Anderson, who have served as the best guides and mentors I could have asked for through the last five years.

I am extremely fortunate to have found myself in the Networks Lab, and have grown from the interactions and inquiry that it has engendered. Raymond Cheng, who collaborated with me on the uProxy and unblock work, will find his ideas reflected throughout this text. I am fortunate to have worked with him. The experience would not have been nearly as smooth without Melody Kadenko working behind the scenes to keep everything running.

Thanks to Lucas Dixon and the team at Jigsaw for believing in uProxy, and taking on the brunt of the work to make it reality.

Thanks to Open Technology Fund for supporting my work on publisher-side resistance.

I am humbled by the generosity and inspiration I've found over the last five years. Thanks to all of my collaborators and friends both at the University of Washington and around the world for listening to me and challenging my ideas.

I have had the fortune of calling myself part of many vibrant communities along this journey. I'm proud to have played a role in PUST, ELS, #IT, WIBSTR, and Tor.

And to my parents Odette Batik and Jerry Scott for their unconditional love and support.

Chapter 1
INTRODUCTION

> Just because revolutionary cyber-Soviets or Robin Hood-style cyber-vigilantism are not the answers to our problems does not mean that business and government as we know them today are serving the needs of today's citizens and netizens. New approaches to governance, accountability, and politics clearly are needed if democracy is to survive and thrive in the Internet age. (Rebecca Mackinnon, *Consent of the Networked*. p. 248)

1.1 Problem Space

In its first session in 1946, the United Nations adopted Resolution 59, stating that "Freedom of information is a fundamental human right." Seventy years later, the growing predominance of the Internet as a global communication medium once again highlights the struggle to realize that resolution.

The growth of the Internet has brought with it disruption of pre-existing information control structures. In response to this perceived threat, governments and ISPs have erected technical information controls to limit and shape access to online information.

While online information controls take a multitude of forms, the most successful are implemented at natural choke-points of centralization, the core routers making up the fabric of the Internet. These routers are owned and controlled by large companies and governments which can implement information access policies consistent with local laws and social norms.

Restrictions and disruption of Internet traffic have become the norm, and there are few voices advocating for zero oversight of this communication. While aspirational belief in the

UN resolution remains, the Internet blurs the line between speech and publication. Our social contract limits our speech and actions in the real world. Why should speech online be different? Instead, the point of contention comes in defining where to draw the lines of what constitutes objectionable speech. Is the line at child abuse, graphic images of terrorism, challenges to state authority, or is it at commercial harm or disruption of social harmony?

In parallel to the shifting lines demarking these norms and regulations are circumvention efforts motivated by politics, personal access to unavailable services, and by security and privacy concerns. These "liberation technologies" offer special privilege to the technically enlightened few. Much effort is needed to disguise and deliver information to users in an escalating battle of obfuscation and steganography to stay ahead of current and future control systems.

1.2 Goals

This dissertation is an exploration into the mechanisms of interference used to disrupt web sites today. The text describes how interference can be detected and understood by web publishers, and how new techniques can compensate for the disruptions imposed by purposeful censorship. The goal is not to put an end to disruption but rather to advance the conversation with data and transparency. I hope that these ideas provide inspiration in how best to harden the web as we know it against network controls.

My primary goal is to challenge one of the implicit assumptions that has driven the area of technical circumvention for the last decade: that end users are responsible for getting access to content in spite of obstacles placed by intermediate networks. Viewing censorship as only a problem for users has driven the ecosystem of circumvention tools, detracts from Internet standards development, and perpetuates real-world class divisions into the digital realm.

In contrast, this dissertation shows that *Internet censorship can be understood and effectively resisted without relying on user cooperation.* There are three primary contributions supporting this thesis: Satellite provides a proof of existance of a system remotely measuring

DNS-based censorship without user support. uProxy presents a circumvention system design that is difficult to disrupt while reducing the need for user trust. Activist provides a suite of techniques for publishers to unilaterally control the messaging and to serve content in spite of an adversarial network.

The remainder of this chapter provides an overview of these contributions. These three systems taken together provide the proof of existance showing how network information controls can be countered as a whole by platforms rather than piecemeal by users.

1.3 Satellite: Understanding web censorship

Interference on the web is motivated by denying access to content, but the mechanisms used vary. A significant corpus of work has been produced documenting these mechanisms, as we discuss in 2.5, but this work has not produced proactive systems to alert researchers of new censorship practices or an open and trusted archive for retroactive analysis of previous censorship events.

Satellite is a system for systematic and sustainable collection and documentation of censorship events. Satellite has shown that taking widespread measurements is practical, with multiple years of data showing a multitude of changes in censorship policy. Satellite offers a number of improvements on previous approaches to quantifying web censorship:

- A data respository of web censorship collected consistently and sustainably without end-user participation.

- Evidence from the repository that popular domains are blocked, including in countries and ISPs where there are no vantage points actively cooperating with researchers.

- A technique to isolate censorship events from the confounding appearance of geographically distributed CDN infrastructure.

1.4 Uproxy and Unblock: Circumventing censorship

Web interference is inherently a cat and mouse game between limitations imposed by networks and users attempting to sneak around those limitations.

We introduce two systems to circumvent censorship in a way that is harder to block. Unblock and uProxy both focus on using existing trust relationships between users. Many users already have friends in other countries or networks, and routing your Internet traffic through friends can be much harder to block. The two systems explore different parts of this problem, with different approaches to availability and user interface design.

Unblock is a proxy system similar in structure to Tor, but targeted at low latency circumvention over computers connected by the social graph of their owners. One of the major problems with only sending traffic to friends is that those users with few friends participating in the system will be left with poor service. Unblock applies transitive trust (friends of friends) to address this while minimizing how much damage an adversary can do to the system. We quantify this trade-off between performance and network resilence.

uProxy is also a proxy system, but it targets the ease of adoption through integration with the web browser. uProxy runs as a simple one-hop proxy, and helps users to connect with their friends on existing social networks. uProxy is in use by tens of thousands of users. The uProxy client paves the path for a circumvention client without the need for explicit user installation, since the web technologies it uses are all supported in an untrusted and unpriviledged browser.

1.5 Activist: Resistance for everyone

Censorship circumvention is finally making inroads into the technical realm of standards bodies. While the groups developing protocol and web standards have explicitly considered issues of privacy and security, their multi-stakeholder model has made for slow progress in advancing protections that are seen to have an explicit political agenda. Beginning in 2014, Article 19, an organization working to promote human rights protection, began the process

of requiring a statement considering the impact on human rights to be part of the IETF process for protocol standardization [187].

We introduce activist.js, a library giving publishers a unilateral way to improve the censorship experience for users. The library updates a publisher's website to maintain the ability of users to view and share content even when the network connection to the publisher is censored. We do this through existing, widely-deployed web APIs that require no cooperation from users or browser platforms.

Activist.js addresses several limitations imposed on users through censorship. It is not just access to an individual URL that must be addressed, but also how to share content and techniques for more general circumvention. Activist contributes a number of mechanisms mitigating these controls:

- Resilient links that show publisher content even when the domain is censored.
- Caching of error pages to retain control of messaging in the event of censorship.
- Indirection of network communication to fetch updates when direct communication to the server is blocked.

The creation of Activist also revealed the limits of what cannot yet be solved by publishers. This line of inquiry helps advocates to focus their efforts in browser development and standardization to address the remaining lines of attack by a network adversary. We find that many of the features that could address remaining limitations are not direct reactions to censorship. Rather, they lead to a better web experience more broadly.

1.6 Organization

The remainder of this dissertation is organized into five parts.

Chapter 2 surveys the field of measurement and circumvention of network interference. The chapter covers the technical mechanisms used for censorship, and how they are motivated and implemented. We then look at the underlying principles that have emerged for resisting

censorship and what properties make a circumvention system successful. In the final part of the chapter, we step back to look at where this understanding of censorship comes from, and the techniques that have been used to document network interference.

Chapter 3 presents Satellite, a system to document web censorship openly, sustainably, and without user involvement. Satellite collects observations from existing web infrastructure. We describe the approach, evaluate correctness, and explore some observations of censorship made by Satellite.

Chapter 4 presents Unblock and uProxy, two systems for censorship circumvention based on the principle of social trust. Combining the ideas of collateral damage, IP diversity, and existing trust, these systems present different design points for practical censorship-resistant distributed overlays.

Chapter 5 presents Activist, a library allowing publishers to respond to censorship. Countering the common assumption that circumvention is a process initiated by users, Activist describes a path for web publishers and browsers to build a resilient web without users needing to take any action. We describe the approach, prototype a complete system, and document successful deployment of the technique.

Chapter 6 looks onward to the future. It follows the evolving threats that led to Satellite, Unblock, uProxy, and Activist. From these threats come opportunities for future work and accompanying challenges in the space.

Chapter 2

BACKGROUND

One of the principles guiding the design of the Internet was to put intelligence at the edges of the network [36]. Doing so makes it easier to introduce new communication protocols, since only those machines, rather than all of the routers in between them, need to be upgraded to understand the new technology.

This principle is reflected in the stack of Internet protocols. IP forms the "narrow waist", specifying only a source and destination, along with an indication of what type of encapsulated data is contained in the message. Within this same format flow TCP "streams" - ordered flows of data used for HTTP web traffic and many other protocols, and UDP "datagrams" - individual messages used by applications like DNS and VOIP.

The choice of a "dumb" network makes the life of a censor more difficult, since modifying or transforming packets in the middle is unexpected in protocol design. Many protocols (including DNS and HTTP) do not provide the flexibility to manipulate user experience in the way the censor might desire. For instance, there is no protocol-defined way for a network intermediary to indicate content has been denied to clients using an HTTPS connection. This is because the TCP protocol assumes that messages it receives will be from the remote computer it is speaking with. Messages from a router in the network will not be able to authenticate themselves as the remote server, and the protocol does not support unauthenticated messages.

However, these limitations are offset by another theme of the the original Internet protocols: that security was not a high priority [169]. None of the commonly used Internet protocols, including DNS, BGP, or HTTP were encrypted or authenticated in their original forms. Today, decades later, a challenge we face is retrofitting communication to provide

security and authenticity.

In the rest of this chapter we will first discuss how the web works (Section 2.1), and then survey the technical forms of interference (Section 2.2), and how interference is portrayed to users (Section 2.3). Then we switch gears to survey the mechanisms used to resist and circumvent censorship (Section 2.4), and the techniques used to measure the phenomenon (Section 2.5).

2.1 The Web

One of the most important forms of electronic communication today, the HyperText Transfer Protocol protocol (HTTP) is often referred to simply as "the web." The HTTP protocol defines how web browsers communicate with remote servers; it has been adopted for use by many other applications. HTTP, in conjunction with its encapsulation in Transport Layer Security encryption (HTTPS), is ubiquitous in network communication, and its behavior defines the Internet experience for most users.

The web has evolved substantially since its initial introduction as an inter-connected collection of text documents. In the 1990's, websites like Yahoo paved the path towards online commerce, and in the early 2000's, Gmail became one of the first web "applications." Gmail used new browser capabilities to change the displayed content dynamically, starting a rapid trend toward moving presentation and business logic onto the web client. Since then, the web platform - the API provided to web publishers by web browsers and standardized by the W3C - has evolved rapidly to support new methods of communication, capabilities for interacting with the client device, and significant speed improvements.

Today, web browsers such as Chrome and Firefox provide an API with functionality almost matching that of an operating system. Web applications can save and read data, dynamically control rendering - even directly interact with the GPU, and perform complex window management. This functionality has allowed for a rapid expansion in web apps and services. Providing a service or a web application has major advantages compared to traditional desktop applications. The developer no longer has to worry about supporting

old versions of their code and gains platform independence. The accompanying distribution model has proven extremely attractive.

One sore spot for the web model throughout its development has been offline compatibility. Even as the the first web applications emerged, it was recognized that we are nowhere close to a fully connected world, and it is insufficient to provide services that are only available to users connected to the Internet. Early browsers provided a mechanism to "work offline", which would save a copy of each static document you were currently viewing so that you could continue to access them when disconnected. In 2007, Google released Google Gears, a browser extension providing increased capabilities to web applications, including the ability to store data for continued offline use [30]. Subsequently, these ideas have moved into browser standards, first as the Application Cache API (2010), and recently as Service Workers. The new function provided by these approaches is the ability for a web site to specify some content - HTML, javascript, and images - that should remain on the viewer's computer. That content can then continue to "work" even when the remote server cannot be reached. It does not allow a user to see a site that was not previously visited, or perform actions the developer did not explicitly make possible offline.

A significant issue that hindered development of offline web applications was the significant dissonance between offline applications and how the web platform ensures security and 'integrity' of content. The HTTP protocol provides no indication of authenticity - the content received may be changed by the network and there is no way for the browser to know that the correct information was received. The use of TLS encryption to access the web 'securely' mitigates part of this problem, since it provides a level of trust that the content was sent by a server speaking authoritatively for the domain. However, there still remain issues with resources hosted on CDNs or 3rd party servers, and many domains do not fully support HTTPS. A local application which could affect all future accesses to content was seen as dangerous in this respect, since it runs opposite to the 'stateless' expectation of the Web. In the web standards community today, most new APIs which provide 'privileged' functionality (like ServiceWorkers) are restricted to only run on websites which are served

securely over HTTPS [203], however it has taken a decade to reach this expectation.

The second area where web browsers have struggled to converge on a standard is the interface for how sites can cause a user to establish connections. An early established precedent was that sites could include resources that were hosted on other servers[1]. This choice is at the crux of the complex web security model we live with today and that has enabled a continued stream of cross site scripting (XSS) and sub-resource integrity (SRI) attacks. The general principle guiding communication in the web platform is that if both parties are 'okay' with connecting, then the connection should be permitted. This principle has slowly expanded the ability of web applications to retrieve data from remote servers through XMLHTTPRequests (XHR), maintain a conversation with WebSockets, and track progress and performance of local network conditions.

Making connections between two browsers is a specific form of communication that has been seen as desirable for a long time but only recently gained standardization. In 2008, Adobe Flash 10.0 introduced Real Time Media Flow Protocol to provide the direct connections between two clients, which was most commonly used by multi-player games to reduce latency between players. The protocol was designed to handle video and audio streams at low latency, and the same protocol was often used with a central server for early web video chatting and streaming applications. Competition and development of video streaming technology resulted in a wave of browser extensions and vendor-proprietary protocols like Google Chat, Skype, and Cisco WebEx Connect. In 2013, largely driven by a browser aversion to security flaws and desire to provide a real-time communication solution within Chrome, an initial version of WebRTC was developed. WebRTC provides an interface similar to the previous Flash interface, but embedded within the browser directly without the need for plugins. It has since been adopted by all of the major browsers.

Sending traffic between two browsers is technically much more complex than the HTTPS

[1] Most types of resources were allowed to be served from other domains, a mechanism now dubbed "cross-origin request sharing". These include Javascript code and images, CSS styling definitions, fonts, and arbitrary data files.

protocol used when communicating with a website. HTTPS relies on the web server running on port 80 of a publicly advertised IP address. Most browsers do not have access to port 80 of their local IP or run on a computer without a public IP address [131]. Rather, most end-user computers are on local networks bridged to the Internet by a gateway. The gateway shares a single IP address across of set of clients, performing Network Address Translation where internal addresses are re-written as traffic passes in and out of the public Internet. To make a connection between two computers on separate internal networks, a complex dance is needed to trick the intermediate gateways into allowing the connection [166]. This exchange work best with UDP rather than TCP traffic, and WebRTC is built on UDP. Additional coordination is needed between the two computers, since neither initially will know the port and address of the other. WebRTC supports this through a signaling channel, where messages need at first to be relayed back-and-forth between the two browsers through a server or other existing communication channel in order to set up the direct connection.

Most of the US and Western Europe use one of three browsers: Google Chrome, Mozilla Firefox, and Microsoft Edge (an evolution of Internet Explorer). In the emerging mobile market, however, there is much more diversity in how people access the web. In China, 360 safe browser is probably more popular than Internet Explorer, though the statistic is difficult to verify [200]. In practice, these differences are less intimidating to a developer than they might initially sound. Building a browser is a huge amount of effort. Many of the 'new' or 'alternative' browsers avoid this effort by leveraging existing components to do most of the work. Most browser comparisons of Chinese users indicate a majority share by Internet Explorer and Chrome, since most of the Chinese browsers are indeed just re-packaging of those two rendering engines [38].

2.2 Interference Techniques

Due to the limitations in Internet protocol design, a number of creative solutions have emerged for network interference. It is useful to think about these techniques in terms of the experience they create, the motivations that led to them being implemented, and

the technical complexity and costs of implementation. One categorization of interference techniques is how deeply into the packets do they look. In this spectrum, we start with techniques that only manipulate IP headers, then those that look at TCP headers, then those that examine DNS and HTTP protocol information, and finally those that use 'deep packet inspection' to look at the content of communications.

2.2.1 IP Manipulation

The most basic form of network manipulation occurs at the host level, where packets coming from or destined to specific other IP address or networks can be denied by the service provider. While we optimistically consider the Internet to be *self-healing*, that it routes around failure, this behavior does not happen if network elements block communication between selected endpoints.

A canonical example of routing as an avenue for traffic manipulation occurred in Pakistan in 2008, when Pakistan Telecom claimed ownership of part of the YouTube network space [31]. This achieved their desired goal of preventing customers from accessing YouTube content by sending traffic from clients in Pakistan destined for YouTube to servers they controlled. However, they accidentally also exposed this policy to other ISPs. Due to the nature of BGP[2], their policy resulted in a significant fraction of global traffic intended for YouTube to instead travel to Pakistan for a period of 3 hours.

While this incident is notable because the misconfiguration caused the denial of service to be propagated to much of the Internet, blocking specific IP addresses is more common. Several companies calculate reputation for IP addresses, and deny connections from hosts they associate with spam and abuse [163, 162, 100]. Countries have blocked the IP space of countries with which they are at war [20, 132]. It is also common practice to temporarily deny traffic from networks acting abnormally, such as in reaction to denial of service attacks, or because of receiving malicious traffic.

[2]The Border Gateway Protocol indicates to peers what destination addresses an ISP originates or is willing to forward.

2.2.2 TCP Manipulation

Most routers on the Internet today are capable of understanding the semantics of an IP packet, and commonly make routing decisions based on not only the IP address, but also on the "5-tuple" of the source and destination host IP address and port number, and the protocol of the message. That is, a policy will be able to express that TCP traffic traveling to destination port 25 should be denied while traffic coming from TCP port 80 should be routed on a more expensive link.

This is the most basic level that control over content is exerted, since no special hardware needs to be acquired to implement policy at this level. For instance, many networks block users from sending email directly, since it is an uncommon task and is more often performed by spammers than for legitimate reasons. This is accomplished by blocking outgoing connections on port 25, the port used by SMTP. On the other hand, this control can also be used to control traffic in an adversarial manner. In Iran, there was a period of 2 days in 2012 where connections leaving Iran to port 443 were denied [29, 161]. This action prevented the bulk of encrypted communication with web sites outside of the country; it forced users to access external sites using the unencrypted port 80 which could be more easily monitored by the network adversary [161].

Technically, there are two different mechanisms by which a TCP-level policy can be implemented. The first is the same mechanism that will work for IP packets - packets that match are silently discarded. The other mechanism is to inject a packet indicating that the connection should be closed or reset. Injection of TCP `RST` packets provides additional flexibility in how a policy is implemented. It is speculated that for high-speed networks, it may not be possible to implement policy at line rate - potentially due to cost or the configuration of the network, or potentially because the network is performing a more processor-intensive policy than just examining the TCP header. With `RST` packets, a TCP stream in progress can be disrupted by a machine that is not on path[3], allowing for more flexible network con-

[3]The term "on path" refers to any one of the routers forwarding traffic between the two end hosts.

figurations. The Great Firewall of China is an example of a network which injects TCP `RST` packets [42].

When a policy is implemented on a TCP 5-tuple (as opposed to the information in an IP header), it becomes difficult to fully enumerate or observe that policy. Even if an observer had access to a diverse set of source IP addresses, the 16 quadrilion combinations of source and destination ports, and destination IPv4 addresses is too many to practically enumerate. Especially with measurement techniques requring cooperation from both end hosts, it is infeasible to learn a full policy. Even though the censor will have limited capacity for rule making, finding which restrictions are implemented at any point in time is like finding a needle in a haystack – the search space is huge.

2.2.3 DNS Manipulation

Manipulation of traffic based purely on the 5-tuple is often not sufficient for the goals of a network censor. In particular, the objectionable content may not be hosted on a unique IP address. Instead, if that site is hosted on a shared CDN like CloudFlare or a shared blog, censorship of those flows would also block many other unrelated sites.

One response is to take advantage of the fact that IPSs provide DNS resolution service to their users. The domain name system is the network system that allows computers to convert human readable domain names into IP addresses. The registration of names is strutured as a tree split on the periods in a domain name. For example, the '.com' section of the tree is owned by Verisign, who delegate the 'google.com' subtree to Google. A DNS lookup follows this same chain of delegations, first asking verisign who owns the subtree, and then asking the customer the IP address.

In order to reduce the load on DNS servers, and prevent every client from needing to individually query the resolution of each domain, the DNS system makes use of caches - servers which aggregate client queries, and can serve common queries without re-requesting the authoritative answer. In practice, ISPs often run these servers to provide fast resolution to their users, where most common domains can be resolved locally without going back to

the official server. Running a DNS resolver allows the relevant ISP or country to disrupt access to any domain they want to block.

Disruption of the DNS protocol by a local resolver is technically easy. The DNS protocol is not authenticated – although proposals for doing so have existed for more than a decade. The DHCP protocol which is used to allocate IP addresses to consumers of an ISP also provides a mechanism for the ISP to provide a default DNS resolver to the client. This is generally preferable for clients, since the DNS resolver suggested by the network will typically be nearby, and it will be able to answer queries much faster than resolution by either a public service or by the authoritative server. A local filter within the resolver is easy to add: specifying a list of domains which return specific IP addresses. This incorrect resolution is observed in 117 countries; in 2016 it appears to be the most common form of network interference in the world [172].

These attacks are all enabled by the lack of authentication within the DNS protocol. The response from the authoritative DNS server is both unencrypted and unsigned, thus the local proxy can easily substitute a false result. We might hope that the client would have a way to ensure that the answer it receives is as defined by the owner of that domain. Such a protocol exists, DNSSEC, which allows DNS responses to authenticate through a tree of authority signatures [19]. Ten years after the development of the DNSSEC protocol, adoption remains low. First, the feature required cooperation between the domain owner, the owner's name server provider, and its registrar. Second, key management at the registrar level is quite complex. Third, the protocol forces the domain owner to attest not only that a subdomain exists (like `www.example.com`), but also that other subdomains (`malicious.example.com`) do not exist. While this prevents an adversary from being able to impersonate a participating domain, it also allows an attacker to discover all of the subdomains that are registered and makes it difficult for sites to operate wildcard or per-user subdomains.

In addition to DNSSEC, steps have been taken towards providing confidentiality to the DNS protocol, but they remain even less developed or adopted. These proposals range from T-DNS (sending DNS over a TLS-encrypted TCP connection) and DNSCrypt (using

a custom encryption wrapper) to DNSCurve (a custom encryption protocol built within the existing DNS protocol format). These protocol have faced similar challenges to adoption as DNSSEC, because they impose a higher load on DNS servers and because adoption would require deployment to infrequently updated devices including consumer home routers and ISP switches.

The misuse of DNS has led to countermeasures by users in several countries, notably Turkey and Iran [78, 20]. By governmental mandate in both countries, ISPs configured their local DNS resolvers to resolve contentious domains to an IP address which showed a page indicating the request has been blocked. In Iran, this IP address was a local address, 10.10.0.30, which is not available outside of the country. In both cases, it quickly became common knowledge that the disruption could be easily avoided by configuring your local device to request DNS resolution not from your ISP but from one of the public DNS services. This resulted in an iconic image of '8.8.8.8' spray painted on a wall in Turkey[4] as an advertisement to help others avoid censorship by using the Google Public DNS service.

Redirection of unwanted domains to a network-controlled server is not the only way to configure a DNS resolver to manipulate traffic. In Pakistan, the local resolvers instead indicate a failure when sensitive domains are requested [151]. This results in the client getting a technical error claiming that the domain they are attempting to access cannot be reached. This mechanism has less transparency than resolution to a block page, in that it appears to be a technical failure of the website. China will provide a somewhat random IP in response to requests for domains that are blocked [214], which also generally behaves as a technical failure. Providing a random IP requires the browser to attempt a connection to the resolved IP, and can potentially cause the user to wait much longer before realizing the requested website will not load.

Censors soon come to realize that simply returning manipulated responses is insufficient. After users realize that the ISP resolver removes their ability to access desired content,

[4]One report of this phenomenon is documented by France 24.
http://observers.france24.com/en/20140321-graffiti-turkey-DNS-twitter-ban

changing the DNS resolver is simple enough that it can and has been adopted by large numbers of users [78]. To combat the ease of circumvention, the ISP can monitor DNS requests, providing a custom routing policy for all UDP packets to destination port 53. Those which are requesting resolution of sensitive domains can then be manipulated either by returning a meaningless response (the standard technique), or by injecting a fake response as in China [193].

DNS manipulation of a different form has occurred in the United States. The good news is that the US government (apparently) cannot compel ISPs to incorrectly resolve contentious domains or to instruct clients to use a DNS resolver controlled by the government. Instead, the government has successfully 'seized' control of the Domain names themselves, defining them as property owned by entities which are considered criminal within the laws of the country. Since the registration process places domain names into the hand of registering companies, many of whom operate within the United States, this tactic has been effective at preventing access to gambling, pornography, and copyright-infringing sites [137].

2.2.4 HTTP Manipulation

Perhaps counterintuitively, the systems which are the most complex to implement technically are also the ones which first highlighted the extent of Internet censorship. It was not routing policy which caused China to be outed in its enthusiastic manipulation of Internet traffic, but rather its keyword-based policy, which looked at the actual content being transmitted. Because these systems take significant technical work, they indicate substantial resources and attention have been devoted to the problem.

China remains one of the most technologically sophisticated networks at a national level. While some remnant remains of the original system for keyword-based discrimination, the country has supplanted this mechanism with newer forms of control [207, 133]. These newer forms include pre-emptive probing of content to determine IP-level blocking, and injection of content to induce denial of service attacks. Despite these advances, HTTP-based manipulation remains in place, and receipt or querying of sensitive content over an unencrypted

HTTP channel can result in a user's connection receiving degraded responses or resetting new connections for a matter of minutes [193].

The newest form of manipulation in China is also a form of HTTP manipulation. When foreign visitors accessed Baidu in March and April of 2015, the response from the server was sometimes manipulated to cause additional Javascript content to be loaded [133]. This script would cause the browser to make a series of connections to sites targeted by the Chinese government, and become part of a large distributed denial of service attack on those services. This technique has been dubbed 'The Great Cannon,' as a riff off the Great Firewall term used to describe China's previous censorship mechanism.

Elsewhere, keywords, URLs of specific pages or directories, or naive topic classification based on page content are used for censorship [58]. Some of these systems, like Squid, have been seen scaled from individual WiFi hotspots up to ISP-scale filtering systems [76]. At larger scale, purpose-built hardware can perform similar matching, with higher performance and an easier to use interface [134]. These systems are rarely able to look 'into' encrypted packets, limiting the filtering capacity for HTTPS traffic to the 5 tuple.

HTTP manipulation is used not just for censorship, but also by malware and malicious networks who use it for profit. The Great Cannon was seen as a major escalation in Chinese censorship capabilities, but the same technique has been observed earlier at an ISP level. ISPs and other core networks use manipulation of HTTP streams to send malware to selected targets, and for injecting or manipulating advertisements for direct profit [146].

2.2.5 Motivation

The 'what' of Censorship is as complex as the 'how', and while left largely out-of-scope of this thesis, it illustrates how technical systems of control are implemented. In particular, the diversity of political situations and centralization of power cast light onto where in the network censorship takes place [193, 15, 48]. Beyond ownership, whether the censoring infrastructure is directly operated by a government, or by corporate ISPs under a government mandate, results in different technical approaches. The motivations behind censorship have

ramifications on the technical mechanisms that are employed both to implement and counter it.

We can see these effects in the mechanisms chosen to implement censorship. If the list of objectionable content that an ISP is asked to disrupt consists of individual videos, they may be more inclined to use an HTTP level approach with a Deep Packet Inspection (DPI) box. Access to objectionable domains can be more simply prevented with a DNS mechanism. A governmental approach may result in policy which applies to international traffic, not between mobile and home users. We find numerous examples of policies implemented non-uniformly by ISPs, and especially that consumers of smaller ISPs often escape the imposition of more technically advanced controls [194, 84].

2.3 Censorship Visibility

The ways in which censorship becomes apparant to users and researchers differs by technical implementation and political climate. In some networks, users are presented with a page indicating that content has denied, providing confirmation that the network has taken action. More commonly though, network interference manifests as a technical error, and users cannot easily determine if the remote server has failed, there's been a temporary technical failure at their ISP, or if the content has been blocked for policy reasons. We will referthis issue of how interference manifests itself to users as the visibility of censorship.

Censorship visibility has proven to be one of the most difficult aspects of the phenomenon to document. It is easier for researchers to document the space of censorship in networks with explicit notification of their policies, but perversely these networks are often less intrusive in their censorship. Censorship visibility as a metric – how much interference users experience in their normal lives – is much harder to characterize. To demonstrate this phenomenon, we describe two censorship regimes, those of Turkey and China. One of the interesting points is that while the policies implemented in Turkey were less technically sophisticated and more transparent, they were also seen as less acceptable and more 'visible' to users than the more subtle methods employed in China.

2.3.1 Turkey

In 2007, Turkey passed legislation to extend the enforcement of existing laws in the online space [5]. The new law catalogued nine areas of existing crime, and created a regulation entity, the Presidency of Telecommunication and Communication, with the ability to block content violating these laws through "DNS tampering or IP blocking." It also allowed national courts to demand the blocking of any additional website not covered under the mandate of the new entity.

Even before the law's passage, and escalating afterwards, Turkey blocked popular websites. `youtube.com` was blocked in 2007 before the law was introduced for a video judged to defame the founder of the current republic, Mustafa Atatürk. YouTube was subsequently blocked for a two year period in 2008. Likewise, sites including `wordpress.com` and `twitter.com` have been blocked for the hosting of political speech. The implementation of censorship was performed by returning incorrect DNS resolutions from ISP controlled resolvers [193] (cf. Section 2.2.3).

While interference with high profile sites has been frequent since the introduction of the law, circumvention methods have also gained significant prominence. In addition to significant activism and widespread documentation on how to change DNS resolvers, circumvention in urban areas was largely tolerated without the introduction of more enforceable mechanisms. One notable example occurred in 2014 during a period where Twitter was blocked; the president, Abdullah Gul, tweeted his opposition to the ban of the site [186]. While circumvention was common amongst the technically knowledgable, the limits imposed on political speech were seen as effective in shaping perceptions in more rural areas of the country.

Since the initial blocking, Turkey has continued to develop its information controls, and it has become more successful in controlling online content. There are now over 80,000 websites blocked in the country [91]; Twitter now restricts a significant amount of content considered objectionable by the government from being accessed in country [188]. In 2015, the government amended its laws to allow the unilateral blocking of domains, as long as the

judicial branch retroactively approved of the action [93]. The initial censorship implemented only at ISP resolvers has been extended to active manipulation of remote DNS requests, increasing the barrier to circumvention [37].

Despite these controls, there are significant portions of society who oppose the current censorship regime through activism and circumvention. This counter-culture is most iconically represented by the 2013 Gezi Park Protest [9]. The protest, sparked by redevelopment of a public park, quickly turned into a demonstration for increased rights of expression and assembly. Notably, the movement occupied Taksim Square in the center of the city for several months in sufficient numbers that the government was unable to reclaim the area. Especially in Istanbul and the communities around universities, Turkey continues to have a significant population opposed to information controls. In this population, use of proxies and VPNs is widespread, and many users have come to terms with maintaining anonymous or pseudonymous identities online [159, 5].

2.3.2 China

China has one of the longest histories of online information controls. China has faced few challenges to government power online, but it has leveraged its control of the Internet to influence the success of consumer and business services. In contrast to Turkey, the path of censorship in China is not of a growing restriction on speech, but rather a maintained and less intrusive oversight from previous control structures.

China requires an in-country registration process for each domestic web server, dubbed an Internet Content Provider (ICP). It is possible for a foreign company or individual to receive an ICP license, but the process requires Chineselanguage skills and a mailing address in the country. ICP license terms are a key tool for leverage within the Chinese Internet, since renewal of the license requires good behavior. Without a license, it is difficult to find physical hosting providers or cloud services willing to host a service.

One of the most important forms of control that has emerged in China is the relationship between mainland corporate entities and the government. Companies are liable for policing

user generated content and conversations on their platforms. The regulations do not specify how this regulation should be performed, leading to a diverse set of implementations, including client-side blacklists of keywords, server-side filtering of messages, and manual review of conversations [47, 219, 112]. Many companies hire a significant number of individuals to manually enforce the regulations, and several of the large content platforms are known to have an editor position high in the organization whose job is to act as a liaison with the government [185].

The official narrative for censorship in China has evolved over the last decade. The original explanation for online censorship in China was to protect public harmony. This corresponded with the treatment of print media at the time. In addition to content deemed morally reprehensible, notably pornography, the focus was on material challenging the position of the communist party. More recently, the focus has shifted towards national security, justifying filtering as preventing terrorism, radicalization, and technical compromise [18].

There is much more opposition to censorship outside of China than within its borders. The most vocal opposition has come from exiled political movements supported by western governments. Many of the censorship tools most popular within China are sponsored by the Falun Gong movement, which has a stated anti-governmental agenda [175]. The most prominant individual Chinese disident speaking against censorship is Ai WeiWei, an artist now living in Germany. WeiWei took several actions that made him extremely unpopular politically resulting in police brutality and his house arrest [3]. After the Sichuan earthquake of 2008, where WeiWei publicly documented instances of governmental corruption and called for accountability. His choice to leave China is motivated by the safety of his family, and the phenomenon of an external opposition can to some extent be explained by a propensity to 'chase out' the trouble makers.

A visible difference between the censorship regimes of China and Turkey is the lack of opposition to censorship found in China among the educated liberal elite. There are numerous theories for why this may be the case, ranging from the relative isolation caused by the Chinese language, to the clear internal economic benefit created by the isolation, to

the bulk of visible censorship aimed at sources widely seen as destabilizing [128]. One thing is certain: the societal norms in China have led to a different set of goals and fears around the censorship apparatus than is found elsewhere around the world.

2.3.3 Global Visibility

The best view into censorship practices at a global level remains in the hands of analysis efforts like Freedom House and the Open Network Initiative [91, 151]. These efforts provide textual reports written by analysts attempting to characterize the policy behind limitations, and what forms of speech are suppressed by those policies. From this analysis they encode rankings of how severe the censorship is. These rankings provide for easily approachable overviews of how different countries compare.

There are several limitations in these overviews of censorship. For one, the underlying data used to create the analysis and rankings is unavailable, limiting the ability of the research community to analyze policy in dimensions that were not considered in the initial analysis. A second limitation with both projects is their annual update schedule, which creates long delays and generally separates reporting on changes in policy from when they actually happened. A more effective advocacy tool would provide knowledge of changes in policy soon after they happen. A third limitation in these social-science-focused representations of network interference is their focus on country-level policy. In our data, we often find that the technical implementation of policy varies widely between ISPs, and that level of detail is often missing from these reports [194, 84].

There are a few examples of countries where censorship policy is open to public oversight. In Indonesia, the list of content that has been blocked is maintained in an online database maintained by the government [91]. Some entities will explain why content is blocked (generally the category of offense), along with a mechanism for appealing the decision.

An example of visible blocking of content occurs in Estonia. The Parliament passed legislation dictating that online casinos not registered in the country were operating illegally, and they ordered ISPs to deny access to those sites. The list of sites in violation of the law

is published on a government website[5], with updates determined and communicated by the tax board [154].

While these instances of visibility exist, they are very much in the minority. In almost all jurisdictions that censor, there is no ability to learn the list of blocked content or for public oversight of policy changes. This lack of accountability has been highlighted several times by WikiLeaks, which has published the private block lists of several countries and noting content blocked by those lists which did not fall under the initial justifications for creation of the lists [204].

2.4 Circumvention Techniques

To resist the diverse set of information controls that have emerged, an equally diverse set of evasion techniques have developed. Some techniques focus on changing the bytes that are sent between computers so that they are not easily recognized as objectionable. Others focus instead on leveraging classes of valuable traffic that a network is unwilling to block.

Circumvention systems have been split between those which provide free access for anyone (public systems), and those with a registration or other barrier to entry (private systems). Public systems can provide a better user experience, but they also receive the most scrutiny from network adversaries. Private systems have traditionally had a much higher barrier to entry, requiring users to already have friends in the system or pay money to an operator for access.

Public networks like Tor have two characteristic attributes: (a) any client can use any relay to construct a circuit for routing traffic, (b) they rely on a centralized directory system to publish information on current participants. Most one-hop proxies like Ultrasurf, and Freegate use this model, as well as open proxies and commercial VPNs [75].

Public networks have historicly struggled to circumvent censorship because of the need for a logically centralized directory service and because they freely distribute relay addresses

[5] www.emta.ee

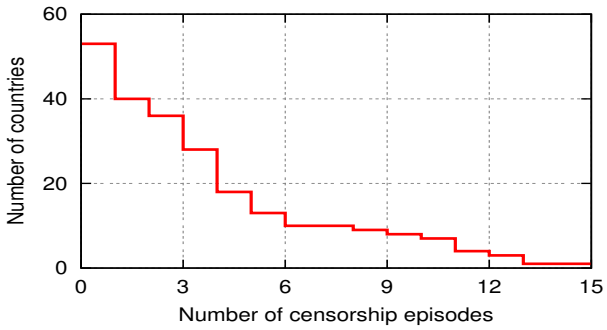


Figure 2.1: Number of observed country-specific censorship episodes against Tor (i.e. blocking Tor when it was previously not blocked) from 2007 to 2012.

to users. For example, Tor provides a few well-known directory servers that return certified lists of relays. As the censor can look up or crawl all relays, these systems are as blockable as the very websites they aim to make accessible.

Blocking of public circumvention tools is only one of the ways that censorship and circumvention are in conflict, as we showed in the previous section. While these tools are engaged in a more active struggle, it is wider ecosystem changes like the increasing prevelence of encrypted web protocols that have caused the most change to online censorship. While most of the forms of interference tools are designed to circumvent remain in place, the ubiquitous HTTPS ecosystem has reduced the effectiveness of keyword-based blocking such that those techniques are rarely seen today.

In the rest of this section, we survey different approaches to circumvention to document the state of the landscape. This understanding of circumvention is used for the systems in Chapters 4 and 5.

2.4.1 Protocol Obfuscation

Protocol obfuscation is one of the most developed techniques for circumventing censorship. Protocol obfuscation refers to the process of changing how a stream of communication looks

on the network, so that it appears innocuous.

Recent efforts in protocol obfuscation started by avoiding IP level discrimination. P2P services, most notably BitTorrent, found that many networks would prevent connections on their official protocol ports. BitTorrent responded to this discrimination by advertising its presence on random ports. Subsequently, ISPs deployed deep-packet inspection, using the initial BitTorrent handshake between users as a signature. Clients responded by introducing encryption (as a form of obfuscation [45]), to make such identification more difficult [125].

The diversity of techniques and their theoretical backing have evolved significantly since these initial efforts. The Tor Project has an effort called Pluggable Transports that attempts to allow mixing and matching of different obfuscation techniques [98]. This approach was in response to a sustained effort to disrupt the protocol by the Chinese government, using both fingerprinting and active attacks against the protocol [160, 207].

At the center of Pluggable Transports is a family of obfuscation schemes called ObfsProxy. The first released version of this obfuscation, obfs2, was a simple protocol in which each side sent seed information, which was then combined to form a shared secret used to encrypt the rest of the communication [104]. This protocol, like that of BitTorrent, could be decrypted by a network middleman that suspected it was in use, and could be be probed to detect if a server would speak the protocol, leading to subsequent development of obfs3 and obfs4. As stated in the motivation for obfs3:

> obfs2 did not use a robust cryptographic key exchange, and the key could be retrieved by any passive adversary who monitored the initial handshake of obfs2.
>
> To defend against this attack, obfs3 negotiates keys using an anonymous Diffie Hellman key exchange. This is done so that a passive adversary would not be able to retrieve the obfs3 session key. (obfs3[105])

While the obfuscating proxy work focused on creating a series of random bytes that would not have a fingerprint, other lines of research attempted to disguise traffic as legitimate

protocols. FTEproxy, and its more flexible successor Marionette, attempt to generate traffic matching a characterization of what "good" traffic should look like [60, 61]. This work follows a large corpus of work showing how circumvention can occur either through or by mimicking other protocols [142, 201, 101]. Circumvention using steganographic techniques take significant amounts of work, but have delivered relatively low performance. Performance is difficult to achieve as these systems incur overhead in sending data to behave like the protocol they are mimicking, slowing the rate at which real data can be sent. Steganographic techniques often compromise realism by diverging from typical protocol traffic patterns to achieve acceptable performance.

More recent obfuscation work has looked at how to hide information from being revealed in the timing and size of data transferred. A recent study found it was possible to identify which website a user visited 85% of the time using only traffic timing and size. The authors propose a padding defense of quantizing traffic into groups of 100 or 500 fully filled packets to protect against this class of attacks [199]. Other attacks have shown how mimicking traffic can be differentiated by failing to compress or otherwise failing to match expected patterns [90].

2.4.2 Collateral Disruption

Collateral disruption takes a complementary approach to obfuscation. Both try to disguise traffic in a way that is difficult to distinguish from 'legitimate' traffic. This makes it hard for an adversary to block the unwanted traffic without also blocking a significant amount of other traffic and causing other good users to be inconvenienced or upset. This principle of maximizing collateral damage or other disruption associated with blocking controversial content has been a primary tactic used for circumvention [58].

Telex was an early research effort focused on making it technically hard for an adversary to block unwanted content without also blocking legitimate traffic. Telex allows participating core Internet routers to divert some traffic to a destination hidden within the encrypted contents of of the flow, rather than to its purported destination [211]. This design was focused

specifically towards a China-like adversary, who would not be able to easily distinguish the good and bad traffic until it left their network.

The mechanism pioneered in Telex remains a proof of concept, but the idea it put forth has blossomed. Telex is difficult to deploy, because to hide traffic bound for a legitimate domain required cooperation either from the domain or from its upstream providers. Instead, a technique known as domain-fronting pioneered by GoAgent has gained widespread adoption using a similar model [71]. Domain fronting notices that, for many shared hosting providers, requests are made to shared servers operated by the hosting provider. For instance, when connecting to a server 'fronted' by CloudFlare, or equivalently using the CloudFront product offered by Amazon, the same server handles connections for many different clients. A modified client can connect to a shared server with a request to an innocuous resource, and then once the secure TLS session has been established ask instead for the otherwise blocked resource.

Domain fronting has been integrated into many projects in the last two years as an effective way to mask traffic within a large quantity of desirable (collateral) traffic. Some use the mechanism only to disguise coordination for telling clients about the current way to access content. Others have used the technique to transport data itself. Most notable in the later category is Great Fire, an organization that republished western news on domain-fronted servers explicitly to make censored content available within China [178]. In announcing their use of the technique, they state:

> For the censors to block our websites and apps, they would have to block all websites and apps being served by CDNs (content delivery networks). The entire blocking of all CDNs would cause a severe disruption of Internet services for everybody in China as CDNs account for over 50% of all global web traffic. The economic damage caused by such a disruption would be major. We believe that the Chinese authorities would not dare block all websites and apps being served by CDNs because they understand the economic implications of this

action. (GreatFire[178])

China did not block the CDNs used by GreatFire in response to their introduction of the technique, but instead took a new tactic. In late March, 2015, the services operated by GreatFire began to suffer a sustained distributed denial of service attack from users outside of China. This attack, dubbed Great Cannon [133], overwhelmed the group's direct site, but more importantly showed an important economic limit to the collateral damage approach. The sustained traffic was sufficient to cost the group tens of thousands of dollars in bandwidth fees - a significant increase in cost the group was not prepared to handle. The Great Cannon has put a damper on the hope of sending all traffic through cloud infrastructure. Despite worries about unsustainable costs, it does continue to offer an important niche. Especially for static content that CDN providers can cache and serve at high loads, it remains a powerful tool for distributing content.

The reality of domain fronting as a tactic against the Chinese firewall has been less rosy than it first appeared. Many CDN networks have felt pressure to disable domain fronting, and they have responded in different ways. CloudFlare has partnered with the local Baidu CDN to allow for better access within China to served sites, with the compromise that they will comply with Chinese regulations and not provide access to content that is deemed objectionable [44]. Other CDNs including Akamai have also struck deals in order to expand their businesses into the Chinese market. Other providers, most notably Google which was already in an adversarial relation with the Chinese government are largely blocked outright [171].

Another reality of domain fronting is that it relies on hiding within a service provided by a large infrastructure provider and will incur associated costs. The running instances of Meek, the implementation of domain fronting used by Tor, incur several thousand dollars of bandwidth fees per month to support 10,000 users [69]. This cost aspect was also highlighted by the Chinese Great Cannon (described in 2.2.4) [179].

Domain fronting remains valuable in other circumstances. For smaller countries reliant

on international services, the current shared-infrastructure model makes it difficult to differentiate traffic without CDN cooperation. There are real concerns with economic denial of service against the technique. Thus, the use of domain fronting for initial coordination, leading to alternate mechanisms for the main body of communication can mitigate those costs.

2.4.3 IP Diversity

IP addresses (as described in 2.2.1) are frequently a component of how censorship decisions are made. The Tor anonymity system is frequently censored, and in many cases these policies are based on the list of participating relays published by the system. Tor's original goal as a system was not to provide access, but rather only anonymity. The presence of many users eager but unable to use the software has focused developer effort on addressing the problem of access for the system.

To quantify this censorship, we analyzed availability provided by the Tor network. Using data the Tor project has maintained from usage of its network in 243 countries from August 2007 to December 2012 [159], we aggregated the number of clients that connected to each of the Tor directory servers into two week periods by country. We compared these totals with the preceding period. Finally, these ratios were normalized to the total number of Tor users around the world for the two corresponding periods. The two week period acts as a low pass filter, evening out short term variations in usage. By normalizing against the global user count, the analysis also accounts for overall trends in Tor usage.

We analyze this data by defining a censorship episode as an event where the Tor usage in a country where Tor is normally unblocked drops more than four standard deviations below expectation. Figure 2.1 illustrates the results from this analysis. Tor experienced at least one censorship episode in 53 countries (out of 243), with repeated disruptions in many of those countries.

Tor's primary approach for providing access to its overlay, for networks which attempt to make the system inaccessible, is through additional points of connection which are not

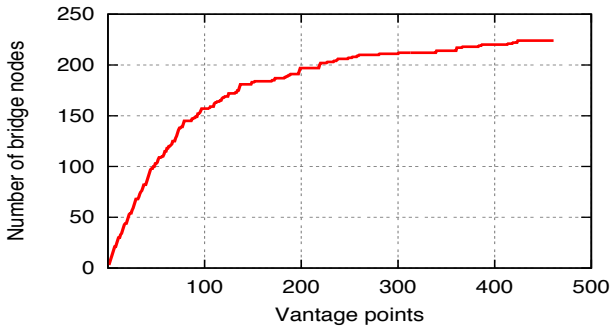


Figure 2.2: Number of discovered Tor bridge nodes versus the number of PlanetLab vantage points.

publicly advertised. Instead these 'bridge' IP addresses are selectively provided to users based on a web requests or via email. Selective dissemination of this form is not effective at preventing an adversary from crawling; in practice bridges are available only in countries that do not actively attempt to prevent access to Tor.

To quantify the exposure of bridges, we looked at how many were visible to us in an adversarial position. Figure 2.2 shows the number of bridges exposed to the different university institutions running PlanetLab. We found that by requesting bridge nodes from these locations, we were able to discover the IP addresses of 240 bridges. This accounts for almost all of the active bridges distributed through that channel during the measurement period. One can easily imagine a censor using similar crawling techniques to block these distribution channel, and indeed the crawling attacks are seen in practice [160].

In attempting to provide bridges to legitimate users, while hiding the service from adversaries, Tor has made several technical advances. The first is to provide a symmetric 'shared secret' to clients at the same time as an IP. A node in the system will not respond unless the client proves it knows that key. This choice was a reaction to active attacks where an adversary suspicious of a connection would try to establish its own connection to a suspected participant to learn if it was participating in the protocol [207].

To increase their ability to distribute non-public IP addresses, the Tor overlay needed

to have a significant number of IP addresses running their software that were not publicly advertising that fact. This need was initially addressed by the involved developers directly, but as more IPs were needed it was extended into a public campaign[6]. The primary call-to-action was for volunteers to register for free low-capacity virtual servers from Amazon and other providers, and to use those services as small, non-public Tor relays. Appropriating a set of IPs from within a lager cloud service in this way provides some of the benefits of cloud fronting, since the IP block can't be fully censored without also affecting many legitimate services using the same cloud providers.

An alternative approach to providing circumvention is employed by VPNgate, a service aimed at circumvention in China through a volunteer network of VPN providers and a central aggregation website and client [147]. The network attempts to avoid discovery through server coordination; clients seen probing servers are blacklisted to preven them from learning about the full network. The actual success of the network as a low-latency and high-performance system came from the fact that many of its servers are hosted in academic networks in nearby countries[7]. Many countries, including China, have purpose-built academic network infrastructure with different behavior from the general commercial network. This university exceptionalism meant that less latency is incurred when sending packets from China to a foreign academic network than to most other foreign IP addresses, since the path will generally first traverse the Chinese academic network and then to an international academic network on a relatively uncongested peering link.

A promising approach is to relay traffic through many different home user networks, so that the set of active participants is not stable. The first successful demonstration of this concept was flashproxy, a pluggable transport for Tor. With flashproxy, users do not connect directly to a Tor bridge; instead, they wait to receive a connection from a website visitor who was instructed to connect to them by a coordination server [70]. These relays do not

[6]The majority of publicity was centered around the EFF's eff.org/torchallenge initiative.

[7]The Chinese educational network, CERNET, has a different firewall policy than the commercial networks [66]. VPNgate contains many academic nodes with better connectivity characteristics [16]

run the full Tor software. Rather, participating websites temporarily co-opt the visitors computer to act as a relay. This strategy of using website visitors as a source of diverse IP addresses is powerful because the visitors do not even know they are participating. This issue is at the center of a large ethical debate about how much a user computer should be able to be co-opted. The creators of flashproxy argue that filtering users due to their participation would impose considerable collateral damage. While flashproxy demonstrated a proof-of-concept of this strategy, in practice it faced several limitations that prevented it from gaining significant popularity. First, it used the Adobe Flash plugin to initiate connections from browsers, which limited it to the subset of web browsers configured to allow Flash without user interaction. More problematically, most web pages loading the system were visited only transiently by most users, meaning that the vast majority of connections were not able to transfer a significant amount of data before disconnecting.

While the Flash plugin continues to loose popularity, the WebRTC browser standard has replaced it as a mechanism for establishing temporary connections from a web visitor's browser. To this end, development is underway on a system called Snowflake, which uses the same mechanism as flashproxy, but using a WebRTC connection. Snowflake faces the same challenge of visitor churn as flashproxy; most users will only participate for a short time.

The approaches to IP diversity discussed so far extend a system to use a lot of different IP addresses, or IP addresses which are scattered among high value services. A related opportunity that has not been as deeply explored is to pretend to have access to additional IP addresses, even when the system does not actually have any relation to those machines. VPNgate makes an attempt at this by providing the IP addresses of critical services within its list of participants, so that the list itself cannot be directly used by an adversary without manually checking whether each entry is actually a participant. This use of decoy IP addresses improves the plausible deniability of true participants since they are more difficult to distinguish from other arbitrary addresses.

Another opportunity explored in CensorSpoofer is to use IP spoofing to send packets that appear to the rest of the network path to be from a different source address [198].

According to the MIT monitoring of network restrictions, roughly 25% of Internet hosts have the capability to spoof packets [27]. Using IP spoofing for censorship circumvention would require a protocol where both legitimate and spoofed IPs could send traffic in one direction to a client without the expectation that the client would ever need to respond on the same connection - in order to make it harder for the client, which could be an adversary, to distinguish whether the connection is real or from a spoofed address.

2.4.4 Social Trust

Unlike open networks, one can imagine using the pre-existing trust between users to establish connections. These "social network overlays" have been explored as a way to improve security and prevent blocking by adversaries. Examples include the Ostra [140] email service and the OneSwarm [99] system for anonymous P2P file-sharing. Social overlays route user traffic to "exit nodes", nodes located in non-censored domains willing to make connections on behalf of other users, in order to provide access to blocked websites. Social overlays are an attractive option for resilient service because the network can be formed in a completely decentralized fashion. As each user joins the overlay by connecting to his explicitly trusted peers, no single user (including the censor) can discover the identities of more than a few participants.

Unfortunately, availability in social overlays tends to suffer from sparse connectedness. We measured the graph properties of YouTube, Flickr and Foursquare using datasets collected by [139, 170]. These networks are likely to be at least as dense as a network targeting censorship resistance, where users may be hesitant to advertise their participation. Nevertheless, most nodes in the measured networks have at most a handful of links to other peers and a large number of users have only one social link. This is particularly problematic in a P2P setting where users, essential for connectivity, may not be available all of the time.

2.5 Measurement

By 2003, it had became clear that nation-state censorship was on the rise. One of the initial calls-to-arms came from Zittrain at Harvard. He published an *Empirical Analysis of Internet*

Filtering in China [220], a list of keywords blocked by the Great Firewall. This hand-curated list was later expanded by ConceptDoppler [46], a system designed to validate the keyword censorship observed in China and to automatically discover new terms as they were blocked.

At around the same time, national watchdogs were taking stock of their own ISP and governmental policies around online censorship. This move to document and measure censorship occurred both in diasporas assessing the state of Internet connections in their native countries, and in western countries like Germany and the UK. This wave of concern is reflected in documentation efforts like the Open Net Initiative [151] and the Freedom House Freedom on the Net report [91].

From these roots, the efforts to measure and expose censorship have grown in response to new censorship techniques and to increased enforcement. A major limitation with this reactionary approach is that it fails to alert the community to changes in censorship policy or to provide comparable baselines or histories. Work in recent years has attempted to systematize measurements and develop new models that can be run more sustainably and consistently.

What to Measure: Determining domains of interest is by itself a tough problem. There are many billions of DNS records in use on the Internet [14, 68], and there are obvious deficiencies with the coverage or representativeness of lists of top sites. Since block lists are an extremely small subset of all domains, random sampling would likely be ineffective. The choice of domains is important, because an incomplete list of domains can lead to an incomplete picture of censorship. For example, in 2013 China provided an incorrect security certificate for `github.com` [13]. Unless a measurement tool was monitoring that specific domain for that specific attack, it would not have noticed the change in policy.

In light of this problem, researchers have converged on two approaches. The first approach is to use a list of globally or per-country 'popular' domains. Companies, notably Alexa and SimilarWeb, release these lists – which are sourced by user browsing data [7]. In the case of Alexa, a daily ordered list of the current 'most popular' 1 million domains is released for free

based on data they collect from users running their browser plugin. The second approach is to use hand-curated lists focused on domains that may be sensitive. Much of the effort in this area is centered around a 'test-list' repository containing lists of domains potentially sensitive in 64 countries maintained by The Citizen Lab and others [120]. This effort works with civil society groups to find active websites popular in the local area focused on a range of social issues that have the potential to be blocked or are already blocked. In practice updates to this list have been sporadic, resulting in a patchwork with some regions having currently sensitive and well categorized domain lists while other areas have a small number or no-longer active domains listed.

While early approaches to identifying keyword censorship used linguistic and natural language processing techniques to automatically detect new terms that could be sensitive, this technique has not been extended to automatically determining new domains of interest [46].

In addition to domains, there's a growing attempt to measure censorship within individual protocols. The Asia Chats research project documents censorship lists found within chat applications, focusing on those embedded directly within clients rather than in the network [47]. For service level censorship, projects like freeweibo.com crawl social networks to look for content that is removed shortly after being posted. At a protocol level, OONI includes tests for many proxy and circumvention protocols to provide insight into the accessibility of different circumvention applications [72].

Another source of measurements comes from the self-reported data provided by censorship circumvention service operators. This data can be used to understand the status of circumvention, for instance using the metrics provided by The Tor Project [159]. It can also be used to determine the accessibility of general targets of censorship, which can be found both in direct reports by operators [108] and in the transparency reports of companies like Google [82].

How to Measure: Researchers have used a wide variety of mechanisms to measure network censorship. While many techniques involve participants or at least cooperating software

within target networks [72, 96, 165] there have also been successful attempts to measure censorship from external vantage points [46, 193]. Several efforts have sent physical devices to participants around the world, which can then be controlled by researchers [96, 165, 183]. Other efforts release software either directly focused on censorship or more generally helping users understand their network performance [72, 116].

DNS has been a measurement focus, largely because it is a commonly manipulated and unsecured protocol. DNS can be measured from an external vantage point, since DNS servers on other networks are often configured to respond to all clients. This technique was proposed for censorship measurement [209] as early as 2006. What we continue to lack is a system which is able to sustainably measure and act as a data repository for these measurements across both countries and time. Ripe Atlas [165] offers shared access to its distributed deployments of probes, but limits the types of measurements and rate-limits measurements such that regular probing of many domains is infeasible.

There are also measurements of interference at lower levels of the protocol stack. Measurement of anomalies in the IP header, like the Time to live (TTL) field and the identification number (IPID) have been used to identify proxies and other devices designed for censorship and surveillance [206]. Side-channels[8] in the TCP fragmentation buffer have allowed researchers to measure the connectivity between remote hosts [65, 133]. These techniques have been powerful in providing deeper analysis about 'where' in the network censorship is occurring [26].

Multiple efforts have provided web interfaces soliciting crowd-sourced measurements from users to learn about blocked content organically. The first major pioneer of this technique was Herdict, a project of the Berkman Center at Harvard [89]. Herdict provides a suggested list of websites that it loads in a frame, and asks visitors to click whether the page appears 'accessible'. From multiple reports in a country, the site then calculates what percentage of users have recently reported problems with monitored domains. Other groups have focused

[8]The term side channels refers to information 'leaks' not in a protocol itself, but from side effects. In this case, probing a remote buffer can reveal other active connections.

Resolver	Response	Behavior
USA (8.8.8.8)	199.59.149.198	Twitter
Russia (77.88.8.8)	199.16.156.102	Twitter
China (180.76.76.76)	159.106.121.75	Failure

Table 2.1: Resolutions of Twitter.com by different resolvers

the technique on specific countries, often in tandem with specific servers to monitor the crowdsourced list of domains [178, 84]. These techniques must contend with false reports from users who either intentionally or through lack of technical understanding misinterpret the measurement process.

Determining Site Presence: While determining which sites are potentially blocked is hard, determining whether a given IP is a valid host for a site can be even harder. When ISPs block websites by redirecting them to a block page, the result is hard to distinguish from the normal behavior of a CDN node for that geographical region. Consider the example of twitter.com. As shown in Table 2.1, the domain resolves to different IPs in the US, Russia, and China. A naive CDN mapping would conclude that there are likely points of presence in all three countries, while a naive interference measurement might conclude interference in both China or Russia, or might give up due to the diversity of IPs returned. In reality, the Russian IP maps to a Twitter CDN node, while the Chinese resolution is due to interference.

DNS has been used to learn this ground truth through limited heuristics and at limited scale. In their investigation of CDNs in 2008, Huang et. al [92] arrive at a list of 280,000 open DNS resolvers, and use them to map the Akamai CDN. They create their list of resolvers starting from DNS servers observed by Microsoft video clients, rather than direct probing. Specific CDNs like Google have been characterized through the use of EDNS queries to simulate the presence of geographically diverse clients [35], but this is only possible for a small subset of resolvers which support EDNS for redirection. Research focusing on censorship, like

the analysis of Open Network Observatory data [80], have used the diversity of autonomous systems (ASes) to determine if IPs are valid for a domain, but do not explicitly consider CDN behavior.

There are also many commercial sites which offer traffic information for web sites. We know that some of this data is crowd-sourced through browser plugins, while other portions come from automatic robot crawling. For instance, the Alexa rankings are based off of a browser plugin which monitors the browsing habits of a small number of participating users. Some sites also show which sites run on identical IP addresses [94]. In practice we find that these systems appear to do direct lookups of IPs, since geographical distribution is not made visible. They also do not appear to do significant identification of CDN IP spaces, since CDN'ed sites are not fully aggregated.

Determining Abnormal Behavior: Categorizing responses as normal or abnormal have typically been performed through the use of heuristics in how the response may deviate from expected behavior. This is true for both determining trust in a DNS response, and determining if a given connection is working as expected. These heuristics include metadata like the ASN and reverse PTR record of the IP [80], behavior of HTTP queries to the server [102], and considering the aggregate prevalence of a given response [72]. More recent work has explored the use of aggregate statistical behavior to determine when network level behavior has changed [210]. These techniques provide valuable direction for Satellite, though there is not yet a comprehensive set of best practices for determining self-consistency and anomalies in our data set.

2.6 Summary

Network censorship is an evolving phenomenon that is only beginning to be understood technically. The active measurement projects in the preceding section are only a few years old, and none have yet reached their claimed goals of accessible insights into online censorship. New techniques of control (Great Cannon) and circumvention (Collateral Damage, Packet

Spoofing as a source of IP diversity) are regularly introduced. Most importantly though, the phenomenon of online censorship is reaching a point where we know how to talk about it.

Reaching a point where we understand much-less know how to circumvent censorship on a wide scale is by no means a solved problem. In Chapter 6, we consider the larger trends at work in more depth. The network censor is an extermely powerful adversary and the desires for censorship are not going away. Instead, we can expect only that they will evolve and that success in this field will only mean that censorship is implemented at other points in the system, rather than being either visible or circumventable.

One point of optimism is in the commitment to openness and transparency in the measurement community. Led by the OONI project, several academic efforts have publicly released their data sets. This helps new efforts get started, increase the value of the data, and hold the community to higher standards of reproducibility.

We have reason to be expect a diminished capacity for network censorship more broadly as well. Major efforts are underway to encrypt as much of Internet traffic as possible, since lack of confidentiality has proven to be a significant vector of compromise [83, 168]. Encryption is a technique we know makes it harder to differentiate traffic and for censors to function effectively. Along with more capable client devices and the growing prevalence of shared infrastructure, this will limit the effectiveness of active network interference as a form of information control.

Chapter 3

SATELLITE: MEASURING ACCESS

After several generations of measurement platforms, it remains difficult to identify the extent to which web access is censored. This lack of understanding is reflected in the questions we cannot easily answer: Which countries have servers operated by Google or Microsoft? Which websites have degraded availability due to network interference? Which sites are powered by various content distribution networks (CDNs) such as Akamai or CloudFlare? Which ISPs run caching proxies or other stateful middleboxes? Without answers to these questions, we can't give good advice to users or hope to build software systems which connect to blocked content without user guidance.

While we have some understanding of what measurements can address these problems, there is no existing data set or measurement platform that holds the answers. In fact, there are many challenges both in collecting the measurement data and analyzing it to characterize the current state of web censorship. First, we would need measurements from globally distributed vantage points in order to characterize global website accessibility. Second, since the deployment and accessibility characteristics vary significantly across websites and time, the data-sets should be collected at a fine-grained and timely manner. Third, the analysis of how websites employ CDNs and the identification of network interference are interrelated and have to be tackled jointly in order to obtain an accurate characterization. For example, when ISPs block websites by redirecting them to a block page, that server could be misconstrued as a CDN node for that geographical region. Conversely, websites served through globally distributed CDNs can be confused with willful redirection of traffic by a local ISP. We need to determine the expected IPs of CDN deployments in order to characterize the abnormalities that are interference.

We make the observation that it is both advantageous and necessary to study these two issues together. This need arises from the fact that both CDN routing and network interference often occur during the domain resolution phase of a connection. In addition, the two seemingly unrelated problems share similar challenges and require similar measurement data to resolve. Crucially, studying one of these problems without accounting for the other will lead to biased results.

We address the need for timely global measurements with a system that uses a single end-host to collect DNS resolutions from a large number of globally-distributed and open DNS resolvers. Instead of pursuing crowd-sourced deployments or analyzing limited snapshots of data obtained from ISPs in privileged positions, we instead focus on what is possible from active measurements by a single end-host. If successful, this reduces the barrier for organizations to run their own independent measurements. While measurements from a single host may be biased compared to those of a distributed system, the validation challenges are similar, since in both cases no individual point of collection can be trusted.

Satellite is a fully open project consisting of the code for data collection and analysis, a growing year-long repository of collected data, and derived views of site structure and interference. Satellite was consciously built as an open system to minimize the trust that needs to be placed in the system or its operators. Satellite is designed to be operated by several independent organizations to allow for independent auditing and confirmation of collected data. Through this strategy, we hope to reach a point where others can trust collected data without the need to replicate the collection work. This approach also improves our confidence in the sustainability of the project, and our ability to amass a longitudinal data set of changing Internet behavior.

Through interpretation of Satellite data, we are able to correlate the addresses of domains across ISPs and learn the customer pools of CDNs. Looking at the pools of IPs, we can learn the points of presence of CDNs and which CDNs have business relationships with which ISPs. By looking at which locations resolve to which points of presence we can understand the geographic areas served by different points of presence. By tracing the patterns of

divergence from clusters, we are able to separate the effects of network interference from confounding site distribution factors.

The major contributions of Satellite are:

- A single-node measurement system for monitoring global trends in network interference and CDN deployment without user cooperation.

- An algorithm for the joint analysis of network censorship and CDN points of presence from measurements of domain resolutions.

- Data on the reachability and routes to 10,000 popular domains over the last two years.

3.1 Design & Implementation

The implementation of Satellite is motivated by a number of explicit design goals:

- **External Data Collection**: We want the system to measure interference without in-country resources. This avoids the need to recruit or worry about the safety of volunteers, while still providing high coverage.

- **Continuous Measurement**: We want the system to be able to quickly notice changes in network interference.

- **Transparent and Ethical Measurements**: We want the system to be transparent, so that others can easily trust and make use of collected data. We aim to minimize harm to DNS server operators from collected data.

- **Joint analysis of CDN deployments and Network Interference**: We want a system which simultaneously measures shared infrastructure and interference of web access, since the two are tightly intertwined.

3.1.1 System Overview

The Satellite system is arranged as a pipeline which collects and analyzes data. It is run as a weekly job that schedules data collection and performs initial aggregation, analysis and archiving of each data set. The implementation details of the pipeline are described in more detail in Section 3.2. At a high level, Satellite is structured into the following discrete tasks:

Identifying DNS resolvers by scanning the Internet. We detect active, open, long-lived DNS resolvers through active probing.

Assembling a target domain list by expanding a list of popular domains to ensure CDN coverage.

Performing active DNS measurements where candidate domains are measured against discovered resolvers.

Collection of supplemental data to provide organization metadata and geolocation hints.

Aggregation of DNS resolutions by combining records at the AS level to allow for efficient processing in subsequent analysis.

Mapping of CDNs versus network interference through the calculation of fixed-points in clusters of domains believed to use shared infrastructure.

Export of measurement results by publishing visualizations and data sets with footprints of CDNs and significant observed anomalies.

3.1.2 Ethics of Collection

Our measurements prompt machines in remote networks to resolve domains on our behalf. This traffic to remote networks may result in unintended consequences to these relays, and as such we do our best to minimize harm in keeping with best practices [57].

Open DNS resolvers are a well known phenomenon, and lists of active resolvers can be downloaded without the overhead we incur in scanning. We find that the act of scanning the IPv4 address space to find active resolvers does generate abuse complaints from network operators. By maintaining a blacklist of networks which have requested de-listing (less than

0.5% of the address space), we have not received any complaints related to our scanning or subsequent resolutions in the last three months. Some operators have asked us to keep their network spaces private, which prevents us from releasing this list publicly. Others running the system should expect to recreate a similar list. We have never received a complaint from overloading a DNS resolver with queries for our tracked domains.

We abide by the seven harm mitigation principles for conducing Internet-wide scanning outlined by the zmap project [59] and consistent with [130, 181]. In particular, we (a) coordinated with the network administrators at our university in handling complaints, (b) ensured we do not overload the outbound network, (c) host a web page explaining the measurements with an opt-out procedure, and have clear reverse DNS entries assigned to the measurement machine, (d) clearly communicate the purpose of measurements in all communications, (e) honor any opt-out requests we receive, (f) make queries no more than once per minute, and spread network activity out to accomplish needed data collection over a full one-week period, and (g) spread the traffic over both time and source addresses allocated to our measurement machine.

To get a better sense of the impact our queries have on resolvers, we operated an open DNS resolver. In a 1 week period after running for 1 month, the resolver answered over one million queries, including 800,000 queries for domains in the Alexa top 10,000 list. Satellite made only 1,000 of these requests.

We have additionally adopted a policy of only probing DNS servers seen running for more than one month to reduce the potential of sending queries to transient resolvers. This reduces our resolver list by 16%[1]. Measurements in IP churn indicate that the bulk of dynamic IPs turn over to subsequent users on the order of hours to days, making it unlikely that our measurements target residential users [213].

[1]Specifically comparing the live resolvers discovered between March 20th and April 20th, 2015.

3.2 Implementation

We next discuss the implementation of Satellite and the 7 discrete tasks outlined in the previous section.

3.2.1 Identifying DNS resolvers

Our measurements are based on gathering data on how domains behave for different clients around the world. There are several options available for this type of collection. Traditionally, researchers have used cooperating hosts in a variety of networks [130, 151]. More recently, the EDNS extension allows clients to indicate that they are asking for a response that will be used by someone in a specific geographic area [196, 35]. Very few domain name servers support EDNS, but we can take advantage of the behavior the mechanism is designed to fix. By making requests to many resolvers, we can learn the different points of presence for target domains. For instance, the `8.8.8.8` resolver is operated by Google and provides a US-centric view of the world, while `180.76.76.76`, "BaiduDNS", provides a Chinese centric view.

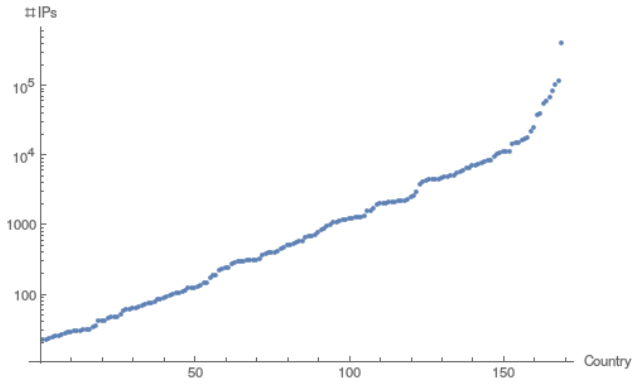


Figure 3.1: Unique IP addresses serving DNS discovered in each country on a log scale. We find 169 countries hosting DNS resolvers in more than 20 class-C networks.

We enumerate DNS resolvers by probing the IPv4 address space with zmap [59]. The

Open Resolver Project [145] is a service measuring the potential for reflected denial of service attacks through DNS. It monitors 32 million open DNS servers, but does not share the IPs of discovered servers. We independently discover 12 million servers which respond to requests with a well-formed response. Of these, 7 million servers across 1.5 million class-C (/24) networks offer recursive resolution and give a correct IP address when asked to resolve our measurement server. These servers span 20,000 ASes and 169 countries, each with at least 20 servers in separate Class-C networks. Figure 3.1 shows how many resolvers were found in each country.

3.2.2 Assembly of domain list

To understand how sites behave, we must first collect a set of sites to monitor. It is unrealistic to monitor all domains on the Internet. Without a priori knowledge of CDNs and their expected IPs around the world, we need to monitor a representative set of domains to serve as a baseline.

We select the top 10,000 worldwide domains as measured by Alexa[7]. All of these domains receive high amounts of traffic. The least popular, `qualcomm.com`, is estimated to receive over 10,000 visitors per day. While not a perfect list, 10,000 domains contains the diversity needed to discover important CDNs. Looking at the smaller Alexa top 1000 domain subset, we find only a quarter of the CDNs found in the broader list. For services like CloudFlare which partition their IP space, our clustering algorithm would be overly cautious without access to an appropriately diverse list of domains.

We make HTTP requests to each domain. Many bare domains (e.g. `expedia.com`) redirect to a prefixed domain (e.g. `www.expedia.com`) served on different infrastructure. When we detect such redirections, we include both the bare and prefixed domains in subsequent steps. We observe these redirects in roughly one fourth of monitored domains.

3.2.3 Active DNS measurement

Our goal in Satellite is to provide a tool for longitudinal mapping of the accessibility and distribution of web entities. To quickly detect updates and policy changes, we must constrain the amount of time we are willing to allow probing to run. Given the goal of weekly measurements of 10,000 domains from a single host, we request each domain from 1/10th (or roughly 150,000) of discovered DNS vantage points, maintaining geographic diversity while spreading network load across available hosts. This results in a measurement period of roughly 48 hours at a probe rate of 50,000 packets per second. We find our measurement machine to be CPU limited at about 100,000 packets per second. Unlike a typical zmap scan, our resolution probes have a high response rate.

Our probing is accomplished by extending zmap with a custom 'udp_multi' mode, where hosts are sent one of several packets. The packet sent is chosen based on the destination IP address only, resulting in a stable set of requests across measurement sessions — the same resolvers will receive the same queries each week. This approach was chosen for efficiency. Multiple scanning processes and accompanying pcap filters increase CPU load and result in dropped packets.

The result of a 48 hour collection process is a 350GB directory containing tuples of resolver IPs, queried domain, time-stamp, and received UDP response. We record the full packet responses we receive, under the assumption that in the future we may find other fields of the DNS responses to be of interest. The raw format of base-64 encoded packets is extremely verbose, but since the response packets for each domain are largely the same, a full run can be compressed to 20GB, or roughly 1 TB per year.

3.2.4 Supplemental data collection

There are several pieces of supplemental data that are needed to understand the measurement data. For IP addresses of interest, we collect the reverse PTR and WHOIS organization information to improve our ability to map IPs back to their controlling organizations. For

these organizations and the IPs they control, we also collect supplemental information to understand their geographic points of presence.

IP Metadata We retrieve meta-data on resolved addresses to identify what organizations they belong to and whether two addresses are likely to be equivalent. Specifically, we collect the reverse PTR records for IP addresses and the WHOIS organization entry controlling the address. Reverse PTR records are contained in the 'in-addr.arpa.' pseudo-TLD in the DNS hierarchy. They are maintained by the organizations controlling the IP address and often provide a canonical name when the IP belongs to a known service. The WHOIS database is a database of IP ownership maintained by IANA and its delegates that contains organizational responsibility, in the form of technical and abuse contacts, for IP addresses.

We perform direct lookups for both the PTR and WHOIS organizational contacts for all distinct IP addresses resolved. We then perform a clustering of each data set: All IPs with the same WHOIS organization are clustered into a WHOIS cluster, and all IPs with consistent PTR records are clustered together. To cluster PTR records, we use a simple heuristic: if all but the final dot-separated section of the returned records are equal, we put the IPs in the same cluster. For instance, a west coast resolution of `apple.com` has the PTR record of `a23-200-221-15.deploy.static.akamaitechnologies.com`, while an east coast resolver sees `a23-193-190-30.deploy.static.akamaitechnologies.com`. Since both cases end with `deploy.static.akamaitechnologies.com`, they are clustered together as part of the same entity.

Geolocation During our collection and aggregation process we maintain a network, rather than geographical, view of the data. We prefer aggregation at a Class-C address level, which reduces calculations without losing precision or mixing IPs owned by different entities. Our other form of aggregation is on the AS level, to represent the aggregations of IPs which will see a similar view of the rest of the Internet. The AS which 'owns' an IP range is responsible for managing abuse and routing of packets for those IPs. As such, even when

a sub-range is delegated, we assume the full AS experiences a consistent routing policy. This is a simplification: for example, the Comcast AS contains clients on both of the east and west coast of the US, and these will be sent to different data centers. Our use of AS aggregation will consider these results as a single combined data point. Likewise, the Google and Edgecast systems operate servers in many countries. When addresses in these ASes are used as resolvers, we consider them to be in the closest location to our measurement machine, the US.

For the visualization of infrastructure locations we also have to associate IPs with geographical locations. For this, we use three data sources: the country of registration for the *whois* point of contact (AS location), the MaxMind [136] country-level database (IP location), and the list of anycast prefixes from Cicalese et. al. [40]. When MaxMind geolocates different IPs within an AS to multiple countries, we use that list. Otherwise, we use the country of registration. Since MaxMind cannot handle geographic diversity hidden by anycasting, we explicitly geolocate the points of presence of anycasting IPs and use the closest point to a given resolver.

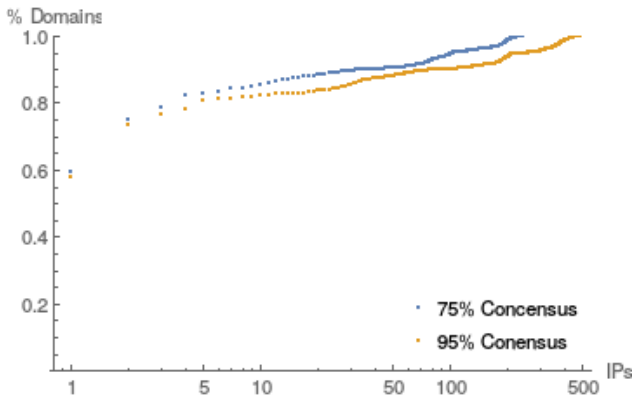


Figure 3.2: A CDF of the number of IP addresses hosting different domains at two thresholds for dominant addresses. For 59% of domains, one IP address accounts for almost all resolutions, and for 80% of domains, 10 IP addresses account for almost all resolutions.

We find that 1% of distinct IPs found by Satellite are anycast addresses. To estimate

the points of presence of these IPs, we measure latency from a range of vantage points, as in [109], resolving topology with [130]. In the future, we hope to learn these latencies through the DNS requests we already make using the technique in [87]. We find that since the CDNs we are identifying are highly distributed, we end up with observations which are either very small latencies indicating a point of presence near the vantage point, or are large enough to not impose additional constraints.

While these geographic heuristics are not infallible, they are largely accurate at the country level [174, 158]. As such, they provide a grounding for initial data exploration. When considering specific interference or deployment situations, it remains important to identify the relevant subsets of data. For instance, when we consider Iran in Figure 3.9b, we manually limit our analysis to ASes of known ISPs in the country.

3.2.5 Aggregation of DNS resolutions

To support interactive exploration and analysis of collected data, Satellite automatically aggregates the observed responses of each weekly collection. This automatic processing also materializes several views of the aggregated data which are used in the subsequent analysis.

This automatic process attempts to parse each received packet as a DNS response, validates that it is well-formed, and records the IP addresses returned. We tabulate these values for each resolver AS and domain. The resulting mapping is roughly 3 GB and is used as the basis of subsequent processing. The 100-fold reduction comes from stripping the formatting and other fields of DNS responses and from aggregating responses by resolver AS. Scanning this file to calculate basic statistics takes under five minutes on a single 2.5GHZ core, and the format lends itself to parallel execution when more complex tasks are needed.

In addition to initial aggregation, we automatically build lookup tables for the set of IPs which have been resolved for each domain and the total set of IPs seen. We also calculate the set of domains associated with each IP to facilitate reverse lookups of other domains potentially co-hosted on an IP. On a recent execution of Satellite, we saw a total of 5,337,315 distinct IPs resolved, located within 6,742 distinct ASes.

The domain resolutions we collect provide insight into the inner workings of popular websites. In Figure 3.2, we show the diversity in the responses for each domain from different DNS resolvers. If almost all responses return the same IP address, we can make the inference that the dominant IP is the canonical server for the domain. In other words, the domain is 'single homed'. In our monitored domains, we see this behavior in roughly 60% of domains, the far left data points in the graph. Near left in Figure 3.2 are domains which use simple load balancing schemes. Roughly 80% of domains have four or fewer 'dominant' IPs. This figure does not reflect the use of anycast IP addresses. The far right on the figure indicate domains which use geographically distributed infrastructure. These require more complex analysis to determine whether individual resolutions are correct or manipulated. For example, we record over 500 IP ranges for the `google.com` cluster, and over two hundred for Akamai hosted domains like `www.latimes.com`.

3.2.6 Mapping of CDNs versus Network Interference

We know that many CDNs resolve domains to different IP addresses based on the client location. While the diversity of IPs makes it more difficult to understand which are 'unexpected' deviations, the primary insight we can use is that in many cases these CDN infrastructures are shared by many websites. The set of websites on a shared infrastructure is often independent of the set of websites targeted by network interference.

Consider the case of `thepiratebay.se` and `strawpoll.me` hosted on Cloudflare. From a US location, like the DNS resolver operated within UC Berkeley (AS25), both domains resolve to IPs in the `141.101.118/24` subnet. However, across many networks in Iran (for instance AS50810), the first resolves to `10.10.34.36`, an internal LAN address, while the second continues to resolve to Cloudflare owned IPs.

To automate this form of detection, we automatically find cliques of domains hosted on the same infrastructure, and use the combined resolutions of those domains to map the IPs of the underlying infrastructure. Using multiple domains helps overcome the randomness present in individual domain resolutions and identifies when one domain behaves strangely

in a specific geographic region. We do not use IP metadata to map provider infrastructure, but rather the sets of IPs (potentially across providers) that form the footprints of popular domains.

To process the data, we perform a joint analysis using the algorithm in Figure 3.3 (described in text below). Then, we use the stable values from that computation to extract cliques and deviations, which represent shared infrastructure and interference respectively.

Joint Analysis Algorithm Given a bipartite graph linking IP addresses and domains, our goal is to separate the graph into two components: 'real infrastructure', and 'interference'. An intuition of how to think of this separation is shown in Figure 3.4. To find this separation, we compute two quantities: A similarity metric `DomainSimilarity`, for how close two domains are, and a trust metric `IPTrust`, for how likely an IP is to be an authentic resolution for a given domain. In Figure 3.4a, *a.com* and *b.com* have a high `DomainSimilarity`, since they resolve to the same IPs. In Figure 3.4b, IP `210.211.21.90` has a low `IPTrust` score, since many otherwise unrelated domains resolve to it. This process is similar to the HITS algorithm for finding "authoritative" sources for pages [110].

The `DomainSimilarity` metric specifically represents the fraction of the time that two domains resolve to the same IP addresses. We use the different IP addresses as independent dimensions in which the resolutions of each domain can be represented as a vector. The distance between Domains is then the cosine distance between the two resolution vectors. The `DomainSimilarity` quantifies shared infrastructure; one IP address that serves many domains will cause the similarity between all of those domains to increase.

The `IPTrust` metric calculates the confidence that any given IP address resolution of a domain is correct. The confidence in a resolution is the average similarity between that domain and the other domains which have resolved to that IP. To score whether we believe that `thepiratebay.se` resolves to `10.10.34.36`, we would look at other domains which have resolved to `10.10.34.36` and consider their `DomainSimilarity` with `thepiratebay.se`.

We now discuss cases where a provider allocates non-disjoint but partially overlapping

$domains \leftarrow$ the set of all domains

$ips \leftarrow$ the set of resolved IPs

function EDGE($domain, ip$)

return |ASes where $domain$ resolved to ip|

end function

function DOMAINSIMILARITY(dom_a, dom_b)

▷ $0-1$ value representing confidence that two domains are hosted on the same servers.

return

$$\frac{\sum_{ip \in ips} \text{WEIGHT}(dom_a, ip) * \text{WEIGHT}(dom_b, ip)}{\sqrt{\sum_{ip \in ips} \text{EDGE}(dom_a, ip)^2} * \sqrt{\sum_{ip \in ips} \text{EDGE}(dom_b, ip)^2}}$$

end function

function IPTRUST($domain, ip$)

▷ $0-1$ value representing confidence that an IP is a server for a domain.

return $\frac{\sum_{d \in domains} \text{EDGE}(d,ip) * \text{DOMAINSIMILARITY}(domain,d)}{\sum_{d \in domains} \text{EDGE}(d,ip)}$

end function

function WEIGHT($domain, ip$)

▷ EDGE weighted by IPTRUST.

return EDGE($domain, ip$) $*$ IPTRUST($domain, ip$)

end function

Figure 3.3: Pseudocode of CDN and interference detection joint analysis algorithm. The two functions `DomainSimilarity` and `IPTrust` are iteratively computed to a fixed point. The result allows direct determination of both the IP addresses hosting clusters of domains and which resolutions are anomalous.

sets of IPs to different domains. For example, if a domain `a.com` resolves to IPs A,B, and C, while `b.com` resolves to C, D, and E. If the different IPs are in the same Class-C network, then our analysis will see both `a.com` and `b.com` as resolving to the Class-C network that corresponds to A, B, C, D, and E. This attributes a high `IPTrust` value to

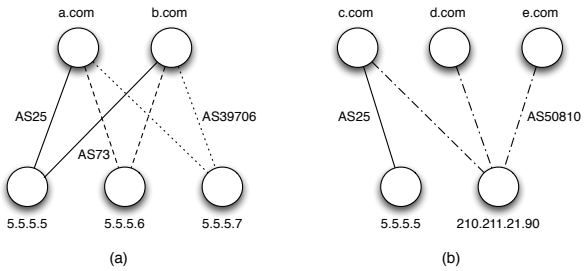


Figure 3.4: An illustration of the relationship between domains and IP addresses. Each edge corresponds to a resolution of a specific domain by a specific DNS resolver, labeled by the AS of the resolver. In this example, we see *a.com* resolves to `5.5.5.5` in UC Berkeley, AS 25. In (a) we see a clique of domains supported by the same infrastructure, while (b) shows otherwise unrelated domains resolving to the same IP within AS 50810.

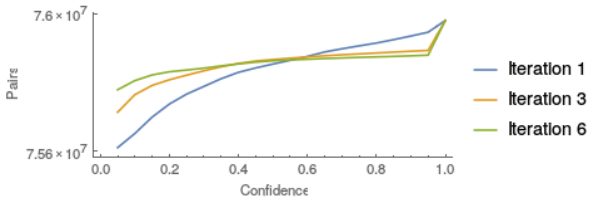


Figure 3.5: `DomainSimilarity` distribution after iterative calculation. After the first iteration, 25,000 edges with similarity above 95% are found. After five iterations 75,000 strong similarities are found.

the class-c network for the two domains. Class-C is chosen as the most specific public announcement of IP ownership, limiting accidental grouping of different providers. If the IPs are in different Class-C networks, the `IPTrust` can still be high if the `DomainSimilarity` is high. In cases where there is only a small fraction of IP space overlap, metadata is not present, and `DomainSimilarity` is low, Satellite will consider the two domains to be in separate clusters. This will attribute a low `IPTrust` to C.

For intuition behind this calculation, consider the representative case of the Fastly CDN. Taking one IP range, `23.235.47.0/24`, we find that Satellite clusters 72 domains as Fastly.

For these, the `IPTrust` metric ranges between 0.75 and 0.98. This IP range was also the resolution for 22 other domains, across which its average `IPTrust` was 0.20 and maximum was 0.30.

To derive an initial estimate of `DomainSimilarity`, we set `IPTrust` to 1.0. We then iteratively calculate these two quantities until a fixed point is approximated, generally in 5-6 iterations. Figure 3.5 shows the effect of iteration on the distribution of domain similarities. Without iterating to the fixed point, many domain pairs have a similarity coefficient close to 0.5. Subsequent iterations concentrate the emergent clusters to more clearly define shared infrastructure (close to 1.0).

Cliques and Deviations `DomainSimilarity` and `IPTrust` form the core metrics we need to determine both CDN footprints (the cliques of similar domains and associated set of IP addresses they are served from), and network anomalies (sets of domains sent to IPs with low trust in isolated ASes).

CDN cliques: To find clusters of domains with similar resolutions in the matrix of calculated `DomainSimilarity` values, we use a greedy algorithm of first making arbitrary clusters, and then finding the best 'swaps' possible until a local maxima is found [63]. This clustering technique has been found to perform close to human labeling.

Table 3.1 shows an example of the highest popularity sites that were clustered into the clique representing the Akamai infrastructure. The largest clusters are shown in Table 3.2. We count the 10 largest shared hosting platforms hosting 1967 domains, making up almost 20% of those measured.

At a global level, strongly connected components represent domains hosted by the same servers. This may be domains resolving to one IP everywhere, or domains with the same CDN configuration which consistently resolve to the same IPs from different vantage points. If we narrow our consideration to the ASes based in a single country, blocking can also appear as a cluster with the block page IP clustered with all of the blocked domains. These clusters are only found in the ASes of individual countries, and the difference between detected clusters

Domain	Alexa Rank
www.ebay.com	18
cntv.cn	79
indiatimes.com	110
dailymail.co.uk	114
etsy.com	149
cnet.com	151
deviantart.com	168
forbes.com	175

Table 3.1: The highest ranked domains identified in the 'Akamai' cluster.

CDN	Size	Representative Domain
CloudFlare	726	reddit.com
Amazon AWS	647	amazon.com
Akamai	410	ebay.com
Google	141	google.com
Dyn	112	webmd.com
Rackspace	77	wikihow.com
Fastly	72	imgur.com
Edgecast	68	soundcloud.com
Incapsula	55	wix.com
AliCloud	54	163.com

Table 3.2: Largest CDN clusters. The top 10 CDNs account for 20% of monitored domains.

globally and nationally is a strong signal for this behavior. On the other hand, this is only one of many ways to interfere with DNS. Some forms, like the response of random IPs used by some Chinese ISPs [15], will reduce `IPTrust` without creating these obvious clusters.

It should be at first surprising that Akamai, one of the largest CDN providers, is represented by a low number of domains. We find that while Akamai transfers a large amount of traffic, we count many of their domains as independent entities for two reasons. First, Akamai often uses relatively small set of dedicated IP addresses to serve the primary domain of specific customers in order to support SSL on some older brwosers. Second, Akamai servers are often IP addresses of the ISP where they are hosted. These appear as if owned by the ISP, rather than Akamai. These two factors cause many Akamai customers to be treated as independent entities by Satellite, and not seen as part of their shared serving infrastructure.

We can compare the relationship Akamai has with customers to that of Cloudflare, which also provides 'white-label' services for large customers to customize their presence through custom DNS name servers and SSL deployed for older clients unable to perform server name identification. Cloudflare partitions its customers across several distinct IP spaces. Some of these IPs have reverse PTR and WHOIS information identifying them as Cloudflare, while others do not. The use of IP addresses within Cloudflare ASes and Cloudflare associated WHOIS information allow Satellite to cluster these services as one entity with more certainty than the less obviously related Akamai customers.

Network interference: The question of "who is blocking what?" can be answered by finding ASes where a majority of resolutions have low `IPTrust` for a given domain. For example, Iran regularly sends `thepiratebay.se` to `10.10.34.36`; we see `IPTrust` of 6.6×10^{-9} for those resolutions, since the IP is also seen for a number of other blocked domains which do not otherwise overlap.

To extract instances of interference that are reflected in the `IPTrust` metric, we look at the distribution of values for resolutions with the same AS. When the average trust for an AS-domain pair is depressed in a statistically significant manner (we currently look for a mean that is four standard deviations below the average for that domain, with variance calculated

over the distribution of all resolutions for the domain) we consider it to be 'suspicious'.

There are several ways in which a domain can have low `IPTrust`, corresponding to different forms of interference. We handle these through a decision tree, which provides a conservative estimate of known forms of interference. Crucially, this approach benefits from the fact that we are able to point to the mechanism which triggers each flagging. The categories we classify as interference are:

1. Too few resolutions or too many unparsable responses are received.

2. A domain which is otherwise 'single-homed' (meaning a single IP address is found regardless of client location) resolves to non-standard locations.

3. A domain with an otherwise 'dominant' AS resolves to many ASes.

4. Resolution deviates from an expected CDN cluster.

Satellite assigns the cause to be the first of these classes which is applicable. Our initial AS-level aggregation allows us to directly find invalid or suppressed resolutions. Figure 3.2 showed that over 60% of domains considered are single-homed, which we use for the third and fourth decisions. Finally, for domains which appear to be hosted on shared infrastructure, we use the `IPTrust` score computed above.

3.3 Evaluation

3.3.1 Address Validation

To validate our ranking and clustering algorithms, and our data collection process more generally, we make web requests to each resolved IP address as a potential location of each sampled domain. More specifically, we connect to each IP which has been seen as a candidate, and request the '/favicon.ico' file, using the domain as the 'Host' header. Slightly under half of the monitored domains have this file and can be validated in this way. We record hashes of all returned content, and compare these hashes against copies of the favicons fetched using

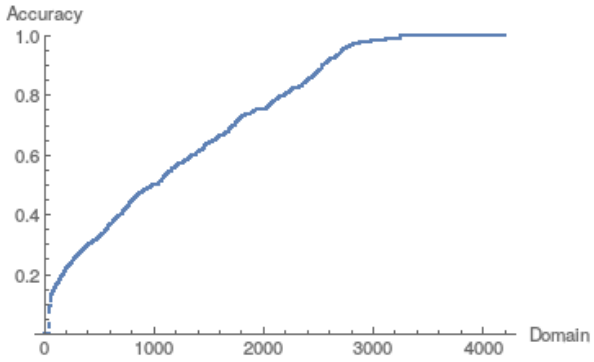


Figure 3.6: For each of the 4,521 domains with favicons, the fraction of distinct IPs resolved with a `IPTrust` score over 0.5. Our automated classification matches favicon presence for over 90% of IP-domain pairs.

local DNS resolution to determine whether an IP is correctly acting as a host for a given site.

Over a total of 965,522 completed resolutions, 82% of resolved IPs are deemed 'correct'. 5,479 domains are skipped in this validation, because no authoritative favicon is present. Validation is performed on the other 4,521. Skipped domains are not used when we evaluate clustering performance.

In Figure 3.6, we show the agreement between this validation process and the confidence scores for IPs used in our clustering algorithm. We treat an `IPTrust` score of 0.5 as trusted, but find similar results for other thresholds. While there is noticeable divergence between the `IPTrust` score and the favicon results, over 95% of those failures are false-negatives (our algorithm was overly conservative in creation of clusters, and gives low scores to IPs the favicon process showed to be correct). The vast majority of these occur in situations where a single partition of IPs is normally resolved for a domain, but other IPs are also able to respond correctly when queried. Both Akamai and CloudFlare exhibit this behavior. Partial aggregation of these clusters has a minor effect on this view, since when domains are fully partitioned onto separate IPs we only consider our trust of those IPs we've actually seen

resolved.

This validation technique is susceptible to manipulation by an adversary which returns the correct favicon image on an otherwise malicious server. We are not aware of any block pages behaving in this way.

In principle, validations like the use of favicons or signals like reverse DNS lookups can also be used in the clustering process to further refine which IPs are believed 'correct' for domains. To us though, this result shows that the DNS resolutions themselves are able to produce largely reliable mappings of CDN IP addresses.

We can also validate our clustering algorithms against the ground-truth of IP prefixes advertised by some CDN providers. For this validation, we consider the Fastly CDN, which uses a compact set of prefixes maintained at `https://api.fastly.com/public-ip-list`. We find that all 12 IP prefixes found by Satellite as the Fastly CDN cluster are included in the officially advertised list. The Satellite cluster contains 72/80 domains found using this ground truth list of IP prefixes. For geolocation, the MaxMind database reports multiple locations, accounting for 5 of the 10 Fastly countries, including the US, Australia, and three of four locations in Europe (mistaking Germany for France). The Australian class-c network prefix is identified as anycasting, which we resolve to 4 of the 5 additional locations – New Zealand, Japan, Hong Kong, and Singapore – agreeing with the results of [40]. These two techniques lead us to correctly find 8 of the 10 locations, missing Brazil and mistaking Germany for France.

3.3.2 Website Points of Presence

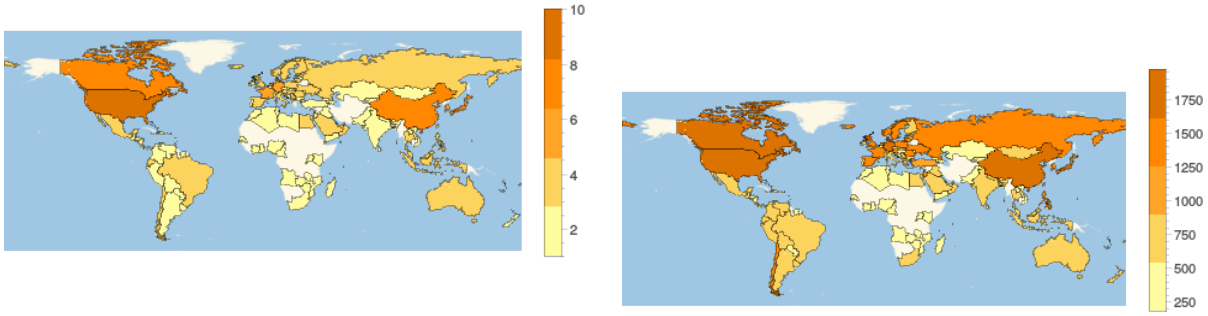
While we have shown in this paper that the Satellite technique is able to accurately map the IPs which are operated by targeted websites, we have not yet shown the implications of that data. Here, we attempt to characterize the dominant content distribution entities in the Internet today, and provide some insight into where they operate and the international nature of the Internet today.

In Table 3.3, we show the IP space we estimate for the largest CDN clusters. These plat-

CDN	**IP Space**	**Clustered ASes**
CloudFlare	107008	75
Akamai	264960	489
Google	476416	1036
Cloudfront	128512	21
Incapsula	12288	17
Fastly	8192	17
Dyn	2304	9
Edgecast	24832	65
Automattic	3584	5
AliCloud	41728	42

Table 3.3: IPs in each of the ten largest shared infrastructure platforms. Variance in size between Dyn, Fastly, Automattic and the others is due to use of Anycast. Some ASes are significantly undercounted by clustering, Akamai has points of presence in over 1,000 ASes.

form each have unique network structures, and use a range of technologies including rotating IPs and anycast, which make it difficult to directly compare scale from these numbers. For instance, most Google IPs resolve to IPs within Google's own AS, while IPs from Akamai are largely resolved to IPs located in the ASes of consumer ISPs.



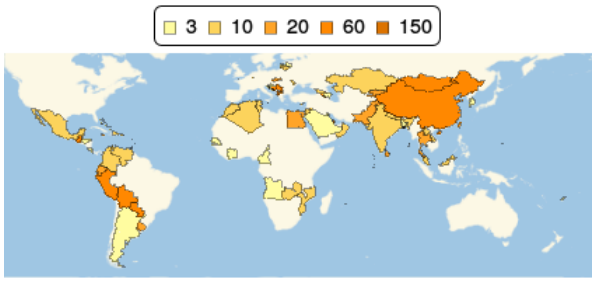
(a) Points of presence of the CDNs from Table 3.2. Anycast is not included, indicating conservative counts.

(b) Number of sites resolved locally in each country.

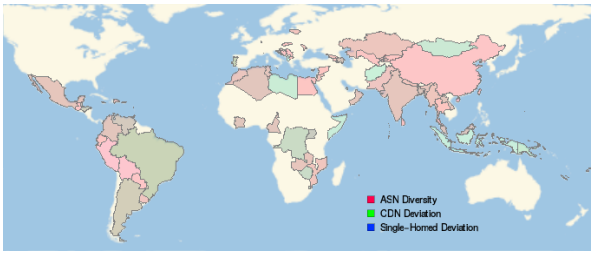
Figure 3.7: CDN characterization in Satellite.

In Figure 3.7a, we use the geolocation of ASes to count which countries these providers are located within. One striking feature of this geolocation exercise is to note that the 10 largest content distribution networks use IP addresses allocated to ASes registered in at least 145 countries. We trust MaxMind for these locations, but attempt to be conservative, including neither anycast resolution nor clustering the true extent of partitioned providers like Akamai. This undercounting is reflected in Table 3.3, which indicates the primary cluster we use for Akamai accounts for under half of the over 1,000 ASes they report [4].

In Figure 3.7b, we plot how many domains are resolved within each country. We see at least 18% of all domains resolving to an in-country IP address for resolvers in China, while other countries like Mexico resolves only 5% of domains locally. This view of domain locality can be used to understand which publishers have complied with local regulations, and to



(a) Number of domains inaccessible in each country.



(b) Types of interference by country.

Figure 3.8: Interference characterization by Satellite. Anomalies are geographic, with some regions like China providing a diversity of false IP addresses, while others like Libya using a single block page. There are no occurrences of only 'CDN Deviation', or 'Single-Homed Deviation' in (b). The relative shades indicate the mixture of the different categories present in each country.

track how much Internet traffic will transit international links.

3.3.3 Interference

Our confidence scoring of how well IPs represent domains helps us address an ongoing pain point in interference measurement: how to know if a returned IP address is 'correct'. The primary issue in this determination traditionally has been whether an IP that is not the same as the canonical resolution is a CDN mirror or an incorrect response. Using CDN footprints along with more simple heuristics for single-homed domains allow us to identify instances of inaccessibility with higher confidence.

We measure interference through positive identification of the four categories in 3.2.6. These categories are conservative, but remain valid for not fully clustered CDNs.

Figure 3.8a shows the number of largely inaccessible domains found in a single snapshot of collected data. We find at least 5 of the monitored domains to be inaccessible in at least

one Autonomous System in over 78 countries.

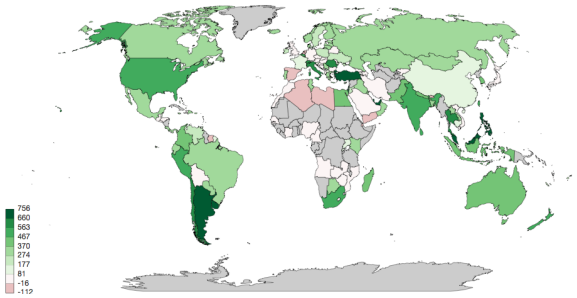
We then divide the instances of observed interference across other factors. Figure 3.8b shows a comparison of interference for sites on CDN infrastructure versus those which are single-homed. While roughly 80% of sites are single homed, we see as much interference is directed at distributed sites, perhaps due to their popularity. This indicates that naive approaches have been missing a significant fraction of total interference instances.

It is possible for a censor to mask their interference from Satellite. Injecting DNS responses using a system of the type known to be in use by China could be targeted to miss an external observer, by only responding to requests originating within the Country or responding correctly to external queries. While much less visible to Satellite, these forms of interference would themselves be visible, and could even be less effective internally. The switch to other techniques like IP or keyword-based blocking would also not be visible in the current DNS data set.

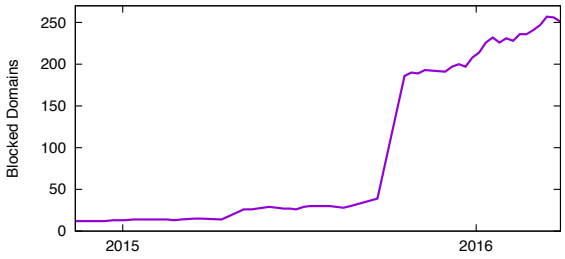
3.3.4 Broader Implications

Our stated purpose in building Satellite and collecting data on the presence and accessibility of popular sites was to allow for new insights into the changing structure of the internet. What are those insights? Many of the implications are inextricably tied to real world events and politics, and reflect on the censorship practices and business environments of nation states. While we aren't comfortable claiming to understand these sociopolitical structures without accompanying real-world evidence, we can show value in the data in light of the larger trends occurring in Internet Governance.

In Figure 3.9a we show the delta of how many more domains are resolved within each country compared to six months prior, based on location of IPs with trust above 0.5 on a per-domain basis. What this shows for each country is how many new domains are now resolved internally where previously they would have been resolved to international servers. This shows the expansion of CDN infrastructure, but also an increasing ability of governments to regulate access within their national territories [51].



(a) How many more sites resolve locally (to IPs within the country) in September 2015 compared to 6 months prior. This figure is based on a dataset of 8,800 domains which remained in the top 10,000 list at both sample points.



(b) Number of domains detected to have anomalous resolutions in Iran since late 2014. An interactive version is at http://satellite.cs.washington.edu/iran/.

Figure 3.9: Longitudinal shifts in Satellite data.

In Figure 3.9b we show the number of domains which are detected to have anomalous resolution across Iranian ISPs. We see a spike in the second half of 2015, which correlates with statements from the authorities there that they were beginning a second phase of filtering. More recently, Satellite has recorded additional inaccessible domains in the lead up to February 2016 elections.

Chapter 4

UPROXY & UNBLOCK: CLIENT ACCESS

When considering the circumvention options outlined in Section 2.4, Social Trust can be seen as a path forward for both increasing IP diversity and collateral damage. Finding lightweight ways for web users to gain paths through their friends is attractive because those links can be semi-private and hence difficult for an adversary to subvert, diverse because there are many more users than there are servers, and disguised to look like video games, p2p, and video chat traffic. To explore the possibilities and limitations association with circumvention systems based on trusted social links we built two systems: Unblock and uProxy. In the rest of this section we will describe the efforts taken to leverage social trust in these usable systems, the surrounding architectural decisions, and evaluate the resulting systems.

4.1 Uproxy

uProxy is a circumvention system based on the core idea of reusing existing social trust as a mechanism for censorship circumvention. uProxy differs from Unblock in focusing on deployment within a web world. One of the major problems for adoption in systems using social trust is bootstrapping a dense social overlay for routing, which we explored in the use of shortcut links in Unblock. uProxy instead operates as a one-hop proxy - your traffic is sent to a single friend of your choosing, and appears to come from that friend.

uProxy as a system further differentiates itself through distribution as a web browser extension, piggy-backing on existing social networks to avoid tool-specific infrastructure, and a focus on localization and international distribution.

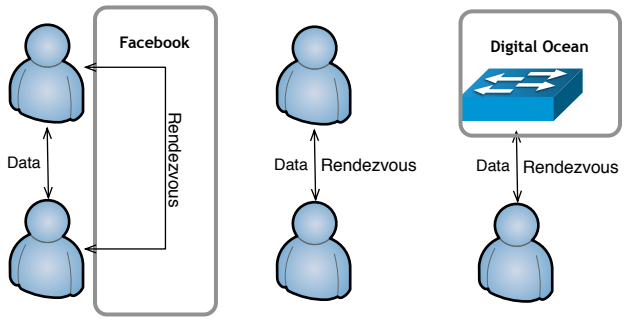


Figure 4.1: Three different interaction models adopted by uProxy. In (a), two users rendezvous through an existing social network, to bootstrap a direct connection. In (b), the use an out-of-band mechanism to rendezvous directly. In (c), a user routes traffic through a cloud machine they have provisioned themselves and which they may share with friends.

4.1.1 Design

uProxy is designed to meet four overarching goals for circumvention:

- **Decentralized**: uProxy should be able to operate without any specific centralized server infrastructure.
- **Usability**: uProxy should be simple enough to be used without instruction by a typical Internet user.
- **Inexpensive**: The application should be open source and either free or cheap to run.
- **Performance**: Internet performance through uProxy should be comparable to other VPN services.

In the rest of this section we discuss the technical features of the system. uProxy consists of 4 high-level components: (a) a portable proxy server and client that is bundled into a browser extension, mobile application, or server daemon, (b) an intuitive user interface for access control and route setup, (c) a plugin framework for rendezvous services that

Figure 4.2: Screenshot of the uProxy user interface. Our goal was to make the workflow as simple as possible, automatically configuring the server as necessary and walking the user through any configuration as necessary.

allows users and devices to discover each other using existing social networks such as Gmail, Facebook, GitHub, and WeChat, and (d) a robust network stack and obfuscation framework that reliably connects 2 hosts.

A Diversity of Proxies

uProxy packages a proxy client and server in every installation. When accessing the Internet through another computer run by a friend, we simply establish a connection to the friend's proxy server instance. In the cloud use case, the same code base runs without a user interface, and the client includes a 3-step (choose host, create account, and choose server location) process to provision and deploy a cloud virtual machine. We also give users the ability to share access to their cloud server with invitation links. We expect the aggregate cost of our system to be cheaper than many commercial VPNs, as all costs involve only raw compute resources, while providing each community of users their own dedicated proxy server at a unique IP address.

User experience

Minimal linear user experience: Users want the easiest route to censored websites. In early versions, we asked users to log in and discover friends on their social network. While a common user experience on the web, this did not match user's expectations. In later versions we oriented the software around linear flows. First, we ask how the user wants to find a relay. Then we either walk the user through inviting a friend to serve as a relay or through setting

up a cloud server.

Packaged interactive documentation: Because any online documentation can be censored, we implemented an interactive walkthrough system. Browser extensions have the benefit of being able to dynamically read and modify the web. We use these APIs to provide real-time information about why actions are necessary (e.g. creating an account with a cloud provider) and to guide the user between steps.

Rendezvous

To avoid manually configuring connections, we use a variety of rendezvous techniques to discover proxies and exchange messages to negotiate a connection. The variety of options creates a usability challenge, since each technique suggests different user experiences.

For example, we support a number of social networks, including Facebook, GitHub, and WeChat. With user consent we discover friends, establish relationships with other clients, and setup P2P connections. In some cases, (like GitHub) we use public APIs, while others (like WeChat) force us to reverse engineer their internal protocols. The architecture for these plugins consists of 4 methods: `login`, `listContacts`, `sendMessage`, and `getMessage`. Other rendezvous mechanisms include cloud databases with external authentication, such as Facebook authentication over Firebase, and our own rendezvous server shielded using domain fronting.

While the social network may be able to tell you are using uProxy it encrypts messages between peers to avoid manipulation. We found that many users had an aversion to 'advertising' their use of a circumvention system (Discussed in 4.1.3). uProxy allows users to send an 'invitation' to existing contacts, and negotiates the system to use for subsequent presence and connection negotiation.

For users with a strong aversion to using any outside services, we support URL-based invitations where a client-generated invitation secret can be manually exchanged, such as over encrypted email.

Network Communication

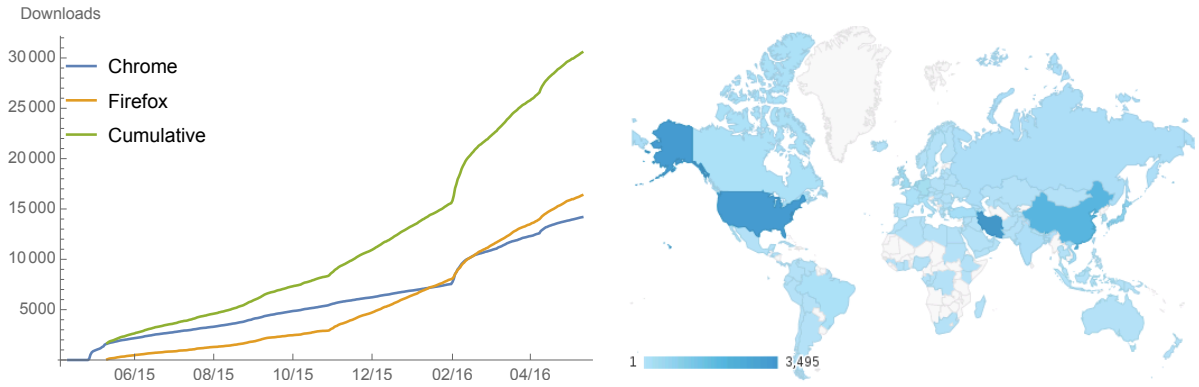
uProxy's network subsystem requires authentication, integrity, confidentiality, packet loss resilience, congestion control, direct connection between peers on typical home networks, enough throughput to stream video, inconspicuous packet types and an implementation that can run entirely in-browser. To this end, we have designed uProxy to both leverage the existing browser-based protocol providing these features, and to extend it for a more flexible protocol that can avoid fingerprint-based censorship.

Browser camouflage uProxy exposes a proxy for the browser which tunnels traffic over WebRTC. WebRTC is a W3C standard for efficient, low-latency communication between web browsers. Originally designed for voice and video, it allows browsers to send and receive arbitrary messages. WebRTC includes a complex connection establishment protocol designed for connecting users behind NAT routers. The connection is UDP-based and encrypted, and session keys are exposed allowing clients to verify the remote peer's identity. Typical consumer systems can achieve a 100Mbps transfer rate using the protocol, before reaching CPU limits. Practically, connections are typically limited by the network connection.

Protocol Shapeshifting While uProxy's traffic cannot be distinguished from other WebRTC traffic, its widespread use could induce an adversary to block all WebRTC traffic. To preempt such a strategy, we include protocol-level obfuscation.

Unlike most obfuscation protocols which operate on a stream of bytes and perform packetization and reconstruction, the DTLS protocol already packetizes and expects loss of data. We perform transformation by intercepting the existing negotiation phase of WebRTC to establish a connection while also routing traffic through a local transformation phase, an approach we call *Holographic ICE*.

The transformation performed by our protocol obfuscator follows the design of FTEProxy, a protocol transformation project that transforms an arbitrary stream of data to match a provided regular-expression [60]. uProxy uses several such patterns, including common



(a) uProxy downloads over the last year. Download counts from Firefox and Chrome.

(b) Locations of uProxy users, as reported by the Chrome web store.

Figure 4.3: uProxy usage and distribution of users over the last year.

expressions used to match HTTP and DNS traffic. On a desktop (2014 3.5 GHz Intel Xeon E5-1650) machine, obfuscation reduces our observed throughput to 11Mbps, which is sufficient for an interactive browsing experience. While not deployed yet, we have plans to further improve obfuscation performance, as well as to byte-pad traffic and introduce timing jitter to deter side-channel attacks.

4.1.2 Deployment

uProxy is implemented in 59,500 lines of TypeScript and JavaScript, with packaging to run as a Chrome packaged app, Firefox add-on, and Android app. The source includes a custom SOCKS5 proxy server and client that connects through WebRTC.

We have distributed uProxy through the Chrome Web Store, Firefox Marketplace, Docker Hub, and GitHub, with over 30,000 downloads since the first public version was published one year ago (Figure 4.3a).

Figure 4.3b provides our best estimate of the geographical distribution of uProxy users. These statistics are provided by the Chrome Web Store, which counts the location at install

time. Statistics for China are notable because the Chrome web-store is blocked, indicating downloads through other VPNs and proxies.

Between April 2016 and May 2016 when we first launched uProxy cloud, 187 users created new Digital Ocean accounts through our in-app registration process.

4.1.3 Experiences

Privacy and Metrics

A common topic of discussion during uProxy development was how to protect the privacy of users, and how to build uProxy without the need for users to blindly trust us. In particular, we recognized the importance of being transparent with our code and practices. We also took extra steps to make sure we were never in possession of raw data reflecting user's browsing behavior, to proactively reduce the risks and consequences if we were hacked or if data was requested under the US 3rd-party doctrine.

Deciding to limit statistics gathering was a hard choice. The team made a conscious decision for uProxy to not send back any individually identifiable information, and to not collect any metrics without explicit and neutrally worded consent from users. We felt that from an ethical standpoint, this was an area where we couldn't compromise, although it made several competing interests much more difficult to satisfy. Not only does it limit the depth of usage that papers like this one can present about uProxy, but it also limits the ability of the team to measure uProxy's popularity or understand areas of user frustration with the tool.

Despite our self-imposed limitations on collection, we were able to develop two strategies for understanding uProxy deployment. The first was to track downloads of the uProxy client from both the Chrome web store and the Firefox add-on market. Both of these installation channels already report download counts to developers, and it is both difficult and less secure to provide a non-standard installation process. The second strategy for metric collection was to include a metric tracking module in uProxy that was (a) explicitly enabled, and (b)

anonymized using the RAPPOR algorithm so that reported metrics were not individually attributable.

Threat Analysis

uProxy is designed to function despite the efforts of an active ISP or nation-state adversary. In this threat model, the adversary has the capability to record and manipulate any traffic from the device running uProxy, although it cannot break encryption. The adversary can pressure corporate entities to block traffic identifiable as uProxy, but cannot prevent all user-user messaging. The adversary can also pose as remote users and target high value users with spearfishing attacks, though we do not consider an adversary with sufficient resources to proactively target all or a significant fraction of the potential user base.

uProxy is resilient to a variety of active adversarial attacks. The system is particularly resistant to active probing attacks since the participants are largely behind NAT devices.

We expect that the development team or our servers could be compromised by an adversary - either through a court action or technical compromise - and that any data in possession of the team could be used against users. We mitigate this by not keeping any personal data about users, and especially attempting to make sure that the team cannot produce any documentation of either the times or IP addresses at which the system was used. We do this by (1) using cloud-fronting so that user IP addresses are only seen by a 3rd party CDN, (2) using anonymized metrics so that the reports cannot be tied to specific users, and (3) creating modes of proxying where there is no need for the client to contact centralized infrastructure.

We do not fully handle the threat of malicious peers who monitor or manipulate user traffic. We partially mitigate this threat by encouraging end-to-end SSL encryption and setting appropriate user expectations.

There are also a set of threats that are not addressed by uProxy that we consider either orthogonal to our design or leave out of scope. We assume that the user has obtained a 'correct' version of the software, that has not been tampered with by the adversary. Our

distribution model is discussed in 4.1.2, but we do not characterize the issues of initial installation as our approach continues to evolve. We similarly assume the user's computer is not compromised with other software run by the adversary.

User experiences

In this section, we discuss lessons learned across our development and deployment since our public launch in 2015. We have conducted structured interviews and user experience studies with groups from the United States (US), Iran (IR), Turkey (TR), and China (CN). Participants consisted mostly of students and members of the Internet freedom community. Most were technically literate, but did not come from a computer science background.

While it is clear that censorship shapes what websites are accessible in each country, cultural differences led to a spectrum of technical expertise, familiarity with VPNs, trust in software, awareness of surveillance, and fear of persecution.

Internet as a space of applications: An early theme across all interviews was how typical users thought of the Internet. Internet access was about the set applications they could use, whether it was through the browser or through native applications. Users rarely had a mental model of the networking involved, or how data is routed or blocked. As systems designers, it was important we map the network challenges to users' desires: "How do I access the applications that I want?"

Application locality: Users in different countries want access to different applications. For Turkish users where Twitter is periodically blocked, or expats in China who are used to applications in their home country, circumvention is a tax. Some users and businesses pay for reliable access from a VPN. Many look for simple and free solutions, and are willing to try different systems until one works [147]. Circumvention can also be passport to other countries, especially for people without exposure to Western applications. For broader impact, it is important for future circumvention systems to partner with content producers to

offer valuable and localized experiences for visitors.

Market considerations: Centralized solutions can provide a simpler user experience, and many offer a free trial service. Do-it-yourself solutions like uProxy may have limited impact where existing centralized services aren't blocked.

Leverage existing word affiliations: We found instances where choosing the right terminology in our descriptions could influence user's perceptions of functionality and trust in widely different ways. For a number of Iranian users, "would you like to enable anonymous metrics collection?" was interpreted as being to enable and disable Tor-like anonymity in the system. Some terms are more familiar in some areas than others. For example, we found most users in Iran and Turkey to be familiar with VPNs, unlike their US counterparts. However, other technical terms like 'proxy server' led to confusion. Overly simplified terminology can also be misleading. Originally we used the phrases, 'give' and 'get' to describe the setup of a route, however some users misinterpreted this as providing access to their computer's files. Similarly, the word 'cloud' drew associations with sharing documents and messages.

Trust is complex: Between malware, adware, and bundleware among other side effects, users rightfully have strong apprehensions about using new software. As a Browser extension, in order to proxy browser traffic we must ask users for access to "Read and change all data on the websites you visit", which led IR03 to immediately quit. As part of our interactive documentation, it was important to highlight what the implications of each action is, so that users could develop a mental model for what information is seen by others. Common questions we received were:

- "Can all my friends see I'm using this software when I log in to a social network?"
- "Can the government see that my friend and I are using this software?"
- "When I share . . . can a friend access my files?"

We aim to make trust explicit, but found that often confusing as users quickly assessed whether to trust uProxy.

Privacy and Anonymity: Especially when censorship was not only technical, users were worried about raising flags about their Internet usage. Even the use of a circumvention tool may be sensitive. In these environments, it is even more critical to adhere to strict security and privacy standards, and clearly communicate software limitations. This is one of the driving motivations for our strict metrics collection system. Systems that broadly claim security and privacy can easily hurt their most vulnerable users. Association with companies can draw immediate positive or negative association. Some users trusted the application more when they knew we could use Google servers to find friends while others saw social media logos as an immediate cause for concern:

> "In Iran... you could arrested for something you have on social media. This is the problem I have when you log in with social media."

Unexpected use cases: A common lesson in building software, including uProxy, is that users will appropriate the technology to fit their needs in unexpected ways. uProxy offers the ability to invite friends through a shared URL. In response, we have found a number of social media communities emerging for the sole purpose of matchmaking peers through the use of these invitations. While we expected uProxy to be used for routing around region-based content restrictions for video streaming, one participant also described using uProxy to share access to scientific articles at their university.

4.2 Unblock

Unblock extends the network model of uProxy to provide a circumvention system designed for web browsing over a multi-hop social-trust based overlay.

Previous research has considered many designs for multi-hop overlay networks[124, 107, 55, 198, 41], but none combine social trust as a security mechanism with a low-latency

circumvention system. Instead, most existing censorship-resistant overlays use relays that are easy to identify and block. This forces users to constantly add and configure new relays, an effort that is financially and logistically exhausting [23]. As we showed in 2.4.3, attempts to hide public relay locations are largely ineffective.

Unblock extends the uProxy notion of trust to a multi-hop overlay network. By asking users to explicitly connect with friends who they trust to conceal their identity, Unblock forms a global social network. Traffic is routed over these links to participants willing to relay traffic out of the overlay (which we call "exit nodes") in a region where the content is not censored. Multi-hop routing, coupled with mechanisms to prevent overlay disruption, hide participants.

The multi-hop design also provides the flexibility for unblock to address one of the major issues in social-trust based networks. Social networks often exhibit a power law distribution for how many connections each user has, where many users only have a small number of friends. Unblock improves performance and availability by introducing randomized shortcut links, untrusted connections that risk exposing a small set of users to an adversary in order to dramatically increase availability. The system also employs a custom set of transport mechanisms optimized for such a multi-hop network.

4.2.1 System Design

Unblock is built on top of an existing social-network based overlay aimed at peer-to-peer file sharing, allowing experimentation on an existing deployment, and the ability to hide traffic within an existing protocol. There are three key features in the Unblock Design:

Social-network based overlay Users in Unblock have real-world trust relationships. They establish a communication link between their corresponding nodes and use it to convey overlay traffic. We use a social overlay because it is easier to keep participation largely secret – individual members might be compromised by social engineering attacks, but it is harder to systematically expose and block a significant fraction of overlay communications. Tech-

nical mechanisms are needed for rendezvous and routing – that is, how to discover the IP addresses of friends, and the paths to exit nodes. A key challenge is that these mechanisms need to be resistant to blocking.

Overlay augmentation To improve availability and performance of multi-hop communication, Unblock augments the social overlay with additional random links that provide shortcuts and a greater diversity of paths. Crucially, this mechanism reveals only a bounded amount of membership information to an attacker.

Optimized transport The augmented overlay path is subject to transport inefficiencies that afflict overlay mix networks. To mitigate the performance impact, Unblock specifies transport layer mechanisms for achieving reasonable latency and throughput.

Social network overlay

There are several decisions that must be made in constructing an overlay. How does traffic flow through the network? Where does it leave the network? How do users find their connections as they leave and re-connect?

Unblock uses exit nodes to create a bridge between the overlay network and the public Internet. These "exit nodes" are self-selecting participants who are willing to offer provide the final hop for anonymous traffic, serving the same function as in the Tor network.

In order to contact an exit node, users must know of its existence. Exit nodes announce their presence periodically through announcement messages over the overlay. When nodes receive an announcement, they *immediately* forward the announcement to their neighbors. The return paths of these announcements create a minimum latency routing tree that is used when communicating to the exit node. Announcements contain a timestamp, nonce, the hash of the public key of the exit node, and an optional set of exit node properties (such as the region where the node is located and domains reachable through the node).

Unblock also includes a Distributed Hash Table (DHT) as a rendezvous service for locat-

ing the current IP address of peers when a node rejoins the overlay[99]. Rather than use an external DHT which can be blocked, Unblock re-uses existing overlay nodes. We want the DHT not to expose the identities of participating nodes; that is, DHT operations should be performed using just local information already known to participants. Other security-focused DHT designs, such as Whanau [123] and membership-concealing overlay networks [192], address these requirements using a Byzantine-resistant algorithm across all members of a social network. We use a much simpler design, spreading the DHT across exit nodes which are already routable.

Both objects and exit node identifiers are hashed onto a circular key space, and objects are assigned to the exit node that is closest to it in the key space. To perform lookups and updates, we use the fact that exit node announcements create a routing table through the system. This table contains the next hop to route to each exit node. When a node wishes to query the DHT it can first look at its local routing table for the exit node closest to the desired key. The node then routes the query to the exit node through the appropriate neighbor, and nodes along the way maintain state in order to route the reply.

This scheme is a variant of Virtual Ring Routing [34], where nodes are able to provide a DHT-like abstraction by routing messages through their neighbors in a physical network. Our approach uses the subset of exit nodes (as opposed to all overlay nodes) in order to improve both security and performance. If all nodes are allowed to serve as DHT storage nodes, then an adversary can mount Sybil attacks and lower DHT availability [189]. While our design incurs higher overhead and a larger routing tables than membership-concealing overlays [192], this overhead was already necessary for relaying user traffic to exit nodes.

Overlay augmentation

To supplement connectivity, we use a hybrid overlay: we add links to approximate a *random overlay network*. The augmented network provides users with additional peers that are likely located at random points in the social network. In addition to providing users with redundant connectivity, these additional *untrusted links* counteract the stringiness of the social network,

greatly reducing the graph diameter.

Our approach is to provide each overlay node with a set of untrusted links based on its position in the social overlay. We view a node's social network connectivity as a unique *capability* and develop a distributed mechanism for providing each node with additional links based on its location. We consider the Sybil attack model introduced by systems such as SybilGuard[215]. When adversaries infiltrate the social overlay, we bound the number of nodes exposed to them to be proportional to the number of *attack edges* they control, where an attack edge is defined as a social link between an adversary controlled machine and a legitimate user. Importantly, our mechanism ensures that adversaries are not able to reveal an arbitrary number of nodes through a Sybil attack wherein they assume multiple identities behind a single attack edge.

We form untrusted links by circulating collections of randomly sampled overlay nodes, referred to as *random node lists* (or RNLs). Each overlay node is identified by its public key and the IP address and port at which it can be contacted. The *RNL* is an ordered list of these identifiers, with the last element being the node that has been most recently added to the list. *RNLs* are propagated through the edges of the social overlay (also referred to as *trusted links*). Nodes probabilistically add themselves to *RNLs* before propagating them further. A node receiving an *RNL* can then establish *untrusted links* to nodes identified by the *RNL*. New shortcut connections to these nodes are labeled as untrusted and are not used for propagating *RNLs*; these shortcuts are used exclusively for routing overlay traffic.

RNLs are propagated through the trusted social network. These paths are recomputed at each *epoch*, defined to be "a long time" – a period of time where the majority of users in the system have changed their IP addresses and therefore the identity of nodes discovered in previous epochs is of little value to an adversary. This period can be on the order of a few days to weeks, depending on the underlying network [213]. At the start of the epoch, each node will take a snapshot of its current trusted links, hash the identity of each neighbor with a local secret, and use these as the set of IDs in a consistent hashing keyspace. The resulting keyspace is used to determine where to forward incoming *RNLs*. The outgoing link is chosen

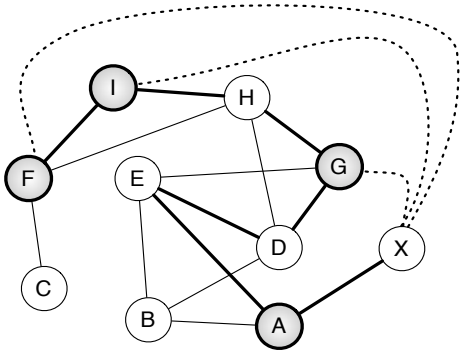


Figure 4.4: Example of the addition of untrusted links. In this example, an *RNL* is propagated through the path *F-I-H-G-D-E-A-X*. Nodes *F*, *I*, *G*, and *A* add themselves to the propagated *RNL*. Node *X* can then establish direct untrusted links with nodes *F*, *I*, *G*, and *A* when it receives the *RNL*. In Unblock, both trusted and untrusted links are used for data transfer.

as the link preceding the incoming link in the keyspace. The keyspace is fixed for the duration of an epoch. The use of consistent hashing implies that *RNLs* are propagated through the same deterministic set of trusted links during an epoch. Further, it also minimizes changes between epochs when new trusted links are added to the social overlay.

Figure 4.4 provides an example of how *RNLs* are constructed. The use of deterministic random walks to propagate a set of identities through the network has several nice theoretical properties we leverage in Unblock.

Enhanced availability Each node will receive *RNLs* proportional to its degree. Nodes will add themselves to *RNLs* based on their own estimate of network size and churn. The mechanism will provide each node on average with a parameterized constant number of other nodes which are an average of $log(n)$ hops away, where n is the estimated size of the network

Reduced path lengths The scheme outlined above also reduces path lengths. Many social networks have the small-world topology property in that a sequence of $log(n)$ random hops through the network often leads to a random node within the network [28, 74, 215]. This fast mixing property probably doesn't hold for all nodes in social networks [141]. Those nodes will not see as significant reduction in path length, but will still benefit from availability improvements.

Mitigating bottlenecks *RNLs* also improve the number of links that cross any cut of the network graph. *RNL* messages propagate at least $log(n)$ hops across each trusted link that separates a censored domain from other uncensored domains. Thus, the augmentation mechanism will increase the number of overlay connections across ISP or state boundaries by a factor of $log(n)$.

Balanced load When users have a large number of existing friends or discovered untrusted links, they will switch to a policy of forwarding *RNLs* but never adding themselves. This avoids hotspots, prevents discovery and blocking of high-degree nodes, and improves the discovery of less-connected users.

Transport considerations

A usable system needs to provide an acceptable level of performance for typical interactive browsing. We believe the choice of protocol mechanism dramatically influence the viability of overlay transport. We use UDP datagrams with custom flow control, the ability to take advantages of multiple paths through the overlay, and a custom application-level protocol for web requests to make the Unblock protocol efficient for web browsing.

Datagram Flow Control The most immediate issue in a multi-hop overlay is that small, latency sensitive flows can get "stuck" behind larger bulk data transfers. To address this issue we use a datagram based transport at each overlay hop and end-to-end congestion

control across the entire overlay path. This minimizes interference between flows that share the same overlay hop.

Nodes in the system can also have very different upload capabilities, which will result in queuing. Flows originating at a high bandwidth node will quickly fill the buffers of subsequent low bandwidth relays. Aggravating this issue, overlay paths span multiple hops, often spanning several continents. End-to-end congestion control responds to congestion over timescales of RTT, leading to slow ramp up and slow recovery from loss. We address these issues by adding explicit per-hop flow control, where nodes communicate how much they are willing to buffer for each active connection.

This mechanism minimizes queueing and eliminates packet loss on overlay nodes by regulating the flow of data from upstream nodes using credits. Credit to send data to a downstream node is replenished through control messages. When a node detects that a queue is building up, it stops issuing credits to upstream nodes, temporarily slowing or stopping incoming flow. This design is similar to mechanisms used in ATM networks [119], which suggest that some queue must be allowed to form to fully utilize the bottleneck node [173].

Nodes in Unblock therefore detect if they are a bottleneck, and manage their credits accordingly. Nodes can detect that they are non-bottleneck nodes when they are limited by credits rather than their own bandwidth. This allows us to fully use available throughput while minimizing latency at intermediate hops.

End-to-end Congestion Control over Multiple Paths The routing algorithm ideally yields multiple paths to a specific exit node. Data from the incoming stream is split into chunks, which are then transmitted across all available paths using UDP datagrams. The receiving endpoint assembles the packets and delivers it to the application in the correct order. Unblock handles congestion over end-to-end paths using a TCP style transfer window for each overlay path that is updated using the traditional additive increase multiplicative decrease mechanism upon packet losses over that path (as in MPTCP [208]).

Application Level Optimization While the transport layer supports tunneling of arbitrary TCP connections, Unblock uses a compressed handshake to reduce the startup latency of the SOCKS protocol. This addition is preferable to running a SOCKS proxy on the exit node directly, because much of the negotiation in SOCKS - authentication and type of connection requested - are determined out of band. This follows a design point found in many circumvention systems, which focus on protocol improvements so that a request is made from the initial message sent to the exit node.

4.2.2 Evaluation

Network Augmentation

Using simulations built on the AKKA framework [6], we found that the shortcut discovery protocol effectively improved the connectivity to any particular exit node in the face of churn, while restricting the number of honest users that are exposed to an adversary. We ran these measurements across data sets representing connectivity of a variety of social networks using connectivity distributions drawn from scraped public connections of foursquare and YouTube. Even with a strong model of an adversary that can block all edges of exposed nodes in the network, shortcuts effectively improve connectivity.

We perform these measurements using simulated networks based on the datasets collected by [139, 216]. For some of these datasets, as in the YouTube social network, we were able to obtain the geographical location of the user. In such cases, we attribute a latency between users using predictions from *iPlane* [130]. Exit nodes and adversaries are chosen at random from these networks. We perform our evaluations for different levels of churn, where uptimes are modeled using Poisson distributions. Lastly, shortcuts are only created between nodes that have degree less than the desired threshold of active connections. This restriction protects high-degree nodes from being overloaded and restricts disclosure of high-value nodes.

Figure 4.5(a) shows the improvement in the availability of paths to exit nodes as we augment the underlying social network for the YouTube dataset with additional untrusted

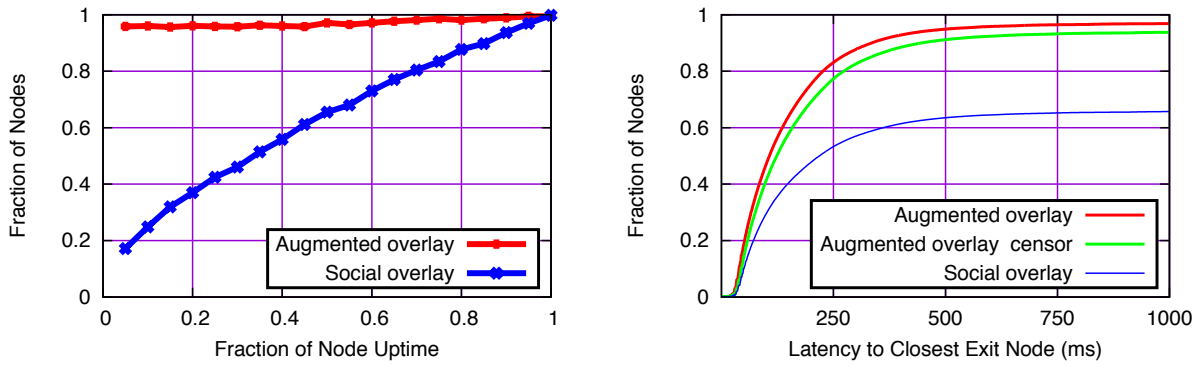


Figure 4.5: (a) Fraction of nodes with paths to exit nodes on the YouTube social network dataset for varying node uptimes and with 10% of the nodes being exit nodes. (b) Impact of untrusted links on latency to exit nodes when 50% of users are online.

links. In this experiment, we set 10% of the nodes to be exit nodes. We perform our experiment for a range of node uptime values. We choose *RNL* parameters to provide 3 online connections. The results show that the augmented social overlay provides dramatically higher availability of paths to exit nodes, especially when the node uptime fraction is low (as is the case with most peer-to-peer systems [182, 86]).

We also examined the improvement in latency of the path to an exit node using the YouTube dataset. Figure 4.5(b) shows the distribution of latencies when nodes are online for 50% of the time. We examine this with and without untrusted links, and observe that the use of untrusted links significantly lowers latency.

Finally, we examined the impact of various types of disruption attacks. We modeled an adversary who had compromised a fraction of the nodes in the social overlay and has the ability to drop protocol messages and disrupt transport channels by dropping packets. In particular, we considered an adversary who dropped RNL messages, forwarded exit node announcements, and then dropped the data packets of an overlay flow. Note that it is more effective for the adversary to forward exit node announcements so as to position itself on more overlay transport paths. Figure 4.6 shows the fraction of nodes with working paths to

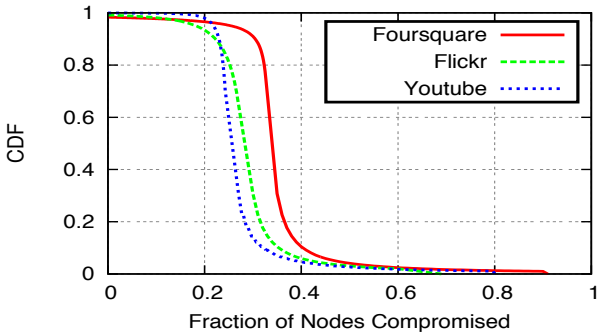
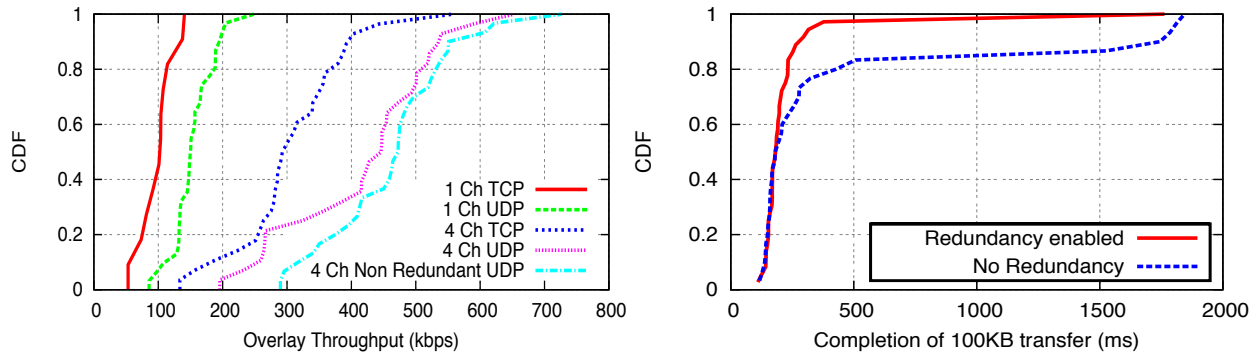


Figure 4.6: Fraction of nodes with paths to exit nodes under adversarial attacks on availability.

exit nodes as we vary the extent of compromise. We find that connectivity in the augmented overlay is adversely impacted only against a determined adversary who has compromised more than 20% of participants.

Transport Optimizations

Using constructed network topologies, we can evaluate the choices we made in designing the Unblock transport layer. We used PlanetLab nodes across the US for these measurements, which have realistic, if perhaps substandard performance and latency characteristics. In all trials, the topology consisted of four disjoint paths from client to server, each with three hops. Nodes were selected randomly from active PlanetLab machines, and reselected each trial. In Figure 4.7a, we compare throughput for the different transport designs we considered: Transferring data using an encrypted UDP transport, transferring data concurrently over multiple paths, and redundant packet transmissions. Using multiple paths with UDP improves throughput linearly until three paths, at which point the bandwidth of either the source or destination node limited further increases. We also examine the throughput of multi-path flows that do not perform any redundant transmissions in order to characterize the capacity lost due to redundancy; this scheme provides only a marginal increase in



(a) Multipath throughput in Unblock.

(b) Latency performance in Unblock.

Figure 4.7: Transport Characteristics of Unblock. (a) shows that UDP performance improves with more paths until the endpoints are bandwidth limited. The non-redundant line represents throughput when packets are only sent once, at the cost of latency (shown in (b)).

throughput indicating that the cost of redundant transmissions is low.

Figure 4.7b considers the latency impact of redundant transmission. We measure the transmission time for a 100 kilobyte flow across the same topology, with and without redundant transmission. While most links in our testbed had robust performance characteristics, when slow or flaky links were encountered, redundant transmissions were able to maintain a low latency connection by mitigating retransmissions and in-order delivery delays.

Chapter 5

ACTIVIST: PLATFORM DEFENSES

As we have already argued, it is critical that publishers improve the resilience of the web platform without user involvement. By focusing on publishers, we can avoid the issues that have plagued user-centric proxies and access tools. We are able to focus on approaches that provide access to all users whereas current techniques reach, optimistically, 60% of Internet users, even in the limited geographic areas where they enter the popular consciousness [217]. We are able to avoid the cat-and-mouse game of access, where users must rediscover tools each time interference mechanisms are updated. A publisher-centric approach also provides for a more direct incentive structure. Publishers are a primary funder of access tools, and that effort can be better spent by improving the technology they use to serve content rather than creating indirect channels for small subsets of users to find.

Mitigating network interference as a publisher does have major hurdles. The majority of effort has been spent educating users because it seems counterintuitive to that an inaccessible publisher could unilaterally make themselves accessible. Publishers are further limited by the expressiveness of the current web platform, which provides limited options for increasing resilience. Browser platforms compete on performance and user perception, and may be concerned about retaliation and market-share impact of features seen as adversarial to government control of information.

Given these challenges, it is still possible to construct a set of steps that can be taken by publishers with varying degrees of buy-in from the web platform. Even without platform support, we prototype a set of mechanisms available today to boost availability in the face of most types of ISP-level network interference. We then propose extensions to the web platform API to remove user-visible changes and increase resilience. By contrast with the In-

ternet "narrow waist" philosophy, our approach is to widen the narrow waist by re-purposing alternate forms of web connectivity to make up for the lack of a direct server connection.

In the rest of this chapter, we first outline the relevant limitations and opportunities for web platform connectivity. We will first outline a few of the central APIs available today, and the security concerns that have shaped them. We then describe a prototype system using only these existing mechanisms in Section 5.2, to show what is possible for publishers today without any platform help. Our prototype combines a novel use of `data` URLs, indirect and CDN-facilitated access, and explicit caching to boost availability. To continue the exploration, we complement this prototype with suggestions for how the web platform itself can evolve to provide our desired functionality automatically. These thoughts are then synthesized with a user overlay and real-time measurement knowledge in the next chapter.

5.1 Existing Defenses

Connectivity: As the web platform has evolved over the last decade, it has morphed from a standard for content delivery to include application logic. This change is reflected in a growing set of APIs for control over network behavior, including Web Sockets, which facilitate interactive communication with a server, Cross Origin Request (CORS) headers [191], which formalize the ability for different web origins to communicate, and Web RTC, a set of APIs for performing browser-to-browser communication. All of these standards stick to the basic notion that web content is allowed to initiate a connection as long as the remote host somehow demonstrates an explicit willingness for such a connection to occur. This standard has emerged from the reality of the web — code running in the browser is not always intentionally run by the user, but may be included by a third party the user is not even aware of. For Web Sockets, a connection establishment handshake is required, where the server must reply to an initial request with a willingness to initiate that type of connection. For other web sites a header, `allow-access-control-origin`, specifies which origins are allowed to make requests. For WebRTC, an intermediated handshake occurs at the beginning of the

connection where each peer sends its transient key and the IP and port it is listening on.

In addition to new connectivity mechanisms, the web platform has also shifted to providing more visibility into the state of connections. Older forms of connectivity such as loading sub-resources, or AJAX requests, only expose a generic error. Newer mechanisms, such as Web Sockets and WebRTC, present an interface much closer to a Unix socket, complete with detailed error messages and control over the conversation contents.

Data URLs: In the drive for improved performance of web protocols, one of the mechanisms that has gained popularity is the ability to 'inline' a sub-resource directly within another. This is commonly used to embed images within CSS stylesheets and HTML pages, in place of a URL the browser must request separately through a new HTTP request. When the images are small icons, this technique can dramatically reduce the number of requests and improve page load time. Data URLs work by using the `data` scheme rather than `http`: `data:text/plain;HelloWorld!` They are structured to include the MIME type, and optionally can base-64 encode the resource to protect binary data from being corrupted in transit.

Service Workers: Web applications can continue to function when offline by registering a 'worker', a contained JavaScript context, which can interpose on requests for server resources and choose how to handle them. This site-defined script has several options for handling requests. It can ask the browser to fetch a request through the standard network stack, using the `fetch` API. It can also service requests using the browser cache or with an explicit buffer of data.

Service workers are designed for offline web applications. The goal is for existing web applications to function reasonably when the visitor is disconnected by making it easy for static requests to be cached or to create an appropriate alternative endpoint for serving data locally. Importantly, this does not change the web application itself. Service workers are interposed so as to see requests, but with no more access to the browser's network behavior

than the application already has.

5.2 Publisher Defenses

In this section, we consider how to use the web mechanisms described in Section 5.1 to improve access for censored users. Our goal is to prototype a system which: (a) works in existing web browsers and minimizes visible changes to users, (b) maintains the existing client-server access model - that is, the confidentiality of user data and verifiability of the server, and (c) works on a broad range of sites without major restructuring of site content. We focus on these goals in order to design a system which is compatible with the web as it exists today. For example, many publishers are wary of systems that remove client metrics information, and a system is much more likely to be adopted if it does not require major structural changes.

Accessing a web site entails a series of technical steps, all of which must work correctly for the user to view content successfully. We divide these technical steps in the following way:

Discovery: How the user finds a link to the publisher's content and follows the URL.

Connection: How the browser establishes a connection to the publisher.

Data Transfer: The process of transferring data and loading the page.

Reconnection: The user's ability to continue accessing content on subsequent sessions.

In the rest of this section we consider these problems one at a time, presenting solutions using available mechanisms. We integrate these mechanisms in our prototype described in Section 5.4 to demonstrate how publishers can directly mitigate interference.

Adversary Model: Our mechanisms are designed to provide resilient access to a publisher despite the efforts of a network adversary who is able to monitor and selectively block or modify traffic. We do not consider an adversary who can break TLS encryption, and we make the assumption that the adversary is unwilling to white-list or generally block external traffic. While there have been instances of nation states enforcing white lists of protocols or domains, these have largely proven unsustainable, and it has been clear to the population what is happening [79]. We also assume that blocking is a 'batch' process — that updates to Internet core routing behavior does not take effect instantaneously. Today, there are instances of ISPs that route 'suspicious' flows through complex analysis pipelines, but even when flows can be classified in real time, it takes on the order of 15 minutes to propagate that knowledge into IP level blocking [207].

Our expectation is that while the adversary may have influence over social network platforms, that level of control does not extend to direct control of URLs that the platform should censor [47]. In particular, we make the assumption that not all social networks can be blocked and that users with an existing relationship are able to directly communicate URLs.

5.2.1 Learning the URL

A DNS-translated URL is typically considered to be the canonical location for a resource. While these references are susceptible to a number of denial of service attacks, they are valuable in their ease of transmission. In the context of getting access to content however, URLs are missing some desirable properties: Which IP addresses of those available in DNS should be chosen? If the connection cannot be established, what should happen next? If the content is fetched over an unauthenticated channel (e.g., HTTP), what checksum should the content have? In practice, these issues mean that while URLs are easy to share, they are often not enough to get access to content.

To address this weakness of URLs, we propose making use of data URLs in a new way — directly entered in the browser address bar by the user as opposed to its normal

use case of transparently embedding small images inside HTML. An example of this type of URL is shown in Figure 5.1, and the construction is described in more detail at our demo website at `https://willscott.github.io/peer-fallback/demo/firstvisit.html`. When loaded into the browser, the URL in Figure 5.1 will cause the browser to render the page it defines, in this case the contents shown in 5.1b. The goal of this page is to balance size (each byte of the page will make the data URL longer) with functionality (what happens when a direct connection fails?). In the prototype we present here, we perform two actions. The first line, a `meta` tag, instructs the browser to begin navigation onwards to the canonical URL. The second line, a `script` tag, tells the browser to begin loading a static script to execute additional code. `fallback.js` is a generic script — it does not need to be specific to a single domain. Further, the request to fetch the fallback script does not need to reveal what domain the user is attempting to load. The script determines the domain from the browser once it is running. We request this script over HTTPS and from a popular CDN so that interference in its loading will cause noticeable collateral damage.

This URL is long at 330 characters, but not significantly longer than those in use by websites which track state in the URL. A standard Amazon product listing browsed to by an anonymous user has a URL that is roughly 160 characters. A search in the ACM digital library has a URL that can reach 200 characters.

In Table 5.1, we characterize how well our proposed `data` URLs fare across different forms of sharing between users. We find that some social networking services link directly to the embedded canonical URL, and ignore the surrounding data URL. We believe it is unlikely that this behavior is a conscious choice, but rather the use of regular expressions not focused on our use case. These platforms are likely to improve their behavior if `data` URLs gain popularity.

It is also worth noting that many links are shared through shortening services, like `bit.ly`, `t.co`, and `fb.me`. These services can offer an identically constructed fallback behavior as a value added service to publishers.

```
data:text/html;c=https://activistjs.com?#74B49E15BA7782878CF12DFA23144
053837D038A;base64,PHNjcmlwdD5kb2N1bWVudC53cml0ZSgnPG1ldGEgaHR0cC1lcXV
pdj1yZWZyZXNoIGNvbnRlbnQ9MTsnK2xvY2F0aW9uLmhyZWYuc3Vic3RyKDE3KSsnPjxzJ
ysnY3JpcHQgc3JjPWh0dHBzOi8vd2lsbHNjb3R0LmdpdGh1Yi5pby9wZWVyLWZhbGxiYWN
rL3BlZXItZmFsbGJhY2suanM+Jyk7PC9zY3JpcHQ+
```

(a) A resilient data URL for https://activistjs.com.

```
<meta http-equiv=refresh content=0;https://activistjs.com>
<script src=//cdn.jsdelivr.net/example/fallback.js>
```

(b) Contents of the encoded URL.

Figure 5.1: An example of a data URL and accompanying encoded page. This form of URL provides the opportunity to execute a script when a server is unavailable. It is constructed to include the original URL for compatibility, as well as a hash of the expected server certificate for validation, in this case `74B49E15BA7782878CF12DFA23144053837D038A`.

Service	Behavior
SMS	Partial
email	Full
Twitter	Partial
Web Link	Full
Facebook	Partial
QR Scanner	Full

Table 5.1: Behavior of our proposed enhanced URLs across a variety of link sharing platforms. Partial behavior indicates that the canonical URL is linked directly, while Full means that the resilient behavior is correctly invoked.

5.2.2 Establishing Connectivity

Here we ask what is possible in an un-privileged browser context when JavaScript code, in this case `fallback.js`, is run with the intention of loading publisher content. In particular, can we widen the interface used for accessing content without introducing vulnerabilities or requiring changes to the web security model? As botnets and other disruption-resilient networks continue to demonstrate, a small amount of semi-dynamic content is enough to bootstrap connectivity.

We focus on two techniques that are actively used in censorship resistant communications and have the potential to work in an untrusted web browser context. The first of these is a strategic use of CDNs to mix our bootstrapping requests with innocuous traffic from the CDN, making it difficult for an adversary to selectively disrupt only objectionable content. The second is development of a peer-to-peer mesh, taking inspiration from systems like BitTorrent, and re-imagined in the browser by projects like PeerCDN [97] and Flash proxy [70]. The high rate of churn of web visitors forces the traffic analysis pipeline to update black lists

in real time, a capability difficult for adversaries to acquire.

Using these techniques, our `fallback.js` prototype first attempts to load hard-coded rendezvous files listing active peers. Several of these files can be maintained by the publisher at known URLs on common domains including `api.github.com`, `dl.dropboxusercontent.com`, and `googleusercontent.com`. Services are chosen where the publisher can claim a URL and keep the contents of that URL updated. This technique for indirection through common domains has gained substantial adoption, although the limitations on access from a normal web page limit which CDNs can be used [71]. Having retrieved a list of active peers, the client then attempts to create connections until it successfully connects with a peer.

These techniques are analogous to those used by botnets to maintain their command-and-control systems, or more historically, the use of radio broadcasts and newspaper articles to communicate small amounts of data to spies. These techniques are hard to block, especially when the intermediary is popular and uses TLS to frustrate selective blocking of specific content.

5.2.3 Data Transfer

Once a connection is established, data must be transferred from the originating server, through cooperating infrastructure, to the blocked client. The major challenge is designing a scheme where the properties we would hope to get from a secure connection, signed and encrypted messages, are preserved.

A major difficulty is that our code runs at the application layer of the network protocol stack[1]. This means that it cannot introspect lower levels, and cannot invoke primitives such as the browser's trust store or networking stack. This is a critical feature of the web security model, but it leaves our code with a dilemma. Either we can rebuild the network stack in JavaScript, or we must choose another protocol for delivering signed and encrypted messages between the client and server, and the publisher must buy-in to this new model

[1]We assume that the client is using a browser with the ServiceWorker API. For older browsers, more restructuring of publisher content may be needed to work with the Application Cache predecessor.

both by supporting the protocol on their server, and designing their site such that it can support this form of indirection.

To address this dilemma, we propose that resilient links include a hash of a PGP-compatible public key, located at a well-known location, like `http://example.com/.well-known/fallback.asc`. This hash appears on line 2 of Figure 5.1a. This provides a root of trust, so that clients can use either the Web Crypto API [177], or a pure JavaScript implementation [117, 81, 152] to generate their own key pair and establish a secure channel. Messages to the server will be sent to a specific handler, which decrypts the message, processes it, and encrypts the response with the key of the sender. Our prototype implements this component as a standalone process that runs on an alternative server port and exists only to proxy encrypted requests of this form. The architectural components of the design are shown in Figure 5.2

5.2.4 Maintaining Connectivity

Having established an indirect connection to a server, or having established a direct connection and hoping to retain it in the future, the final issue is whether we can bolster the ability of a client to retain access on future attempts at connection. This is especially important for cases where content is only unavailable periodically, a form of interference that is seen frequently in practice [48].

At `https://activistjs.com`, we demonstrate the use of this caching, and show that it can be extended even to the older and more ubiquitous Application Cache mechanism. By installing `fallback.js` as a Service Worker or Application Cache for a domain, it will be explicitly cached, and automatically loaded on subsequent connection attempts. By promoting the use of a standard fallback script across multiple domains, it will see increased cache hits and run a very low risk of eviction. Existing clients of a domain using Service Workers will gain the benefits of the resilient links in 5.2.1 even for normal URLs.

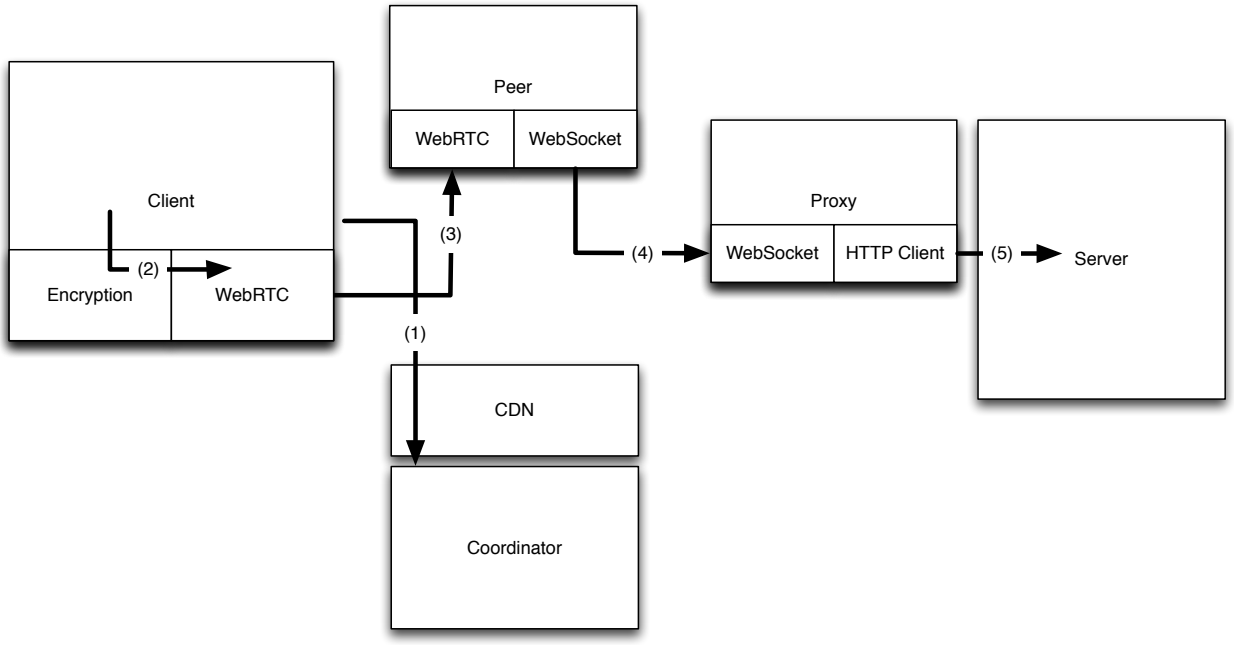


Figure 5.2: Architecture of indirect data transfer implemented by Activist. The client first locates a peer through a CDN-fronted coordination server (1), with whom it establishes a WebRTC connection. Data is encrypted on the client (2), and relayed to the peer (3). The peer forwards contents over a WebSocket connection (4) to a dedicated proxy, removing the need for modification of the server or use of privileged APIs on the peer. The proxy decrypts the request and forwards it to the server as a normal TCP-level request (5).

5.3 Platform Defenses

From the previous exploration of what is available to publishers today, it is clear that there are areas where changes to browser APIs would allow for both more resilience and a better user experience. While it is easy to propose new APIs in the browser that solve one's immediate problem, the ecosystem is extremely complex and mechanisms need to be seen as beneficial to all of the major players to be adopted. While this makes it difficult to speculate on the willingness of players to adopt specific changes, there is some evidence of interest in

making the web more robust to filtering[2].

We can categorize the changes we hope to introduce as solving two broad problems: (a) initiating script execution when loading a URL, even if the authoritative server is inaccessible, and (b) establishing a connection with the characteristics of TLS to a server tunneled through an indirect channel. While there are mechanisms available to publishers to address both of these issues, we argue that those built into the web platform are both more satisfying and more complete.

Adversary Model: Our adversary model when considering extensions to the web platform differs in two important ways from the previous section. First, any mechanism exposed by the web platform reaches many millions of users, so we must consider how that mechanism can be misused by malicious publishers. Second, the platform ecosystem consists largely of companies with global business relationships. The adoption of mechanisms too specific to this application may allow an adversarial state to argue that a vendor is breaking laws, or otherwise threaten consequences to prevent deployment.

5.3.1 Fallback Execution

There remain significant drawbacks to the URL scheme we proposed in 5.2.1. The URL may be confusing to users and does not retain its full benefits in many social media contexts. It seems possible, however, to gain the same benefits those links provide without the need to change existing URLs. Ideally, the browser should take steps to learn whether there is a fallback script pro-actively and transparently.

There are several possible ways that follow existing norms by which a browser could learn of publisher intentions without asking the publisher directly. A list of these preferences could be maintained by the browser vendor, in the same way they currently maintain lists of malware. The list could be distributed with the browser, following a similar model as pinned TLS certificates. Or the preference could be learned during the DNS lookup, following

[2] Particularly in advocacy for HTTPS [202] and against tracking [148].

the precedent set by DANE, which allows DNS to inform a client of the expected TLS certificate [24].

There has also been extensive academic interest in this facet of connectivity. Active Names [190] suggests an enhanced name resolution service that supports computation as part of resolution. Content addressable networks [164] suggest a distributed mechanism by which a client can request static content directly rather than specifying which host it should originate from. The Host Identity Protocol [144] introduces a layer of indirection through a public key which can be used for rendezvous and subsequent dynamic resolution.

DNS records combined with browser pinning provide the most compelling path forward in our view. Chrome experimented with a submission procedure for verification of publisher certificates, but quickly realized it was not a scalable solution [121]. While the use of DNS seems to be a natural location to include this information, it requires caution. Service Workers are considered 'privileged', which means that they can only be installed on an HTTPS website, when the origin server is verified [167]. This means that a corresponding fallback script running with a similar privilege level would also need to be sent with verification of publisher intent. This is not a fundamental problem, and can be accomplished in conjunction with DNSSEC, an existing DNS extension allowing the signing of DNS replies [19]. The primary concern of such a model is that a malicious network can intercept DNS records to strip them of their DNSSEC entries, or perform a denial of service attack on DNS records for undesirable publishers. There are a number of proposals to encrypt or otherwise validate DNS to make such actions harder [150], but all of them have long horizons for adoption, since DNS resolution occurs at a complex boundary between browsers, operating systems, and standards committees, and has proven challenging to update.

Given these constraints, we consider the best available option to be an additional signed DNS entry containing a fallback URL to register as a service worker. This record would be opportunistically returned with normal DNS lookup requests, so that the URL could be loaded if the IP is not available. In addition, an attribute on this record would indicate the publisher's willingness for that information to be pinned by browsers; equivalent to the

HPKP header adopted for pinning of TLS certificates [67].

The ability for browsers to pin this preference, or even bundle associated scripts for content they know to be popular and at risk of interference provides them with the flexibility to create an uninterrupted user experience. Current standardization work in the areas of DNS security, TLS certificate improvements, and the WebAppSec working group indicate that this is in line with the current security model and goals for web platform security [197].

5.3.2 Robust Connectivity

When a domain cannot be contacted, and no fallback record exists, one channel which remains is opportunistic indirection by the browser. Many browser vendors operate services for content indirection already, either as an optimization service for mobile clients (e.g. amazon silk) or for improving access (e.g. Baidu's 浏览器) [22]. There is no common standard for these services, and they have been implemented under an 'all-or-nothing' model. We could instead imagine that these indirection services could be opportunistically contacted when direct connectivity fails.

There are political challenges for vendors to implement such a service, especially since major browsers today are almost uniformly developed by multi-national corporations with business relationships in areas with adversarial networks. In particular, it would be important to separate a standard mechanism for registering an opportunistic fallback service from the operation of the service itself. This division would allow vendors to continue to distance themselves from the service and redirect political pressure, but would introduce a new entity that users would have to explicitly register and form a trust relationship with.

One existing solution used on the web for this form of trusted registration is the ability for sites to act as custom search engines. To participate, a site will include a special HTML tag, which tells the browser how it can be included as part of the browser's search functionality. In response, the browser will show a user interface allowing the user to register the site as a search engine.

In the same vein, we could imagine that services where a user is already active could

include a tag indicating that they would be willing to act as a fallback connectivity service for the user. When the user subsequently encounters network disruption to a different web site, they could be presented with previously seen fallbacks, and could choose to trust one of those services to act as an opportunistic proxy for otherwise unavailable content. This mechanism can be seen as an update to existing proxy configuration mechanisms.

5.3.3 Indirect Connections

A major engineering challenge that we had to address in Section 5.2.3 was creating a satisfying connection to the publisher's server through an indirect CDN or peer. While we prototyped the use of JavaScript cryptography using existing APIs to accomplish this goal, it would be preferable to use the mechanisms for ensuring authenticity already used internally by the browser. Doing so would reduce attack surface, reflect current best practices like certificate revocation, and remove the need for a server-side request proxy. Here, we sketch a possibility for how an indirect SSL connection could be exposed by the web platform.

Existing APIs have been purposely restricted from accessing on-the-wire communications and do not expose an interface appropriate for the back-and-forth handshake that occurs at connection initiation. There are good reasons to structure them this way — the HTTP network protocol includes headers that should not be accessible to the page, like HTTP-only cookies, and ones that should not be controllable by the page, like origin identification headers. In addition, the protocol used for network transfer may be upgraded to HTTPS, SPDY, or another transport.

There is value in extending the browser platform to support validation and re-use of the existing trust store. One way to do this without introducing an entirely new API, would be to extend the fetch API to support an additional request 'mode'. The mode of a request that is fetched is an expression of the security parameters that accompany the request. A mode of 'same-origin' implies that the request should be made without additional network-level validation, but that it can only be made to the same server that the script originated from. The CORS mode indicates that the browser should verify that the remote origin includes a

header allowing the request. We can imagine another mode, perhaps 'locally-terminated', where the underlying HTTPS request is exposed to the client as an RTCDataChannel. The Data Channel is the existing abstraction used by WebRTC to expose a bidirectional stream of data with TCP-like characteristics. This mode would only be supported for HTTPS connections, in order to prevent leaking of browser-internal state. Since the network data is sent back into the browser, there is no worry about a remote server not consenting to the request. Instead, the main concern is leakage of the web site's internal state. However, if the client has the servers TLS credentials needed to decode the private information we can assume the script is running in conjunction with the server, mitigating this concern.

5.3.4 Maintaining Connectivity

Service Workers already provide the capabilities needed for maintaining connectivity on subsequent visits to a publisher. What remains is publisher adoption of existing mechanisms. The focus for browsers will be on scaling this mechanism, and understanding how to keep appropriately cached manifests for many sites, especially on space constrained and mobile devices.

5.4 Evaluation

We have constructed a prototype of the system described in Section 5.2 at `https://activistjs.com` to demonstrate the feasibility of our proposed mechanisms. In this prototype, we make use of a small JavaScript library to rewrite links on pages with their resilient counterparts, and we install a service worker to maintain connectivity when the server is unavailable. Our prototype `fallback.js` script is able to accurately assess network conditions to distinguish interference from instances when the user is disconnected or the server is legitimately off-line.

To test the prototype, we simulate interference through the use of a custom server that can be configured to deny connections from clients simulating several forms of interference. We test the use of bad SSL certificates, host-unreachable ICMP replies and IP-level black-holing (simulating IP interference), non-resolving DNS, and injecting TCP resets to model different

forms of interference. In these simulated interference situations, we have demonstrated the ability to relay content indirectly from the origin server to affected clients. Together, we see the prototype mechanisms as having the potential to provide a noticeably improved experience compared to the level of resilience encoded in the web today.

Table 5.2: The behavior of Activist across a range of browsers and attacks.

Browser	DNS0	DNS1	IP0	IP1	HTTP0	MITM0	MITM1
Chrome	Yes	Yes	Yes	2m Timeout	Yes	Warn	Warn
Safari	Yes	Yes	Yes	3m Timeout	Yes	Yes	Yes
Firefox	Yes	Yes	Yes	5m Timeout	Yes	Yes	Yes
IE	Yes	Yes	Yes	1m Timeout	Yes	Warn	Warn

Attack	Description
DNS0	No UDP response from DNS server.
DNS1	A DNS failure is returned.
IP0	A reset packet is sent in the TCP Handshake.
IP1	No SYN-ACK packet is received to complete the TCP Handshake.
HTTP0	The connection is closed after the client HTTP Request.
MITM0	An Untrusted SSL certificate is provided.
MITM1	An Unpinned SSL certificate is provided.

In Table 5.2 we show that our prototype mitigates a significant number of network interference attacks. Once a script is in place using the Application Cache or Service Worker mechanism, it is able to activate and control how the page is displayed in most interference situations. This analysis can be seen as a worst-case description, since the use of Service Workers can allow the page to display correctly for modern versions of Chrome and Firefox without either a warning in SSL-level attacks, or significant timeout delays from throttling or HTTP-level attacks.

5.5 A Resilient Application

The good news is that the path to resilient web applications is largely an evolution of the ideas already expressed in this text. It is worth explicitly piecing those systems together, and especially considering how the web security model and censors will interact with that integration. We consider these connection points in the form of three questions:

- What can the client and publisher do better if they are informed about the nature of interference expected to occur between them?
- What circumvention mechanisms can be implemented in the web today without imposing unreasonable burden on any of the actors?
- How will the censor react, and can an arms-race be avoided?

Activist is not the only effort underway to build web applications which continue working despite server unavailability. The other notable efforts in this space are focused on a different threat model – not that a malicious network prevents access to the server, but rather that the server had been compromised by a malicious attacker. By keeping state on the client, it can verify subsequent code sent by the server and ensure that only code which has been properly signed by a known developer is loaded. This technique is employed by `cyph.im`, an encrypted messenger web application. It is also the motivation for `hyperboot.org`, a bootloader for offline web applications.

5.5.1 Using Censorship Knowledge

The first question to consider is what more could be done as a client initiates a connection to a website when it is aware of the network interference it can expect to encounter. The specific actions that can be taken will inherently be specific to the type of censorship. For instance, if the client knows that it is in a network that injects TCP RST packets to end connections when undesirable content is noticed, it could have the option of ignoring RST packets it

receives. If the client knows that the domain is blocked, but not the shared provider, it could automatically default to a cloud-fronted connection to avoid the interference.

One of the major outcomes that funders in the space of Internet freedom are pushing for is the ability to directly compare censorship of different circumvention tools. As this thesis fixates on, one of the major issues in censorship today is the amount of effort left to users, and one of those major efforts today is finding working circumvention tools. One of the future visions that has been proposed by circumvention developers [96, 72], is a dashboard showing the status of different circumvention tools around the world. Funders would like to see the tools go one step further, and provide users with appropriate links when their protocol is blocked but other alternatives remain functional.

Complementary to these actions taken by circumvention tools and browsers are the options available to web servers. When a server receives connections from clients in censored areas, it can put additional effort into obfuscating responses so that they won't trigger keyword based censorship. Alternatively, it could implement the activist caching mechanism selectively to reduce any overhead from users who are unlikely to be censored. Finally, web servers have the ability to enforce encryption standards and alternatively issue warnings such that users are motivated to install updates and higher security clients to access content they find desirable.

5.5.2 Mutually Acceptable Circumvention

There are many existing pluggable transports which are already capable of circumventing even the most technically advanced networks. While these protocols exist, the migration and overhead of using them is often considered too great, and they are not used by default even by many circumvention systems. Part of this cost is the performance cost of using the protocols, while another is the integration and support challenges associated with them.

In contrast, we do not see this opt-in approach in web browser features to nearly the same extent. Through the web standardization process, and through the desire to compete on features, new technologies implemented by two of the major browser vendors are quickly

re-implemented by the others.

A major distinguisher between these two models is the incentive model. For circumvention tools, the user has a significant amount of power. If the user wants access to content, the tool feels it is responsible for providing that access. This view is supported by the common financial arrangements.

Web Platform changes have the opportunity to bridge these two arrangements into a more positive experience. The platform does not feel the same responsibility to users, since these programs act on behalf of users, rather than having the ability to essentially pay ransom for user access.

Chapter 6

CONCLUSION

To conclude this dissertation, we contextualize the previous systems in terms of the future adversary they must face, and the future opportunities that arise from their development. The web is not a static environment, and these systems have been built on a foundation that is rapidly evolving. Even in the creation of this dissertation, new privacy threats for browser identification have been found in existing web standards [64], and we can expect many more to follow. In light of this, we consider the different points of opposition where platforms are actively evolving attacks and defenses to find points of stability in clarifying our threat model and what safety means. The development of this technology is also a cause for celebration, since existing weaknesses are not set in stone and can be fixed. We provide summarize a set of problems identified in this thesis as increasingly critical going forward as next steps.

6.1 The Evolving Threat Model

As we move towards a world where collaborative effort between web publishers and platforms brings us to applications that are difficult to disrupt, success is inherently tied to how reliably these applications work in adversarial networks. We can think about the different stages of access temporally: software (in this case, the web browser) is first loaded on the machine, the user later learns of a service they want to visit, and finally they direct their web browser to load that URL.

Compromise of the client, or popularization of pre-compromised clients, is a significant problem. Especially in countries with significant internal economies, such as China, users already regularly use software they associate only with software development efforts within

their own country. This provides a significant opportunity for local norms or local governmental pressure to influence the capabilities of software and produce widespread use of clients which cannot access controversial content. This *Client Problem* extends much further than this work, and we can expect it to be a source of controversy for years to come. At present, the situation is generally positive, since the large effort required to re-implement a web browser is such a high barrier-to-entry that almost all national development efforts have instead chosen to appropriate the open source efforts of the several standardized browsers. This re-use incurs dependence on the original companies to develop standards, and means that the path of least resistance is to follow the development of those browsers rather than facing the work of maintaining patches to disable specific features.

Compromise of URLs has also occurred for both political and commercial reasons. Links are regularly de-listed from search engines, and removed even when sent directly to friends based on opaque platform policies. Some of these removals are motivated by a desire to stop the spread of malicious software or viruses, while others are enforced by a variety of laws. Without natural mechanisms for viral spread of links, the process of discovery becomes difficult. Tor has faced this *Discovery Problem* in its implementation of onion services, which are identified by seemingly random looking host names. Link shorteners and directory services come with a host of issues. Some issues, especially legal liability, come from centralized structures. Others, like the stagnation of names as original claimants become inactive, come from decentralized systems. While the original URL remains crucial to web browsing today, there are significantly more functional discovery systems already facilitating transitions between the Internet and the real world.

This thesis has focused on the third problem, *Network Interference*. However, even this area remains a source of concern for the future. We've already seen calls for countries to completely disconnect their networks from the global Internet, with countries like China and Turkey continuing to take steps in that direction. This fragmentation, along with a host of techniques for information control which are difficult to measure or hold accountable present very real concerns. Even when services protecting free expression are discoverable,

non-neutral costs and manipulation by local networks can have huge impacts on population behavior.

6.1.1 The Client Problem

In a 2014 workshop convened by Google, participants working on measurement and circumvention technologies were asked to predict what network censorship would look like in 10 years. The consensus among the group was that censorship in the core of the network will become unsustainable as encryption gained ubiquity. Instead, the fight for control was predicted to move out to the end hosts, and indeed a growing number of indicators point to this as the next battleground.

The core of the Internet remains and will remain a powerful position for surveillance, but the ability to enforce fine-grained access control is rapidly becoming harder than the effort of widespread software deployment. One harbinger of wider web encryption is letsencrypt [83], a certificate authority providing free and automated certificates to web servers. While it still requires technical competence and a slight performance impact to use SSL, this effort greatly reduces the barrier posed by SSL accreditation, and has issued one million certificates in its first 16 months of existence [1], becoming the 4th largest issuer of certificates. Another signal is the use of end-to-end encryption in messaging applications like WhatsApp and iMessage, and accompanying tension between the parent companies and governments issuing subpoenas for user data.

The battle for control on client devices is not new. In 2008 China commissioned a piece of software named Green Dam Youth Escort, a piece of software aimed at restricting access to online pornography. For a brief period in 2009, the ministry of industry and information technology gave notice that all new computers sold in China would need to be pre-loaded with the software [149]. The move failed to gain traction, in part due to flaws found in the software and resistance from foreign manufacturers. A more recent attempt to gain control of end devices was made by the government of Kazakhstan. In 2015, Kazakhstan legislated that all encrypted traffic should re-encrypted by local ISPs using a governmental certificate that

would be installed on all devices in the country, although again the effort faces substantial resistance [32].

While governments have not had significant success in directly installing software on end user devices, there is instead growing influence and legislation of software vendors. A recent string of analysis performed by the Citizen Lab in Toronto [113, 49, 114] have shown significant amounts of surveillance and capability for remote control in popular Chinese web browsers. This is notable because the regulatory control structure in China enforces the presence of a government position for regulating user speech within any company operating a business of this type [218].

In the US and western Europe, there is a popular misconception that this is a remote problem, and one that we are protected against with strong freedom of expression laws. On the contrary, events like the pressure by the US government for decryption capabilities of Apple iPhones [62] and the demonstrated abilities to intercept communications en masse with the help of phone operators [10] show that this desire for control and access is more universal.

This desire for control of client devices extends beyond governments as well. In particular, control of end-user devices has been a target for criminals and a core part of the underground economy. Malware is a classic example of this secondary market. Access to devices is monetized both for the physical resources (for example in order to send spam), but also for access to user personal information [157]. A recent tactic in this market is to approach developers of web browser extensions and make offers to "purchase" the extension. Once an attacker has authorship of an extension transferred to them, they're able to update it with a malicious payload and gain control of all devices which have the extension installed [8].

To combat the increasing threats aimed at user hardware, platform and hardware makers have responded with increased compartmentalization. Smart-phones have replaced the single file system model of desktops with a much stricter policy where applications are not able by default to access files they didn't create. New versions of Mac OS X have adopted this model, to increase the partitioning between applications and attempt to protect private data from

compromise of other programs on the device. Intel is simultaneously pioneering a parallel form of compartmentalization at a hardware level with its forthcoming SGX technology [25]. SGX provides a mechanism for an application to run "securely" with trust only in the Intel hardware, and not either other applications or the operating system on the device. This approach works by allowing the application to gain an attestation from the hardware that its own memory image is what it expects, and then provide access to an encrypted section of memory only to code executed from that known image.

In this competitive environment with many interests competing for control and influence over client devices, it will be critical to find a balance and compromise which protects the rights of end users. The tactic beginning to emerge from the free software community is the development of open hardware devices like the Novena [73], which are released together with the full specification of the circuitry and processors used in their construction. The benefit from this openness is that it provides an opportunity for the competing interests to flag defects such that nobody has an unfair advantage. While this may seem like a direct reaction only to the hardware control by Intel or similar vendors, open hardware can also be seen as a base needed to advance a neutral system upwards as well. Closed firmware and drivers associated with our hardware devices prevent compatibility and hinder competition by free Linux-derived software systems.

6.1.2 Discovery

In September of 2010, Libya blocked access to the `vb.ly` URL shortener for providing links to pornography which was claimed to violate Libyan law [138]. Shortly after the removal, the registrant wrote:

> [Ben Metcalfe [138]] Our domain 'vb.ly' was deleted by NIC.ly without warning or notice on or around September 23rd 2010. We were subsequently told that our domain has been removed to us being "in clear violation of NIC rules and regulations" relating to "text referring to adult content and offensive imagery from

> [our] main page". ... Again, while we contest that there was NO pornography or adult material on vb.ly, I would suggest that there is a far more concerning issue here if domain registries can decide on the validity of a domain registration based on the content of the website that uses it. I would argue that the two are extricably decoupled and separate entities.

Despite the technical separation which exists between the DNS system, which serves as a tool for discovering IP addresses serving domains and the domains themselves, DNS has proven to be a convenient system for legal action. By 2012, ICE officials in the US had seized 758 domains in the .com and .net registries for violation of copyright [52]. Restrictions on registration qualifications and regulations on domain name ownership have grown increasingly stringent. Many country domain registrars now require that entities registering names must show a physical legal presence in their country to register. Beginning with their 2013 agreement for registration and increasing since, ICANN, the multi-stakeholder organization governing policies for the DNS system as a whole has moved towards requiring valid contact information from more domain owners [95]. These changes point towards an increased use of the naming system as a method for both identifying and punishing those who publish offensive content.

These restrictions limit the *discoverability* of content.

DNS is not the only way discovery is limited. China pioneered Internet censorship not by blocking the exhaustive list of websites serving unwanted content, but by blocking search terms deemed sensitive [220]. That fight continues today, as Europe takes steps to cement "the right to be forgotten," an EU-wide law allowing individuals to limit searchable discovery of negative articles about them. Copyright and intellectual property rights have already been used as controls to limit discoverability of content on search engines and platforms hosting user-generated content. The Lumen Database, operated by the Berkman Center for Internet & Society at Harvard, has amassed millions of content takedown notices served to search engines and platforms to document the 'chilling effects' they leave behind.

Limitations on discovery are also imposed unilaterally by corporate platforms. Facebook will choose not to turn URLs it suspects as being viruses or malicious into links for users to click, and further prevents sending with URLs on a blacklist it maintains [103]. These services use limits to discoverability not just for abusive and malicious content, but also content deemed politically objectionable. Content affiliated with ISIS and other

There have been technical developments reacting to the increasing limitations on discovery. One of these is intertwined in the story of MegaUpload, and its embattled founder Kim DotCom. Kim faces significant charges for copyright infringement due to the discoverability of the service offered by MegaUpload, and has reacted through technical adaptation aimed to provide discoverability without incurring the full risks and liability in hosting a directory of content.

MegaUpload was a prominent file locker run by Internet personality Kim DotCom (born Kim Schmitz). In 2012, charges were filed against Kim from the US for copyright infringement, as users were sharing files without rights and the site was not taking sufficient actions to prevent that abuse. As part of the case, the domain was seized. Critical to the prosecution is the argument that since the data was stored on company servers unencrypted, they knew the material they were hosting and should have taken due-diligence to remove offending content [85].

While the case against Kim continues, he has re-established his service under the brand 'MEGA' which directly reacts to the liabilities he found with current discovery restrictions. Mega is still at heart a file locker, but all content is encrypted such that Mega does not know the contents or file names stored on its servers. URLs for stored files are split, and of the form `https://mega.nz/#!encryptedhashstring`. When visiting the site, the hash, a key used to decrypt the file, is never sent to the server, but rather the encrypted file is sent to the client, who decrypts the content in their browser. These links can be passed between users as easily as normal URLs, while reducing liability for Mega.

The other technical attempt to limit restrictions on discovery has been through alternatives to the domain name system. In 2004, The Tor Project introduced Onion Services,

the ability for a service to be offered through the network without its true location being revealed. Instead of using names for these services, they were instead identified by a hash of the public key of the service provider. From the URL, looking like `facebookcorewwwi.onion`[1], the client is able to ensure that the public key it gets indeed has that hash and is therefore authorized to provide service for that name. While the scheme benefits from the ability to verify authenticity of the service, it is not conducive to memorable names.

In focusing on authenticity of control over discoverability, Tor Onion Services encountered a pitfall associated with the lack of a naming scheme. Forums and other public spaces where links to Onion Services were posted began to feature new links submitted by users, which were indistinguishable from existing services with reputation. The only difference in fact was that at locations were users would pay money, typically through the Bitcoin digital currency, the destination of the money would be different and would instead fund the man in the middle [54]. This attack was enabled by the lack of memorability in Onion Service names, preventing users from easily distinguishing real and impostor domains, even when they had already visited the real service.

Tor is currently in the process of redesigning its Onion Service architecture, and is once again struggling with discoverability. Tor is doubling down on its promise of authenticity of service content by extending Service names to 52 characters, encoding a full elliptic curve public key [135]. This increased length is designed to explicitly prevent discovery 'attacks', where participants in the network could learn the names of the onion services being accessed. There are a host of worries with the new design associated with this choice, not only how to recognize these URLs and make sure a link is 'correct', but also how to even ensure you have transcribed a URL correctly.

Bolstered by Onion Services, a renewed interest in mesh networking and distributed systems, and IPv6, there is renewed interest in decentralized naming directories. Unfortunately, all of these schemes are hindered by substantial design compromises and are accompanied

[1] Facebook spent a considerable amount of CPU effort to generate a key with this hash, identifying one of the limitations of the current Onion Service protocol in the process.

by significant amounts of skepticism.

One of the most recognized systems for a decentralized name directory is Namecoin, a so-called 'alt coin' in the Bitcoin family. Namecoin allows anyone to pay money to claim ownership of a name in a distributed ledger replicated by all participants of the system. If a name is claimed that has already been claimed, participants ignore the subsequent claim, following a first-come first-serve model [115]. One of the exciting developments of Namecoin is that it begins to resolve a long standing set of desires for a distributed naming system [184]. This desire was initially stated as Zooko's triangle: that a naming system could only provide two of three desirable properties: Human-meaningful, Decentralized, and Secure. Namecoin has the capacity to provide all three of these guarantees, and the problem has been accordingly updated to consider the issue of persistence, which remains an open issue [212].

As with the client problem, it may be innovations in the real world that guides a way forward. URLs, while easy to share digitally, have been deemed sufficiently cumbersome to transfer in and out of physical media that other options have been pursued. Among these, QR Codes – 2D bar-codes easily scanned by cellphones – can represent a service without limitations on the length of the URL. While QR Codes do not help with the memorability issues faced by Tor, they can help with convenience. Other advertisements and physical representations of services use names provided by trusted third parties, their identity on Facebook, Google, or Twitter. These mechanisms centralize power and potential for regulation on these identity providers, but reduce our current reliance on a single name registry for all content.

6.1.3 Adversarial Communication

In the fall of 2008 just after the Beijing Olympics I spent a semester studying at Beijing University. The network adversary I experienced was largely unlike what is described in Chapter 2. Most of my communication externally circumvented the deep packet inspection in place through a personal tunnel similar to what is described in Chapter 4, which was

unaffected by censorship. Instead of the governmental adversary I should have seen in retrospect, I actually experienced a much more effective adversary at the college registration desk. Internet service at the university was free, as long as your usage was in-country. In order to enable your account for international traffic, you had to pay a small monthly fee, and more importantly spend the time to successfully navigate a complex bureaucracy.

Most of the Chinese students around me didn't bother, and used the free national Internet.

In 2013-2015, I spent quarters teaching computer science in Pyongyang, and found disturbing parallels in how restrictions on Internet access were implemented. North Korea is known for its national intranet [132], and most students were familiar with that network but not the larger international Internet. The reasons I heard from students mirrored what I'd seen 5 years earlier in China: the local network was much easier to access, was pre-filtered to 'safe' content, and was localized for accessibility.

As we worry about the growing splinternet[2], the reality is that it is already alive and well. What we need to predict to build effective circumvention systems is what forms of communication will persist at high bandwidth and low latency in the future. These are the channels where circumvention can potentially continue to thrive. In this light, the censorship systems we see today are a product of the growing understanding of what those channels are.

There is a second war underway for non-consensual international communications and it is literally overhead. The extreme form of this guerrilla communication is seen both on the illegal satellite dishes seen on rooftops throughout Iran, and in the balloons laden with USB sticks and copies of Wikipedia sent into North Korea by anti-government activists [156, 88]. Both of these cases are driven primarily by the desire of communication between an external diaspora community and individuals within the restrictive country. While the medium – broadcast TV and static databases of knowledge respectively – is indirect, it's seen as a way to stay connected and gain education to stay in sync with remote friends and relatives.

---

[2]The splinternet is a term coined to describe the growing balkanization of the Internet, where communication between countries is rare and highly regulated [118].

In the technology sector, a higher tech form of this idea has been prototyped by several major companies. Google announced its development of Project Loon, a system to deliver Internet access through the use of high altitude weather balloons and dedicated receivers that were able to improve on the latency and power requirements of satellites. Facebook has announced its development of a similar system based on lower-altitude drones, which hover in place as aerial relays. SpaceX has launched a project to provide a satellite-based network for full global Internet coverage.

While significant limitations to access can and will exist through white-lists, there will always be exceptions. We live in a world that is undeniably globalizing and more connected than it has ever been. In 2007, 144,000 Chinese students studied abroad, and by 2012 there were almost 4 million students studying abroad each year globally [180]. These individual relationships held by those with the means to travel internationally will demand the ability to call and chat internationally in a way that is difficult to ignore. In these Peer-2-peer situations, we can expect a continued possibility for network circumvention, even in the harshest situations.

6.2 Next Steps

In light of the evolving landscape of information controls and attacks shaping our access to the Internet, there are several specific directions to focus effort on. One of these is translating the knowledge already present in the technical censorship measurement and circumvention community into broader public awareness. The technical data may show clear examples of regulations over-reaching legislation, but the community with access to that data is not the community that can identify that overreach. New techniques of online information controls continue to outpace our ability to measure them. Throttling of connection speeds and degradation of network neutrality pose a threat that we are unprepared to measure. Censorship by platforms rather than networks, especially in semi-private settings is also very difficult to monitor or understand the extent of, putting huge amounts of trust and power in the hands of a few technology companies. These constitute tangible threats towards

which concrete technical action can be taken, and serve as future directions for the research presented in this thesis.

6.2.1 Advocacy and Impact

Public documentation of online technical censorship exists, and the extent and specificity of this documentation is growing. It is no longer simply a problem of understanding the phenomenon, but also using our existing understanding to respond and present alternative solutions. There are compelling arguments both for and against these methods, but for the most part these are driven by political or expression goals rather than interpreting the data to understand the true effects of current policies.

The goal of the measurement work presented in Chapter 3, along with many of the measurement systems in development elsewhere are to inform these debates. At present, they are sufficient for some forms of advocacy and political understanding of policy - we can learn the opaque policies behind censorship and argue from an empirical perspective that expression is being limited. What these measurements are not yet sufficient for are proactively arguing that policies have over-stepped the laws they are enacted under, or the user experience change caused by these policies.

Some of the remaining work is technical - better visualization and automation of analysis in particular will improve the accessibility and impact of existing measurement work. For users, the dance of finding working circumvention strategies can be improved by a centralized real-time understanding of which circumvention methods are working and which are not. Such an effort requires buy-in from tool developers, including those who hide the tactics they use for connectivity, to be effective. Automatic detection of changes in policy and behavior are equally important in prompting responses when they are relevant and before they are effective in shifting norms to expect them as standard.

The rest of the remaining work is social. Interpretation of our understanding of censorship is as much an art of persuasive argumentation and storytelling as one of data analysis. This translation to advocacy requires a commitment to collaboration between activists and

those technically knowledgeable, and presentation of information in ways that are informed by the needs of advocates. To this point, most of the interactions between the technical circumvention communities and activists have occurred because of the initiatives of activists who want to get better connectivity for themselves and the organizations they work with. This has resulted in some communities being very active in communicating their needs, while others facing similar obstacles are not well represented.

One concrete anecdote of this dissonance between advocacy and developers can be seen in the relative cases of Iran and China. The Iranian diaspora community is well connected to the technical training community, and through groups including ASL19, United for Iran, and ICHRI regularly interact with developers of circumvention tools. In contrast, the Chinese community has many fewer points of contact. Part of this is reflected in the tools which are popular in China. Two of these, goagent and shadowsocks, follow similar lines of thought to meek and protocol obfuscation work developed elsewhere and are differentiated by being developed by the largely separate Chinese development community. While they have enjoyed support from additional accessibility to the local audience, they have also suffered from legal intimidation of the Chinese developers involved.

6.2.2 Catching up with Throttling

A tactic for censorship and manipulating economic competitiveness that has emerged recently in China and Iran among other nation states performing advanced network manipulation is throttling of discouraged traffic. Either through choosing suboptimal routes, or by introducing latency or dropping some packets, networks are able to change the relative desirability of different services without imposing an absolute 'block' or fully deny access to those services.

These actions so far have gone unchecked. There is little infrastructure in place to identify that throttling or degrading is happening, or an easy ability to prove that those experiences have been caused intentionally. This lack of accountability makes the tactic attractive, but it is not inherent. Systems like OONI can run comparative analysis of different services to detect divergences from baselines, or provide evidence that a suspected service is indeed

performing below the level it should not because of its own negligence. Development of transparency in this area is the next step for censorship measurement systems going forwards.

6.2.3 Open Internet Structure

One of the stumbling points in Internet measurement, or in responding to clients accessing a site is a lack of open understanding of the underlying Internet structure. As pointed out in 3.2.4, it is possible to map IP addresses back to the ISP in control of that address space. Locating IPs back to countries or regions within countries is much harder. Large services like Google and CloudFlare create their own databases of estimated location based on their privileged position within the Internet core, while smaller systems must rely on MaxMind, one of very few services providing a (commercial) database of geo-location information. [136]

A major issue with the situation as it currently stands is that the databases we use are both incomplete, and impossible to verify. There are not good ways to independently verify the accuracy of the MaxMind database, or to recreate such a database from public knowledge. Our ability to geo-locate hosts on the Internet in particular is lacking in key areas that are important for understanding surveillance and censorship. The MaxMind data appears to largely be compiled from usage information for end users. Things like searches, geolocation in websites and applications, and the information provided by ISPs allow MaxMind to compile its database, but the data is much less precise towards web servers. One very visible aspect of this discrepancy is that the databases we have now will provide a single answer of 'where' an IP address is located. This disregards the growing use of anycast where a single IP address can be routed to multiple geographic physical machines, and the answers of where an IP address will be located is based on who is attempting to communicate with it. Even the database of which IP addresses use the anycast routing technique is not available or reasonable to create. Developing these databases from crowd-sourced open data is possible, but requires an explicit effort, and planning of which measurements need to be collecting in a coordinated way.

Beyond these issues of databases that do not yet exist, the Internet is also well along its

transition to IPv6 which comes with even more uncertainty. In the last several years, systems like Zmap have taken advantage of the fact that the IPv4 address space is tractable to provide meaningful insight into the composition of the Net. Zmap, combined with efforts like the Internet census, have given us a sense of the different devices directly connected to the net, how many web servers are out there, and provide us with fuel for external and side-channel measurements. This ability is greatly reduced in an IPv6 world, since blind scanning through the network is no longer feasible, and it is difficult to predict where machines will be located, even when limiting the search to the small fraction of the address space that is advertised by BGP routing announcements. Techniques to re-enable efforts like Zmap, perhaps based on passive analysis of advertised traffic, or other techniques to probe for devices will be critical in maintaining the transparency and continuing the measurement techniques that have been created so far.

6.2.4 Walled Garden Oversight

Marked by the fall of digital safe harbor in 2015, western content platforms like Facebook, YouTube, and Twitter have had to re-evaluate their limits on user expression as they face more liability for user speech. Google similarly continues to struggle with 'the right to be forgotten', a European policy regulating how historical information on individuals can be surfaced in search. The limitations imposed on speech by these platforms are real. Telegram and Twitter have reported closing groups around the Islamic state radical movement [106, 188], and watchdogs believe that Twitter has blocked access to millions of tweets from Turkey in the past year, significantly more than the 3000 they claim publicly [17].

As content platforms increase their regulation, and especially when that regulation is significantly automatic - with removal stemming from organic user complaints and from algorithmic similarity - we also need to ensure that transparency into the process is maintained. Currently, the political incentives for transparency are poorly structured. Companies find that their future growth is better ensured through a cooperative relationship with the countries they operate within, and have largely been cautious in opposing regulations of this sort.

Perhaps most notably are the case of Google Voice and other telecommunication providers, where users have been unable to challenge the searches of their data by local governments because the information was voluntarily turned over by their content providers without a subpoena [126].

The limited success so far in establishing oversight and accountability for these private actors have occurred in efforts like GNI, the global network initiative which pressured companies into better transparency reporting through a multi- stakeholder process, and through watchdog groups like EFF releasing comparative scorecards of corporate transparency. One normalized form of transparency has come in the form of transparency reports, e.g. [82, 188], in which companies regularly report on their interactions and removal of content at the request of different governments. It seems like a natural progression of this oversight for companies to normalize reporting of how much content is removed due to detection as spam, abuse, or banned content.

The alternative to holding platforms accountable through soft pressure tactics and through self-reporting of transparency reports are more adversarial approaches to measuring what's happening as a form of censorship. Projects like FreeWeibo monitor the Chinese Weibo micro-blogging platform for content removal, a measurement task that is very similar in nature to watching content deletion occur on Twitter or other public social networks. Removal of content within private groups is much more difficult to measure, but again the keyword censorship detection efforts to characterize the Chinese Great Firewall provide structure which can be adapted to expose the lines of content that is flagged for removal on western platforms.

6.3 Summary

This thesis has presented three new systems: Satellite, uProxy, and Activist. In doing so, we've shown a comprehensive strategy for understanding and circumventing online censorship without the need for current reliance on end users. While technical censorship is a highly contested area with pulls and influence from many other real-world phenomenon, it remains

possible to affect change from a purely technical basis. These systems are demonstration of this opportunity, demonstrating new techniques for the measurement, circumvention, and hardening in the context of network interference.

BIBLIOGRAPHY

[1] Josh Aas. Our milionth certificate, 2016. letsencrypt.org.

[2] Giuseppe Aceto and Antonio Pescapé. Internet Censorship Detection: A survey. *Computer Networks*, 2015.

[3] Weiwei Ai and Lee Ambrozy. *Ai Weiwei's Blog: Writings, Interviews, and Digital Rants, 2006-2009*. MIT Press, 2011.

[4] Akamai. The Akamai Intelligent Platform, 2016. akamai.com.

[5] M Akgül and M Kırlıdoğ. Internet censorship in Turkey. *Internet Policy Review*, 2015.

[6] Akka: Build powerful concurrent & distributed applications more easily. akka.io, 2011.

[7] Alexa, 1996. alexa.com.

[8] Ron Amadeo. Adware vendors buy chrome extensions to send ad- and malware-filled updates, 2014. arstechnica.com.

[9] Amnesty International. Turkey: Gezi Park protests: Brutal denial of the right to peaceful assembly in Turkey. amnesty.org, 2013. EUR 44/022/2013.

[10] Julia Angwin et al. AT&T helped U.S. spy on internet on a vast scale. The New York Times, August 2015.

[11] Anonymous. The Collateral Damage of Internet Censorship by DNS Injection. In *SIGCOMM Computer Communication Review*. ACM, 2012.

[12] Anonymous. Internet census, 2012. internetcensus2012.bitbucket.org.

[13] Anonymous. China, GitHub and the man-in-the-middle. GreatFire Analyzer, 2013.

[14] Anonymous. DNS census, 2013. dnscensus2013.neocities.org.

[15] Anonymous. Towards a Comprehensive Picture of the Great Firewall's DNS Censorship. In *Free and Open Communications on the Internet (FOCI)*. USENIX, 2014.

[16] Anonymous. Vpngate usage in china, 2014. Private communication.

[17] Anonymous. Twitter transparency report is misleading, 2015. Private communication.

[18] Marcus Anthony. The new China: Big brother, brave new world or harmonious society? *Journal of Futures Studies*, 2007.

[19] R. Arends, R. Austein, M. Larson, D. Massey, and S. Rose. DNS security introduction and requirements. RFC 4033, IETF, 2005.

[20] Simurgh Aryan, Homa Aryan, and J. Alex Halderman. Internet Censorship in Iran: A First Look. In *Free and Open Communications on the Internet (FOCI)*. USENIX, 2013.

[21] Taner Aydin. A P2P-based infrastructure for censorship-resistant web access. Technical report, Technische Universität Berlin, 2011.

[22] Baidu. Baidu liulanqi, 2015. `liulanqi.baidu.com`.

[23] James Ball. Internet anti-censorship tools are being overwhelmed by demand. The Washington Post, 2012.

[24] R. Barnes. Use cases and requirements for DNS-based authentication of named entities (DANE). RFC 6394, IETF, 2013.

[25] Andrew Baumann, Marcus Peinado, and Galen Hunt. Shielding applications from an untrusted cloud with Haven. In *Operating Systems Design and Implementation (OSDI)*. USENIX, 2014.

[26] Robert Beverly. Yarrp'ing the internet: Randomized high-speed active topology discovery. *arXiv preprint arXiv:1605.03999*, 2016.

[27] Robert Beverly and Steven Bauer. The Spoofer project: Inferring the extent of source address filtering on the internet. In *Proceedings of USENIX SRUTI workshop*. USENIX, 2005.

[28] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah. Gossip algorithms: design, analysis and applications. In *Computer Communications (INFOCOM)*. IEEE, 2005.

[29] Jon Brodkin. Iran reportedly blocking encrypted Internet traffic, 2012. `arstechnica.com`.

[30] Aaron Broodman. Stopping the gears, 2011. gearsblog.blogspot.com.

[31] Martin Brown. Pakistan hijacks YouTube. research.dyn.com, 2008.

[32] Bill Budington and Eva Galperin. Kazakhstan considers a plan to snoop on all Internet traffic, 2015. eff.org.

[33] Sam Burnett and Nick Feamster. Encore: Lightweight measurement of web censorship with cross-origin requests. *arXiv preprint arXiv:1410.1211*, 2014.

[34] Matthew Caesar, Miguel Castro, Edmund B. Nightingale, Greg O'Shea, and Antony Rowstron. Virtual ring routing: Network routing inspired by DHTs. In *SIGCOMM*. ACM, 2006.

[35] Matt Calder, Xun Fan, Zi Hu, Ethan Katz-Bassett, John Heidemann, and Ramesh Govindan. Mapping the expansion of Google's serving infrastructure. In *Internet Measurement Conference (IMC)*. ACM, 2013.

[36] Brian E Carpenter. Architectural principles of the Internet. RFC 1958, IETF, 1996.

[37] Steven Carstensen. Google's public DNS intercepted in Turkey, 2014. security.googleblog.com.

[38] China Internet Watch. Top web browsers in China, 2014. chinainternetwatch.com.

[39] The Chokepoint Project. chokepointproject.net.

[40] Danilo Cicalese, Diana Joumblatt, Dario Rossi, Marc-Olivier Buob, Jordan Augé, and Timur Friedman. A fistful of pings: Accurate and lightweight anycast enumeration and geolocation. In *Computer Communications (INFOCOM)*. IEEE, 2015.

[41] Ian Clarke, Oskar Sandberg, Brandon Wiley, and Theodore W Hong. Freenet: A distributed anonymous information storage and retrieval system. In *Privacy Enhancing Technologies (PETs)*. Springer, 2001.

[42] Richard Clayton, Steven J Murdoch, and Robert NM Watson. Ignoring the Great Firewall of China. In *Privacy Enhancing Technologies (PETs)*. Springer, 2006.

[43] Cloudflare advanced DDoS protection, 2013. cloudflare.com.

[44] CloudFlare. China network, 2014. cloudflare.com/china.

[45] Bram Cohen. Obfuscating BitTorrent. Live Journal, 2006.

[46] Jedidiah R. Crandall, Daniel Zinn, Michael Byrd, Earl Barr, and Rich East. ConceptDoppler: A weather tracker for Internet censorship. In *Computer and Communications Security (CCS)*. ACM, 2007.

[47] Masashi Crete-Nishihata et al. Asia Chats: Analyzing information controls and privacy in Asian messaging applications. Technical report, The Citizen Lab, 2013.

[48] Alberto Dainotti, Claudio Squarcella, Emile Aben, Kimberly C Claffy, Marco Chiesa, Michele Russo, and Antonio Pescapé. Analysis of country-wide internet outages caused by censorship. In *Internet Measurement Conference (IMC)*. ACM, 2011.

[49] Jakub Dalek, Katie Kleemola, Adam Senft, et al. A chatty squirrel: Privacy and security issues with uc browser. Technical report, The Citizen Lab, 2015.

[50] Ronald Deibert et al. Psiphon. `psiphon.ca`.

[51] Ronald Deibert and Rafal Rohozinski. Liberation vs. control: The future of cyberspace. *Journal of Democracy*, 2010.

[52] Department of Justice. Department of justice seizes more than $896,000 in proceeds from the online sale of counterfeit sports apparel, 2012. `justice.gov`.

[53] Alexis Deveria. Can i use... support tables for html5, css3, etc, 2010. `caniuse.com`.

[54] Roger Dingledine. Warning: 255 fake and booby trapped onion sites. tor-talk email list, 2015.

[55] Roger Dingledine, Nick Mathewson, and Paul Syverson. Tor: The second-generation onion router. In *USENIX Security*. USENIX, 2004.

[56] Marcel Dischinger, Massimiliano Marcon, Saikat Guha, Krishna P. Gummadi, Ratul Mahajan, and Stefan Saroiu. Glasnost: Enabling end users to detect traffic differentiation. In *Networked Systems Design and Implementation (NSDI)*. USENIX, 2010.

[57] David Dittrich, Erin Kenneally, et al. The menlo report: Ethical principles guiding information and communication technology research. *US Department of Homeland Security*, 2011.

[58] Lucas Dixon, Thomas Ristenpart, and Thomas Shrimpton. Network traffic obfuscation and automated internet censorship. *arXiv preprint arXiv:1605.04044*, 2016.

[59] Zakir Durumeric, Eric Wustrow, and J Alex Halderman. ZMap: Fast Internet-wide scanning and its security applications. In *USENIX Security*. USENIX, 2013.

[60] Kevin P Dyer, Scott E Coull, Thomas Ristenpart, and Thomas Shrimpton. Protocol misidentification made easy with format-transforming encryption. In *Computer and Communications Security (CCS)*. ACM, 2013.

[61] Kevin P Dyer, Scott E Coull, and Thomas Shrimpton. Marionette: A Programmable Network Traffic Obfuscation System. In *USENIX Security*. USENIX, 2015.

[62] Order compelling apple, inc. to assist agents in search. US District Court for the central district of California, February 2016. ED 15-0451M.

[63] Micha Elsner and Warren Schudy. Bounding and comparing methods for correlation clustering beyond ilp. In *Proceedings of the Workshop on Integer Linear Programming for Natural Langauge Processing*, pages 19–27. Association for Computational Linguistics, 2009.

[64] Steven Englehardt and Arvind Narayanan. Online tracking: A 1-million-site measurement and analysis. Technical Report Draft: May 18, 2016, Princeton University, 2016.

[65] Roya Ensafi, Jeffrey Knockel, Geoffrey Alexander, and Jedidiah R Crandall. Detecting intentional packet drops on the internet via TCP/IP side channels. In *Passive and Active Measurement*. Springer, 2014.

[66] Roya Ensafi, Philipp Winter, Abdullah Mueen, and Jedidiah R Crandall. Analyzing the Great Firewall of China over space and time. In *Privacy Enhancing Technologies (PETs)*. Springer, 2015.

[67] C. Evans, C. Palmer, and R. Sleevi. Public key pinning extension for HTTP. RFC 7469, IETF, 2015.

[68] Farsight Security, Inc. Farsight DNSDB, 2010. `dnsdb.info`.

[69] David Fifield. Summary of meek's costs, february 2016, 2016.

[70] David Fifield, Nate Hardison, Jonathan Ellithorpe, Emily Stark, Dan Boneh, Roger Dingledine, and Phil Porras. Evading censorship with browser-based proxies. In *Privacy Enhancing Technologies (PETs)*. Springer, 2012.

[71] David Fifield, Chang Lan, Rod Hynes, Percy Wegmann, and Vern Paxson. Blocking-resistant communication through domain fronting. In *Privacy Enhancing Technologies (PETs)*. Springer, 2015.

[72] Arturo Filastò and Jacob Appelbaum. OONI: Open observatory of network interference. In *Free and Open Communications on the Internet (FOCI)*. USENIX, 2012.

[73] Klint Finley. The Almost Completely Open Source Laptop Goes on Sale. *wired*, 2014.

[74] Abraham D. Flaxman. Algorithms and models for the web-graph. In William Aiello, Andrei Broder, Jeannette Janssen, and Evangelos Milios, editors, *Expansion and Lack Thereof in Randomly Perturbed Graphs*. Taylor & Francis, 2008.

[75] FLOSS Manuals. howtobypassinternetcensorship.org.

[76] The Squid Software Foundation. Squid: Optimising web delivery. squid-cache.org, 2011.

[77] Freedom house: Libya, 2013. freedomhouse.org.

[78] Sean Gallagher. After DNS change fails, Turkish government steps up Twitter censorship, 2014. arstechnica.com.

[79] Eva Galpernin. Iran ratchets up its internet censorship, 2012. eff.org.

[80] Phillipa Gill, Masashi Crete-Nishihata, Jakub Dalek, Sharon Goldberg, Adam Senft, and Greg Wiseman. Characterizing web censorship worldwide: Another look at the opennet initiative data. *ACM Transactions on the Web*, 2014.

[81] GlobalSign. Pki.js, 2014. pkijs.org.

[82] Google transparency report, 2011. google.com/transparencyreport.

[83] Internet Security Research Group. Let's Encrypt, 2015. letsencrypt.org.

[84] Open Rights Group. Blocked! report mobile and internet service providers blocking sites, 2011. blocked.org.uk.

[85] Bryan Gruley, David Fickling, and Cornelius Rahn. Kim Dotcom, pirate king, 2012. businessweek.com.

[86] Krishna P Gummadi, Richard J Dunn, Stefan Saroiu, Steven D Gribble, Henry M Levy, and John Zahorjan. Measurement, modeling, and analysis of a peer-to-peer file-sharing workload. In *SIGOPS Operating Systems Review*. ACM, 2003.

[87] Krishna P Gummadi, Stefan Saroiu, and Steven D Gribble. King: Estimating latency between arbitrary Internet end hosts. In *Internet Measurement Conference (IMC)*. ACM, 2002.

[88] Thor Halvorssen and Alexander lloyd. We hacked north korea with balloons and usb drives. The Atlantic, January 2014.

[89] Herdict web, 2008. herdict.org.

[90] Amir Houmansadr, Chad Brubaker, and Vitaly Shmatikov. The parrot is dead: Observing unobservable network communications. In *Security and Privacy (SP), IEEE Symposium on*, pages 65–79. IEEE, 2013.

[91] Freedom House. Freedom on the net, 2011. freedomhouse.org.

[92] Cheng Huang, Angela Wang, Jin Li, and Keith W Ross. Measuring and evaluating large-scale cdns. In *Internet Measurement Conference (IMC)*. ACM, 2008.

[93] Hurriyet Daily News. Approved article gives turkish gov't power to shut down websites in four hours, 2015. hurriyetdailynews.com.

[94] Hypestat, 2011. hypestat.com.

[95] ICANN. 2013 Registrar Accreditation Agreement. Technical report, ICANN, 2013.

[96] ICLab, 2013. iclab.org.

[97] Instant IO, Inc. PeerCDN, 2013. peercdn.com.

[98] Internews. Pluggable transports. pluggabletransports.info, 2015.

[99] Tomas Isdal, Michael Piatek, Arvind Krishnamurthy, and Thomas Anderson. Privacy-Preserving P2P Data Sharing with OneSwarm. In *SIGCOMM*. ACM, 2010.

[100] Peter C Johnson, Apu Kapadia, Patrick P Tsang, and Sean W Smith. Nymble: Anonymous IP-address blocking. In *Privacy Enhancing Technologies (PETs)*. Springer, 2007.

[101] Ben Jones, Sam Burnett, Nick Feamster, Sean Donovan, Sarthak Grover, Sathya Gunasekaran, and Karim Habak. Facade: High-Throughput, Deniable Censorship Circumvention Using Web Search. In *Free and Open Communications on the Internet (FOCI)*. USENIX, 2014.

[102] Ben Jones, Tzu-Wen Lee, Nick Feamster, and Phillipa Gill. Automated detection and fingerprinting of censorship block pages. In *Internet Measurement Conference (IMC)*. ACM, 2014.

[103] Matt Jones. Link shim - protecting the people who use facebook from malicious urls. Facebook Security Notes, 2012.

[104] George Kadianakis. obfs2 (The Twobfuscator). torproject.org, 2013.

[105] George Kadianakis. obfs3 (The Threebfuscator). torproject.org, 2013.

[106] Sarah Kaplan. Founder of app used by ISIS once said 'we shouldn't feel guilty.' on wednesday he banned their accounts. `washingtonpost.com`, 2015.

[107] Josh Karlin et al. Decoy Routing: Toward unblockable Internet communication. In *Free and Open Communications on the Internet (FOCI)*. USENIX, 2011.

[108] Karl Kathuria. Psiphon and the 2013 iranian election. `psiphon.ca`, 2013.

[109] Ethan Katz-Bassett, John P John, Arvind Krishnamurthy, David Wetherall, Thomas Anderson, and Yatin Chawathe. Towards ip geolocation using delay and topology measurements. In *SIGCOMM*. ACM, 2006.

[110] Jon M Kleinberg, Ravi Kumar, Prabhakar Raghavan, Sridhar Rajagopalan, and Andrew S Tomkins. The web as a graph: measurements, models, and methods. In *Computing and combinatorics*. Springer, 1999.

[111] Jeffrey Knockel and Jedidiah R. Crandall. Counting packets sent between arbitrary Internet hosts. In *Free and Open Communications on the Internet (FOCI)*. USENIX, 2014.

[112] Jeffrey Knockel, Jedidiah R Crandall, and Jared Saia. Three researchers, five conjectures: An empirical analysis of TOM-Skype censorship and surveillance. In *Free and Open Communications on the Internet (FOCI)*. USENIX, 2011.

[113] Jeffrey Knockel, Sarah McKune, and Adam Senft. Baidu's and don'ts: Privacy and security issues in baidu browser. Technical report, The Citizen Lab, 2016.

[114] Jeffrey Knockel, Adam Senft, and Ron Deibert. Wup! there it is: Privacy and security issues in qq browser. Technical report, The Citizen Lab, 2016.

[115] Daniel Kraft et al. Namecoin, 2012. namecoin.info.

[116] Christian Kreibich, Nicholas Weaver, Boris Nechaev, and Vern Paxson. Netalyzr: Illuminating the edge network. In *Internet Measurement Conference (IMC)*. ACM, 2010.

[117] Max Krohn and Chris Coyne. Keybase, 2013. keybase.io.

[118] Aparna Kumar. Libertarian, or just bizarro? Wired, 2001.

[119] H. T. Kung, Trevor Blackwell, and Alan Chapman. Credit-based flow control for ATM networks: credit update protocol, adaptive credit allocation and statistical multiplexing. In *SIGCOMM*. ACM, 1994.

[120] The Citizen Lab. test-lists: URL testing lists intended for discovering website censorship, 2014. github.com/citizenlab/test-list.

[121] Adam Langley. HSTS preload submission, 2014. hstspreload.appspot.com.

[122] Renai LeMay. Optus' filter can be defeated by 'trivial' dns change, 2011. delimiter.com.au.

[123] Chris Lesniewski-Lass and M. Frans Kaashoek. Whanaungatanga: Sybil-proof distributed hash table. In *Networked Systems Design and Implementation (NSDI)*. USENIX, 2010.

[124] Vincent Liu, Seungyeop Han, Arvind Krishnamurthy, and Thomas Anderson. Tor instead of ip. In *ACM Workshop on Hot Topics in Networks*. ACM, 2011.

[125] Ludde, uau, The_8472, Parg, and Nolar. Message Stream Encryption Protocol. Vuze Wiki, 2006.

[126] David Lyon. Surveillance, snowden, and big data: capacities, consequences, critique. *Big Data & Society*, 2014.

[127] M-Lab Research Team. ISP interconnection and its impact on consumer internet performance. Technical report, Measurement Lab, 2014. measurementlab.net.

[128] Rebecca MacKinnon. China's "networked authoritarianism". *Journal of Democracy*, 2011.

[129] Rebecca MacKinnon. *Consent of the Networked: The Worldwide Struggle for Internet Freedom*. Basic Books, 2013.

[130] Harsha V Madhyastha, Tomas Isdal, Michael Piatek, Colin Dixon, Thomas Anderson, Arvind Krishnamurthy, and Arun Venkataramani. iPlane: An information plane for distributed services. In *Operating Systems Design and Implementation (OSDI)*. USENIX, 2006.

[131] Gregor Maier, Fabian Schneider, and Anja Feldmann. NAT usage in residential broadband networks. In *Passive and Active Measurement*. Springer, 2011.

[132] Alexandre Y Mansourov. North Korea on the cusp of digital transformation. *Nautilus Institute*, 2011.

[133] Bill Marczak et al. China's Great Cannon. Technical report, The Citizen Lab, 2015.

[134] Morgan Marquis-Boire et al. Planet Blue Coat: Mapping global censorship and surveillance tools. Technical report, The Citizen Lab, 2013.

[135] Nick Mathewson et al. Next-generation hidden services in tor. Specification Proposal 224, The Tor Project, 2013.

[136] LLC MaxMind. Geoip, 2006. `maxmind.com`.

[137] Corynne McSherry. U.S. government seizes 82 websites: A glimpse at the draconian future of copyright enforcement?, 2010. `eff.org`.

[138] Ben Metcalfe. The .ly domain space to be considered unsafe, 2010. `benmetcalfe.com`.

[139] Alan Mislove, Massimiliano Marcon, Krishna P. Gummadi, Peter Druschel, and Bobby Bhattacharjee. Measurement and analysis of online social networks. In *Internet Measurement Conference (IMC)*. ACM, 2007.

[140] Alan Mislove, Ansley Post, Peter Druschel, and Krishna P. Gummadi. Ostra: leveraging trust to thwart unwanted communication. In *Networked Systems Design and Implementation (NSDI)*. USENIX, 2008.

[141] Abedelaziz Mohaisen, Aaram Yun, and Yongdae Kim. Measuring the mixing time of social graphs. In *Internet Measurement Conference (IMC)*. ACM, 2010.

[142] Hooman Mohajeri Moghaddam, Baiyu Li, Mohammad Derakhshani, and Ian Goldberg. Skypemorph: Protocol obfuscation for tor bridges. In *Computer and Communications Security (CCS)*. ACM, 2012.

[143] GC Moreira Moura, CHG Ganan, QB Lone, Payam Poursaied, Hadi Asghari, and MJG Van Eeten. How dynamic is the ISPs address space? Towards Internet-Wide DHCP churn estimation. In *Networking*. IFIP, 2015.

[144] Robert Moskowitz, Pekka Nikander, Petri Jokela, and Thomas Henderson. Host identity protocol. RFC 5201, IETF, 2008.

[145] Jared Muach. Open resolver project, 2013. `openresolverproject.org`.

[146] Gabi Nakibly, Jaime Schcolnik, and Yossi Rubin. Website-targeted false content injection by network operators. *arXiv preprint arXiv:1602.07128*, 2016.

[147] Daiyuu Nobori and Yasushi Shinjo. VPN gate: A volunteer-organized public vpn relay system with blocking resistance for bypassing government censorship firewalls. In *Networked Systems Design and Implementation (NSDI)*, pages 229–241. USENIX, 2014.

[148] Mark Nottingham. Unsanctioned web tracking. TAG finding, W3C, 2015. `w3.org`.

[149] MIIT of PRC. 关于计算机预装绿色上网过滤软件的通知 [notification regarding requirements for pre-installing green filtering software on computers], 2009. `miit.gov.cn`.

[150] OpenDNS. DNSCrypt, 2014. `opendns.com`.

[151] Opennet initiative, 2011. `opennet.net`.

[152] Openpgp.js, 2015. `openpgpjs.org`.

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

[154] Parliament of Estonia. Gambling Act. RT I 2008, 47, 261, 2008.

[155] Peer5. P2PXHR, 2013. `github.com/Peer5`.

[156] Tony Perry. Iran: Satellite dishes are illegal but oh-so-popular. LA Times, August 2008.

[157] Andreas Pitsillidis, Chris Kanich, Geoffrey M Voelker, Kirill Levchenko, and Stefan Savage. Taster's choice: a comparative analysis of spam feeds. In *Internet Measurement Conference (IMC)*. ACM, 2012.

[158] Ingmar Poese, Steve Uhlig, Mohamed Ali Kaafar, Benoit Donnet, and Bamba Gueye. IP geolocation databases: Unreliable? In *SIGCOMM Computer Communication Review*. ACM, 2011.

[159] Tor Project. Tor metrics protal. `metrics.torproject.org`.

[160] Tor Project. China blocking Tor, 2010. `blog.torproject.org`.

[161] Tor Project. Iran partially blocks encrypted network traffic, 2012. `blog.torproject.org`.

[162] Anirudh Ramachandran, David Dagon, and Nick Feamster. Can DNS-based blacklists keep up with bots? In *Conference on Email and Anti-Spam (CEAS)*. Microsoft Research, 2006.

[163] Anirudh Ramachandran and Nick Feamster. Understanding the network-level behavior of spammers. In *SIGCOMM Computer Communication Review*. ACM, 2006.

[164] Sylvia Ratnasamy, Paul Francis, Mark Handley, Richard Karp, and Scott Shenker. A Scalable Content-addressable Network. In *SIGCOMM*. ACM, 2001.

[165] NCC RIPE. Ripe atlas, 2010. `atlas.ripe.net`.

[166] J Rosenberg, R Mahy, P Matthews, and D Wing. Session traversal utilities for nat (stun). RFC 5389, IETF, 2008.

[167] Alex Russell, Jungkee Song, and Jake Archibald. Service workers. Working draft, W3C, 2015. `w3.org`.

[168] Peter Saint-Andre et al. A public statement regarding ubiquitous encryption on the xmpp network, 2014. `github.com/stpeter/manifesto`.

[169] Jerome H Saltzer, David P Reed, and David D Clark. End-to-end arguments in system design. *Transactions on Computer Systems (TOCS)*, 1984.

[170] Salvatore Scellato, Cecilia Mascolo, Mirco Musolesi, and Vito Latora. Distance matters: Geo-spacial metrics for online social networks. In *WOSN*. USENIX, 2010.

[171] Elliot Schrage. Testimony of Google Inc. before the subcommittee on asia and the pacific, and the subcommittee on africa, global human rights, and international operations. US House of Representatives. Hearing, February 2006.

[172] Will Scott, Thomas Anderson, Tadayoshi Kohno, and Arvind Krishnamurthy. Satellite: Joint analysis of CDNs and network-level interference. In *Annual Technical Conference (ATC)*. USENIX, 2016.

[173] S. Shalunov, G. Hazel, J. Iyengar, and M. Kuehlewind. Low extra delay background transport (LEDBAT). RFC 6817, IETF, 2006.

[174] Yuval Shavitt and Noa Zilberman. A geolocation databases study. *IEEE Journal on Selected Areas in Communications*, 2011.

[175] Fei Shen. *Encyclopedia of Social Media and Politics*, volume 2, chapter Great Firewall of China. SAGE, 2014.

[176] Shodan. shodan, 2013. `shodan.io`.

[177] Ryan Sleevi and Mark Watson. Web cryptography API. Last call WD, W3C, 2014. `w3.org`.

[178] Charlie Smith. Collateral freedom and the not-so-Great Firewall. GreatFire Analyzer, 2015.

[179] Charlie Smith. We are under attack, 2015. Private communication.

[180] Suemedha Sood. The statistics of studying abroad. BBC Travel, 2012.

[181] Neil Spring, Ratul Mahajan, and David Wetherall. Measuring ISP topologies with Rocketfuel. *SIGCOMM Computer Communication Review*, 2002.

[182] Daniel Stutzbach and Reza Rejaie. Understanding churn in peer-to-peer networks. In *SIGCOMM*. ACM, 2006.

[183] Srikanth Sundaresan, Walter de Donato, Nick Feamster, Renata Teixeira, Sam Crawford, and Antonio Pescapè. Broadband Internet Performance: A view from the gateway. In *SIGCOMM*. ACM, 2011.

[184] Aaron Swartz. Squaring the triangle: Secure, decentralized, human-readable names. `aaronsw.com`, 2011.

[185] Tao. Journey to the heart of internet censorship. Technical report, Reporters Without Borders, 2007.

[186] Nick Tattersall and Orhan Coskun. Furious reaction, political split after Turkey bans Twitter. *Reuters*, March 2014.

[187] Niels ten Oever and Avri Doria. IRTF Human Rights Protocol Consideration Research Group, 2015.

[188] Twitter transparency report, 2012. `transparency.twitter.com`.

[189] Guido Urdaneta, Guillaume Pierre, and Maarten van Steen. A survey of DHT security techniques. *ACM Computing Surveys*, 43(2), 2011.

[190] Amin Vahdat, Michael Dahlin, Thomas Anderson, and Amit Aggarwal. Active names: Flexible location and transport of wide-area resources. In *DARPA Active Networks Conference and Exposition*. IEEE, 2002.

[191] Anne van Kesteren. Cross-origin resource sharing. Recommendation, W3C, 2014. `w3.org`.

[192] Eugene Y. Vasserman, Rob Jansen, James Tyra, Nicholas Hopper, and Yongdae Kim. Membership-concealing overlay networks. In *Computer and Communications Security (CCS)*. ACM, 2009.

[193] John-Paul Verkamp and Minaxi Gupta. Inferring mechanics of web censorship around the world. In *Free and Open Communications on the Internet (FOCI)*. USENIX, 2012.

[194] Vasilis Ververis, George Kargiotakis, Arturo Filasto, Benjamin Fabian, and Afentoulis Alexandros. Understanding Internet Censorship Policy: The Case of Greece. In *Free and Open Communications on the Internet (FOCI)*. USENIX, 2015.

[195] Bimal Viswanath, Ansley Post, Krishna P Gummadi, and Alan Mislove. An analysis of social network-based sybil defenses. In *SIGCOMM Computer Communication Review*. ACM, 2011.

[196] Paul Vixie. Extension mechanisms for DNS (EDNS0). RFC 2671, IETF, 1999.

[197] W3C. W3C security activity, 2014. `w3.org/Security`.

[198] Qiyan Wang, Xun Gong, Giang TK Nguyen, Amir Houmansadr, and Nikita Borisov. Censorspoofer: Asymmetric communication using IP spoofing for censorship-resistant web browsing. In *Computer and Communications Security (CCS)*. ACM, 2012.

[199] Tao Wang, Xiang Cai, Rishab Nithyanand, Rob Johnson, and Ian Goldberg. Effective attacks and provable defenses for website fingerprinting. In *USENIX Security*. USENIX, 2014.

[200] chrome超360成国内使用人数最多的浏览器. Traffic Logs, 2013. webkaka.com.

[201] Zachary Weinberg, Jeffrey Wang, Vinod Yegneswaran, Linda Briesemeister, Steven Cheung, Frank Wang, and Dan Boneh. StegoTorus: a camouflage proxy for the Tor anonymity system. In *Computer and Communications Security (CCS)*. ACM, 2012.

[202] Mike West. Upgrade insecure requests. Working draft, W3C, 2015. w3.org.

[203] Mike West and Yan Zhu. Privileged contexts. Working draft, W3C, 2015. w3.org.

[204] WikiLeaks. Australian government admits less than 32% of secret censorship list is related to underage images, 2009. wikileaks.org.

[205] WikiLeaks. Australia bans reporting of multi-nation corruption case, 2014. wikileaks.org.

[206] Philipp Winter. *Measuring and circumventing Internet censorship*. PhD thesis, Karlstads universitet, 2014.

[207] Philipp Winter and Stefan Lindskog. How the Great Firewall of China is Blocking Tor. In *Free and Open Communications on the Internet (FOCI)*. USENIX, 2012.

[208] Damon Wischik, Costin Raiciu, Adam Greenhalgh, and Mark Handley. Design, implementation and evaluation of congestion control for multipath TCP. In *Networked Systems Design and Implementation (NSDI)*. USENIX, 2011.

[209] Sebastian Wolfgarten. Investigating large-scale Internet content filtering. Master's thesis, Dublin City University, Ireland, 2006.

[210] Joss Wright, Alexander Darer, and Oliver Farnan. Detecting internet filtering from geographic time series. *arXiv preprint arXiv:1507.05819*, 2015.

[211] Eric Wustrow, Scott Wolchok, Ian Goldberg, and J Alex Halderman. Telex: Anticensorship in the network infrastructure. In *USENIX Security*. USENIX, 2011.

[212] Bob Wyman. The persistence of identity (updating zooko's pyramid). wyman.us, 2006.

[213] Yinglian Xie, Fang Yu, Kannan Achan, Eliot Gillum, Moises Goldszmidt, and Ted Wobber. How dynamic are IP addresses? In *SIGCOMM Computer Communication Review*. ACM, 2007.

[214] Xueyang Xu, Z Morley Mao, and J Alex Halderman. Internet censorship in China: Where does the filtering occur? In *Passive and Active Measurement*, 2011.

[215] Haifeng Yu, Michael Kaminsky, Phillip B. Gibbons, and Abraham D. Flaxman. SybilGuard: defending against Sybil attacks via social networks. In *SIGCOMM*. ACM, 2006.

[216] R. Zafarani and H. Liu. Social computing data repository at ASU, 2009. `socialcomputing.asu.edu`.

[217] Radio Zamaneh. Report says Iranians spend nine hours a day on social media, 2015. `payvand.com`.

[218] Haiping Zheng et al. Regulating the Internet: China's Law and Practice. *Beijing Law Review*, 2013.

[219] Tao Zhu, David Phipps, Adam Pridgen, Jedidiah R Crandall, and Dan S Wallach. Tracking and quantifying censorship on a chinese microblogging site. *arXiv preprint arXiv:1211.6166*, 2012.

[220] Jonathan Zittrain and Benjamin Edelman. Empirical analysis of internet filtering in China, 2002. `cyber.law.harvard.edu`.