Addressing Denial of Service Attacks on Free and Open Communication on the Internet
— Interim Report 1 —

Antonela Debiasi, Roger Dingledine, Arthur Edelstein,
Alexander Færøy, Matthew Finkel, David Goulet,
Kat Hanna, Maggie Haughey, George Kadianakis,
Iain R. Learmonth, Alison Macrina, and Maria Xynou
kat@torproject.org

Tor Tech Report 2018-11-001
November 19, 2018

Contents

1 Introduction

Article 19 of the Universal Declaration of Human Rights asserts the right for all people "to seek, receive and impart information and ideas through any media regardless of frontiers." [55] And indeed, around the world, people are trying to safely speak out about their situations, and safely learn what others have to say. This free and open communication on the internet is critical to the growth of strong communities and societies. As the American Library Association wrote in 1953, “the suppression of ideas is fatal to a democratic society." [4]

At the same time, nation-state censors and others attempt to *deny* this communication, threatening the health and safety of communities. In this project, we leverage our past work on the Tor anonymity system and the pluggable transport ecosystem to explore and analyze techniques to detect, understand, and mitigate these denials of service. In this first report, we present the current state-of-play in these areas, as well as some research directions and plans for future work. In subsequent reports, we will describe our findings and our progress with work on the project.

This report is organized as follows: the remainder of this section gives a brief overview of Tor and describes how it is used to evade censorship; Section 2 describes pluggable transports and how Tor Browser uses them; Section 3 discusses applications that currently use the Tor network as well as some challenges to using popular apps in censored locations; Section 4 discusses onion routing client performance on networks with resource constraints; Section 5 outlines how we measure key metrics about the Tor network, including censorship of Tor; Section 6 describes some attacks on the Tor network and how we defend against them; and Section 7 discusses building a stronger relationship between the Tor Project and academic researchers.

1.1 About Tor

The Tor anonymity system [16] protects internet users from tracking, surveillance, and censorship. The Tor network is made up of thousands of volunteer-run *relays*—servers that are usually located in data centers—distributed across the world that enable users to make private connections to services on the internet. Currently, the vast majority of connections to the Tor network are made using the Tor Browser. But a growing number of applications use the Tor network (see Section 3.1 for a list), and we expect that many more will do so in the future.

When Tor Browser starts up, it connects to a single relay in the Tor network, known as the *guard* relay. The connection to the guard is always encrypted, and all web connections are passed through this fully-encrypted connection so that third parties (such as an ISP) observing the user’s internet connection cannot see which website the user is visiting.

Then, for each connection to a public website, Tor builds a tunnel through the guard relay and two other randomly-chosen Tor relays. Connections from Tor Browser thus pass, encrypted, through three Tor relays before exiting from the Tor network onto the public internet. The way the encryption works means that the guard relay can see the address of the computer making the request, but not the website the user is visiting; the middle relay can see the address of the guard relay and the address of the third relay (called the *exit relay*); and the exit relay can see the address of the middle relay as well as the destination website.

A Tor connection through a guard relay, a middle relay, and an exit relay is called a Tor *circuit*. Tor relays are usually part of many Tor circuits and relay traffic for many Tor users at the same time.

Tor’s *directory authorities* are a group of trusted servers that keep track of all of the relays that are currently able to help users protect their privacy. Directory authorities periodically publish a list of the addresses of these available relays, and applications such as Tor Browser choose relays from this list when they connect to the Tor network.

1.2 Tor and censorship

Governments and other censors use a variety of tools to deny people access to the free and open internet. As a result, systems for getting around censorship, including Tor, must adopt a variety of measures as well.

Because the connection between Tor Browser and the guard relay is encrypted, parties observing a user's internet connection (such as an ISP) cannot selectively block connections based on their content or destination. Some governments and ISPs have responded to this censorship evasion by attempting to prevent users from connecting to the Tor network in the first place.

One way to do this blocking is to prevent people from downloading the Tor Browser by blocking connections to the Tor Project website and its mirrors. We responded to this concern by launching the *GetTor service* [29], which allows users to make a request via email or twitter. The service responds with a link to a cloud storage site that the user can download Tor Browser from.

Another way to block access to the Tor network is to block connections to addresses published in the publicly available list of Tor relays. Tor *bridges* are Tor relays that are not in the public list, which makes blocking them more difficult. How to distribute bridge addresses to users while keeping them from the censors is discussed in Sections 2.1.1 and 2.2.1.

More sophisticated censors use *deep packet inspection* (DPI) to recognize and block connections. This method examines the data moving across the internet and tries to determine whether it's Tor traffic. To counter this blocking, Tor uses *pluggable transports*. Pluggable transports disguise the traffic between the user and a Tor bridge so that it doesn't look like Tor traffic. Pluggable transports are discussed in Sections 2.1.2 and 2.2.1.

2 Better Pluggable Transports for Actively Censored Regimes

2.1 Bridge distribution and pluggable transports

The Tor network is an open network. Tor directory authorities periodically publish information on the state of the network, including a list of the addresses of all of the relays that are available for use. This information must be available so users can use the network, but its availability allows various actors, such as ISPs or nation states, to prevent users from connecting to the Tor network. It also allows service providers (like websites) to block requests coming *from* the Tor network.

Tor bridges are Tor relays that are not listed in the public directory. This makes it hard for censors to block connections to them. Of course, because we can't publish a list of bridge addresses, we need another way for users that need them to discover them.

Using a bridge is often not enough to evade censorship. Some censors use deep packet inspection (DPI) to prevent people from using Tor. DPI is a technique that examines data as it moves across the internet and tries to determine what kind of traffic it is, for example, regular web traffic, encrypted web traffic, video streaming, or Tor. Pluggable transports try to solve this problem by disguising the Tor traffic between the user and the bridge so that it doesn't look like Tor traffic.

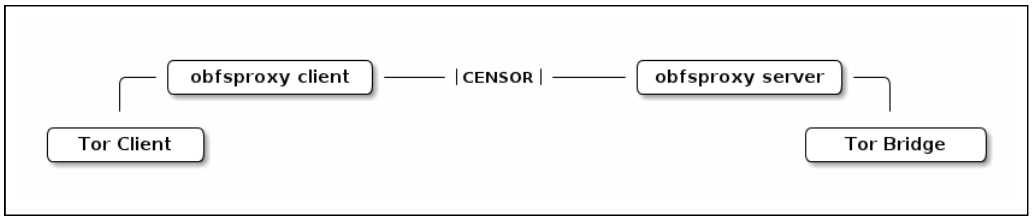


Figure 1: Pluggable transport example using obfsproxy

2.1.1 Bridge distribution

Because bridge addresses are not listed in the public directory of relays, applications that use Tor need another way to find them. Tor Browser comes pre-configured with a few dozen bridge addresses that might help in places where Tor is blocked. But since anyone can download Tor Browser and see the list, censors can block these bridges along with blocking the publicly listed relays.

If the bridges that come with Tor Browser don't work, an option for finding other bridge addresses is *BridgeDB*. Users can access BridgeDB at https://bridges.torproject.org/ using an ordinary web browser. The user solves a CAPTCHA and is given some bridge configuration lines, which include the bridge address. A user can also request bridges via email. This option is useful if a censor blocks the BridgeDB website. The request must be made from a Yahoo, GMail, or Riseup email address and sent to
bridges@torproject.org.

The BridgeDB service partitions the bridges that it gives out, with the goal of limiting the rate at which attackers can discover bridges. Specifically, an attacker who wants to enumerate many bridges needs to interact with the web interface from many different IP addresses around the internet, and needs to interact with the email interface from many different email addresses.

When requesting bridges from BridgeDB, either over the web or by email, a user can also specify additional information, such as whether they need a specific pluggable transport or require addresses that work over IPv6.

There are several drawbacks to the BridgeDB solution: the user has to manually make the request; the user may not know whether they need a pluggable transport or an IPv6 address; the user has to manually enter the bridge configuration lines into a Tor Browser configuration dialog.

Tor Browser 8.0, released in September of 2018, includes the ability to request bridge addresses from within the application. More information about the way Tor Browser helps get around censorship is in Section 2.2.1.

2.1.2 Pluggable transports

A pluggable transport is a program that is used between the user's Tor client, such as Tor Browser, and a Tor bridge. The pluggable transport's job is to change the appearance of the data that flows between itself and the bridge so that a censor won't be able to recognize it as Tor traffic. See Figure 1 for an example.

The pluggable transport API [93] enables developers and research groups to invent pluggable transports that are resistant to different types of censorship. Then a pluggable transport can be chosen based on the conditions a user is experiencing in their location. The availability of *goptlib*, a Go programming language library that takes care of all the communication between the Tor client and the pluggable transport, has made the Go language very popular for developing pluggable transports [24].

The following sections describe some pluggable transports that either have been used by Tor, or that have the potential be used by Tor in the future.

2.1.2.1 Obfsproxy Obfsproxy makes Tor traffic look like a random stream of bytes. This type of pluggable transport is known as a *look-like-nothing* pluggable transport and works best when censors have a list of protocols that they try to recognize and then block. There have been three versions of obfsproxy.

obfs2

Obfs2 was the first pluggable transport that the Tor Project developed. Inspired by the Obfuscated OpenSSH project, obfs2 simply transforms Tor traffic into a random stream of bytes. It uses a naive protocol for exchanging encryption keys, which makes it vulnerable to passive attacks that can reconstruct the keys and then recognize the obfuscation.

obfs3

Obfs3 is another older pluggable transport very similar to obfs2. obfs3 uses Diffie-Hellman, a better key exchange protocol, which makes it immune to passive attacks that try to deduce the encryption keys.

obfs4

All of the pluggable transports before obfs4 were vulnerable to a class of active attacks called *active probing attacks*. A censor can connect to a bridge and pretend to be a client using a pluggable transport. If the bridge replies with pluggable transport traffic, then the censor can tell that it's a bridge and blocks it immediately. This tactic has been used by China to detect and block bridges [106].

Obfs4 is the first deployed pluggable transport protocol to defend against this attack. Obfs4 requires that the client prove it knows a secret before the bridge replies. This means obfs4 can resist active probing attacks.

Obfs4 is one of the most popular pluggable transports in use today.

2.1.2.2 ScrambleSuit ScrambleSuit, like obfsproxy, is a pluggable transport that makes traffic look like a random stream of bytes. But ScrambleSuit also changes the lengths of its data packets, as well as the time between sending them, to confuse censors that rely on those characteristics to recognize Tor traffic.

2.1.2.3 FTE Format Transforming Encryption (FTE) was one of the first pluggable transports that did not belong in the look-like-nothing group. Instead, FTE uses an encryption scheme to make Tor traffic look like regular http web traffic, i.e. it looks like the user is just visiting unencrypted websites [48].

This type of pluggable transport is useful in the face of a "whitelisting" censor, that is, a censor that has a list of protocols that it recognizes and allows.

2.1.2.4 Flashproxy

Flashproxy [25] is a pluggable transport that allows censored users to quickly find and use short-lived proxies operated by volunteers around the globe. The strength of the system is that these proxies can be operated by a volunteer simply visiting a website—no technical skill is required. Because the user's Tor client connects to one of these browser proxies rather than a Tor bridge, there is no need to find bridge addresses. Flashproxy hides its traffic within Websocket, a common web protocol.

A major weakness of Flashproxy is that the vast majority of censored users are subject to *network address translation* (NAT) which means they can not accept incoming connections from the internet. While it's fine for the volunteer flashproxies to be behind NAT (because they make outgoing connections to both the censored users and the Tor network, to glue them together), the limitation that the censored users need to be directly reachable on the internet drastically reduces the usefulness of the design.

See Section 2.1.2.7 for a description of Snowflake, an improved version of Flashproxy that solves this problem.

2.1.2.5 Meek

Meek [23] is a pluggable transport that uses *domain fronting* [26] to get around censorship. In domain fronting, a client makes a connection to an allowed website and that website redirects the connection to a censored site that is in the same domain. This technique is generally used within the domains of large cloud service providers. Meek hides its traffic within encrypted web requests and responses between it and the allowed website.

There are two major problems with domain fronting. Cloud providers charge for all of the traffic that goes into and out of their networks, so it is expensive to deploy. It also requires the cooperation of the owner of the domain, which makes it susceptible to political pressure on domain owners. For example, until April 2018, Amazon and Google supported domain fronting; they disabled it, reportedly in response to pressure from the Russian government [80]. Currently, the only place where Meek is deployed is the Microsoft Azure cloud.

So far, meek has proven resilient to blocking.

2.1.2.6 Marionette

Marionette is a fairly new pluggable transport developed by the US company RedJack. Marionette's design [18] is similar to that of FTE. Marionette can handle a relatively large amount of traffic quickly and can be used to make Tor traffic look like traffic from a variety of different protocols. For now, Marionette is focused on making Tor traffic look like regular web traffic.

Marionette is written in Google's Go programming language and uses the goptlib library.

Marionette is in the late development stage. The next step is to deploy it to a set of stable bridges so that we can evaluate it in a variety of censored locations.

2.1.2.7 Next generation pluggable transport: Snowflake

The most promising next generation pluggable transport is Snowflake [82]. Snowflake is similar to Flashproxy in that it does not require the user to find bridges, and it connects the user's Tor client to a short-lived proxy. Like Flashproxy, the proxies are run by volunteers visiting a web page.

Snowflake hides its traffic within the WebRTC protocol—the same protocol that Google Hangouts uses. The advantage of using WebRTC is that it has the ability to allow incoming connections to computers using it even if they are subject to NAT.

Snowflake is a work in progress. Although we have a working prototype in the alpha releases of Tor Browser for Linux and Mac OSX, there are challenges to getting it working on Windows. In addition, there are several outstanding research questions, including understanding what is missing from the code base, understanding whether a system like Snowflake does indeed work in denied countries; understanding barriers to getting pluggable transports like Snowflake working on Android too, since an increasing percentage of the world is moving from desktop to mobile.

The Guardian Project [89] currently has a grant to begin to answer some of these questions in the context of Android and Orbot. They also hope to work on enabling any Android device, such as Chromebooks, Android TVs, and other home devices to act as volunteer Snowflake bridges.

The Tor Project is in the process of hiring a software developer to explore some of these questions and to work on other anti-censorship technologies as well.

2.1.3 Designing new pluggable transports

A common anti-censorship strategy is to make it unappealing for censors to use a certain blocking method because of *collateral damage*—allowed content that will also be blocked. For example, domain fronting works because most censors don't want to block access to large cloud providers. A variation on this theme is to mimic traffic that is too important to block, such as web traffic or WebRTC traffic.

In designing new pluggable transports, we need to consider how a censor might attempt to block them, as well as what collateral damage might result from that blocking. Adversary Lab [1] is a service that analyzes captured network traffic to identify statistical properties. Using this analysis, developers can create filtering rules to block sampled traffic, and learn how to improve a transport so that blocking it would maximize collateral damage.

2.2 Tor Browser and pluggable transports

2.2.1 Tor Browser

In many places in the world, censorship of the web is common. Users are routinely blocked from connecting to websites by ISPs, governments, corporate firewalls, and public access points such as internet cafes. Websites are shut down, logs of users' connections are seized, and users can be punished for their individual browsing history.

Tor Browser is designed to combat this systemic assault on the rights of individuals. Uniquely among all available web browsers, Tor Browser empowers millions of users around the world to bypass censorship and read any website they want, without interference.

Since the project began in 2008, as the Tor Browser Bundle, Tor Browser has evolved from a bundle of related projects into a clean, streamlined, and usable web browser. At its core, Tor Browser is a modified version of Mozilla Firefox, providing more privacy protections than the standard browser [88]. Fortunately, with Mozilla's support, many of the modifications written for Tor Browser are now also integrated into Firefox [95].

Sometimes censors attempt to prevent users from downloading and installing Tor Browser by blocking access to the Tor website. Our GetTor service provides a way for users to get Tor

Browser from another site. GetTor was developed by an intern participating in Google's Summer of Code. Due to resource constraints, it is not currently maintained and does not automatically check for new versions of Tor Browser and so sometimes delivers out of date versions. An auto-update mechanism is planned that will ensure that users who contact the service will always get the latest version of Tor Browser.

Tor Browser 8.0 provides a new feature called *moat* to make it easy for a user to request bridge addresses from within the application. Moat presents the user with a CAPTCHA to solve, then Tor Browser receives additional, unpublished, bridge addresses over a secure encrypted connection. Tor Browser automatically configures itself to use the new bridge addresses.

The bridges Tor Browser provides use pluggable transports to evade blocking based on deep packet inspection. The pluggable transports currently used are obfs4, FTE, meek (through the Microsoft Azure cloud), and obfs3. For descriptions of the types of pluggable transports see Section 2.1.2.

Although using tools to get around censorship currently requires some level of technical knowledge, we have made improvements to help users connect to the internet from censored locations. We have been working with local speakers of non-English languages to find metaphors that can help explain complex concepts. As a result of this user-focused effort, Tor Browser is now available in 25 languages. For more information about our international user testing for Tor Browser 8.0, see Section 2.2.1.1.

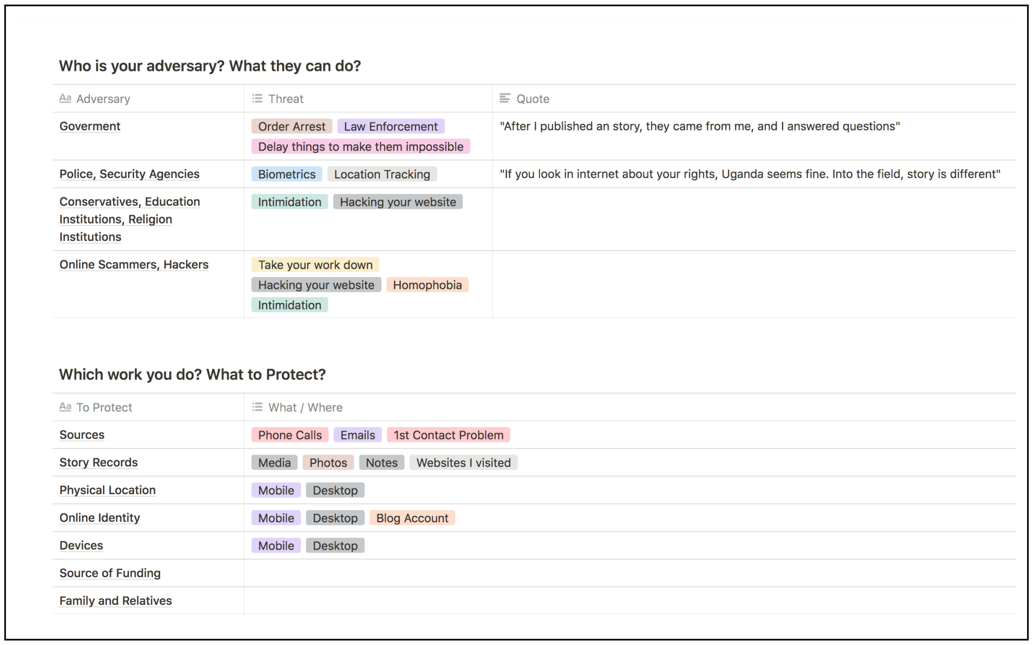
We are doing ongoing work to make it easier for users to bypass censorship. We are in the planning stages for a much simpler interface to provide fully automated bridge setup for users behind national firewalls. We expect this work to make anti-censorship technologies much more widely accessible to users.

Measures within Tor browser that keep people safe are often especially relevant to users under censorship. Our reproducible build process makes it so that anyone can (and should!) verify that the software we release is based on the source code we say it is. An independent party can take our source code and the specifications for our build environment and build Tor Browser. All of the resulting packages should be identical, byte for byte, to the software we release. This is an important way to verify the integrity of our software and thus increase user safety.

Another feature that enhances user safety is the domain isolation feature (this is sometimes referred to as "first party isolation"). Browser tabs that are connected to one internet domain run in a separate environment from tabs connected to another domain. Each domain has its own circuit through the Tor network, and cookies and other trackers from each domain are stored separately from those of other domains. These separate environments make it difficult for websites or tracking networks to link activity between domains. This means that it is safe, for example, to be logged into Facebook in one tab and to post anonymously to a blog in another tab. This is a Tor feature that Mozilla has also implemented in Firefox.

2.2.1.1 Usability testing and global outreach To make useful software, especially software that aims to keep people safe, it is essential to understand users' contexts and goals. This is especially true for tools that require some level of technical knowledge to use successfully, such as tools for evading censorship.

As part of our global south initiative, members of the Tor user experience and community



Who is your adversary? What they can do?

Adversary	Threat	Quote
Goverment	Order Arrest, Law Enforcement, Delay things to make them impossible	"After I published an story, they came from me, and I answered questions"
Police, Security Agencies	Biometrics, Location Tracking	"If you look in internet about your rights, Uganda seems fine. Into the field, story is different"
Conservatives, Education Institutions, Religion Institutions	Intimidation, Hacking your website	
Online Scammers, Hackers	Take your work down, Hacking your website, Homophobia, Intimidation	

Which work you do? What to Protect?

To Protect	What / Where
Sources	Phone Calls, Emails, 1st Contact Problem
Story Records	Media, Photos, Notes, Websites I visited
Physical Location	Mobile, Desktop
Online Identity	Mobile, Desktop, Blog Account
Devices	Mobile, Desktop
Source of Funding	
Family and Relatives	

Figure 2: Threat Model.

teams traveled through India, Uganda, Colombia, and Kenya to run usability testing. We ran in-person testing of several user experience (UX) improvements in Tor Browser 8.0 with users of different technical backgrounds. We conducted more than 45 interviews [96] that gave us an understanding of the diverse mental models and technical levels of knowledge of our users.

We met people like Juana, a coffee farmer in Colombia who is part of a self-managed group of women coffee growers. The group uses Tor to communicate securely with one another. We met Jon, an environmental activist and journalist in Hoima, Uganda who uses Tor to anonymously publish his blog.

We learned about several threat models (Figure 2) of the people we spoke to in Uganda. Concern about local police retaining hardware devices, and the practice of the political party currently in power seizing journalists notes and other records, were common in most of these communities.

In addition, this work allowed us to speak with people who use our software under extreme conditions, such as poor telecommunications infrastructure, very expensive data packages or very old hardware. This helped us to understand their context, empathize, and consider solutions that work for them. We believe that Tor Browser must be usable by people without a technical background, as well as by advanced users. We want to empower our users through education so that they can have control of their browsing.

We use an open design process [70] where developers and designers work together to achieve the best solution. We use knowledge gained from our research to help us to create the most usable flow for users.

As part of user education, Tor Browser's on-boarding—a short in-browser orientation provided when a user first opens the application—aims to teach users the complex concepts underlying the Tor network using local language metaphors and simple words.

In many places, Tor Browser is considered to be the highest standard for protecting people's privacy and helping them evade censorship. However, many people who learn about Tor express fear, at first, based on stories they have heard or misconceptions they have. Our on-boarding aims to give our users confidence by explaining how Tor's privacy and security measures can protect them. All of the information we give to users is especially critical when users are in censored environments. They need to understand the trade-offs and the risks they may run when they decide to opt in or opt out of a feature.

After the user downloads Tor Browser, installs it successfully, and first starts it, if they are in a censored country, they need to make decisions about using bridges and pluggable transports. But not all of our users know whether they are in a censored location. An important subject for future work is exploring the technical options for detecting whether the location is censored without compromising user safety. We could probe the internet from Tor Browser to detect whether the user's connection is being censored and then automatically configure bridges and pluggable transports. This would certainly improve Tor Browser's usability. However, such probing may be detectable by the censor and alert them to the fact that the user has Tor Browser installed, which may not be desirable. Currently, Tor Browser errs on the side of safety and does not attempt to establish any connections until the user configures it.

In the future, with the help of OONI (see Section 5.2) and Tor Metrics (see Section 5.1), we should have a good idea of the state of censorship and network interference for most countries. When we know this, we can provide users with suggestions about how to set up their Tor Browser to get around censorship based on their locations.

Tor Browser 8.0 also includes a new circuit display (Figure 3), which shows the relays the current browser tab is using. This visual aid helps improve user understanding of how Tor sends their traffic through the Tor network.

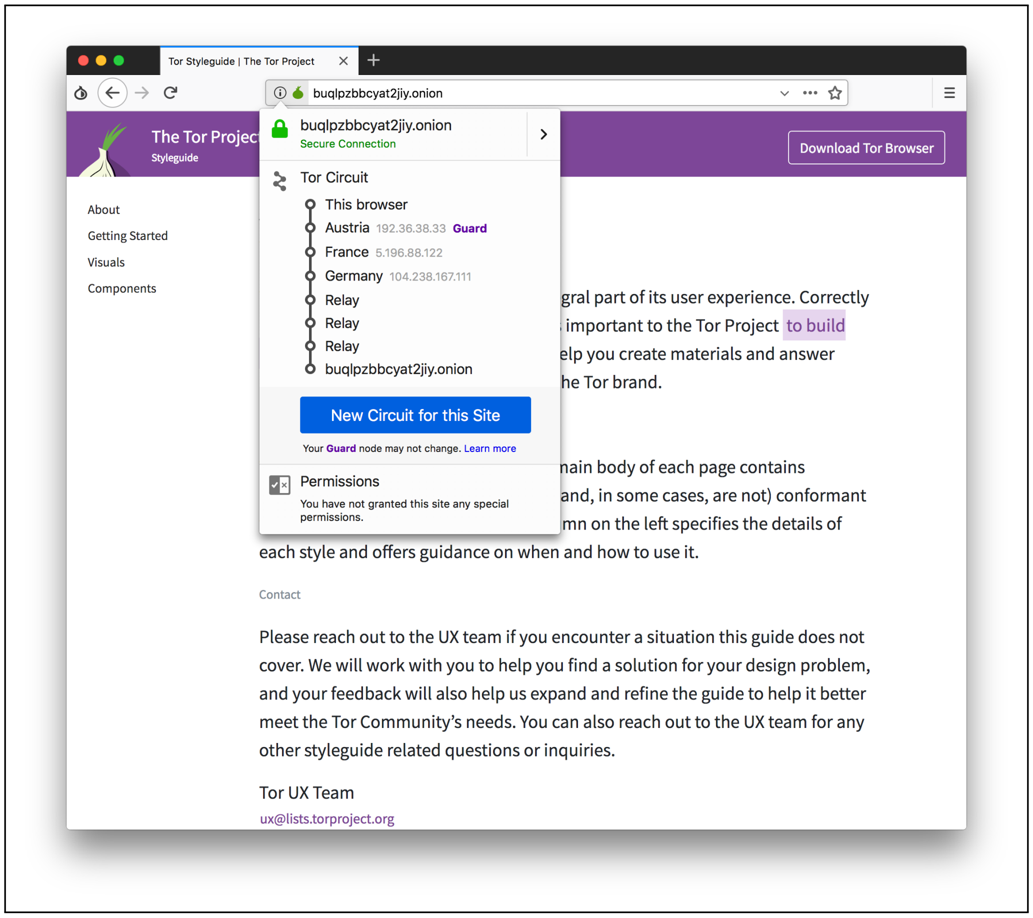
2.2.2 Tor Browser for Android

Historically, Tor Browser has only been developed for desktop computers running Microsoft Windows, Mac OS X, and various distributions of Linux. But most people now use mobile devices, either as their primary means of getting on the internet, or at least in addition to using desktop computers. In the past, Tor relied on partner organizations to make mobile browsers that connect to the Tor network. Recently, however, we've begun making significant progress on Android support ourselves.

At the beginning of September 2018, we released the first alpha version of Tor Browser for Android. This work is heavily based on the Guardian Project's Android app Orfox. Orfox provides a Tor-connected private web browser on Android, but doesn't come with all of the privacy protections that are in the desktop version of Tor Browser. The goal of our recent work in this area is to provide an Android app that gives users a private browser on Android equivalent to Tor Browser on desktop.

The current alpha version of Tor Browser for Android still requires an additional app, Orbot, that connects the browser to the Tor network. In the near future, Tor Browser for Android will include Tor as part of the app, making it easier to install and use.

Tor Browser for Android provides an on-boarding experience for new users that is similar to the one provided for desktop. This helps new users understand how Tor works to protect their privacy.



buqlpzbbcyat2jiy.onion
Secure Connection

Tor Circuit

This browser
Austria 192.36.38.33 Guard
France 5.196.88.122
Germany 104.238.167.111
Relay
Relay
Relay
buqlpzbbcyat2jiy.onion

New Circuit for this Site

Your Guard node may not change. Learn more

Permissions

You have not granted this site any special permissions.

gral part of its user experience. Correctly
important to the Tor Project to build
elp you create materials and answer
he Tor brand.

nain body of each page contains
and, in some cases, are not) conformant
mn on the left specifies the details of
each style and offers guidance on when and how to use it.

Contact

Please reach out to the UX team if you encounter a situation this guide does not cover. We will work with you to help you find a solution for your design problem, and your feedback will also help us expand and refine the guide to help it better meet the Tor Community's needs. You can also reach out to the UX team for any other styleguide related questions or inquiries.

Tor UX Team
ux@lists.torproject.org

Figure 3: New Circuit Display

The current version of the app does not directly support methods for circumventing censorship—we rely on Orbot for this, as well. Similar to Tor Browser on desktop, Orbot supports the meek and obfs4 pluggable transports. These provide users with some of the best censorship circumvention techniques available today.

When a user configures Orbot to use a language that is primarily used in certain censored countries, it automatically configures and enables the meek pluggable transport to help the user get around censorship.

The user can always select and configure another bridge from Orbot's home screen, if necessary. This screen gives users four options:

- If a bridge is already configured, then the user can choose to directly connect to the Tor network without using a bridge.
- The user can enable one of the built-in obfs4 bridges (called *community servers*).
- The user can enable one of the built-in meek bridges (called *cloud servers*).
- The user can request another bridge from BridgeDB at bridges.torproject.org. The user would then need to configure the bridge manually using Orbot's Settings menu.

In the future, like Tor Browser for desktop, Orbot and Tor Browser for Android will support *moat* to make requesting and configuring bridges and new pluggable transports easy.

When Tor Browser for Android can connect to the Tor network without the assistance of Orbot, we still envision the two being able to help each other, perhaps by sharing configuration information. For example, if a user already has Orbot installed to connect a different app to the Tor network, when they install Tor Browser for Android, it could automatically configure itself to use the bridges that Orbot uses.

Our goal is to make Tor Browser for Android support all of our existing methods to get around censorship. As research and development continue, we will add new pluggable transports like Snowflake. We hope to learn from the work the Guardian Project is doing in this area, as described in Section 2.1.2.7.

As we do further work on Tor Browser for Android to include the features that are already in Tor Browser for desktop, we plan to implement many of the user interface improvements discussed in Section 2.2.1. But because mobile displays are very different from desktop browsers, some of these, such as the circuit display that shows the user which relays the browser is using, will need to be redesigned.

We also plan to add an indicator that will let users know when they are connected to a website running an onion service, so they can tell that their connection is more secure than a connection to the regular web.

One subject of ongoing investigation is browser fingerprinting. The information the browser exposes to websites can be used to identify either an individual user or a group of users. Mobile devices have more physical sensors than stationary computers, and that exposes more potential pieces of information from which to construct a fingerprint.

3 Beyond Web Browsing

Users, especially those on smartphones, want privacy and the ability to bypass censorship for more than just web browsing. Millions of people use chat and social media applications, create and share media files like images and video, and more. In fact, since Tor's performance penalties are most visible for delay-sensitive applications like web browsing, we would like to explore secure messaging and other asynchronous applications. We want to consider more use cases than just web browsing.

3.1 Existing apps that use Tor

As a first step, we would like to understand whether existing prototypes and tools currently using Tor work with Youtube and other popular services. These tools include messaging and email apps, secure operating systems, tools that send traffic from existing apps over the Tor network, monitoring tools, and file sharing apps.

3.1.1 Web browsers

In addition to Tor Browser, there are a number of web browsers that use the Tor network and protect users' privacy to varying degrees. These include OnionBrowser [56], Brave [5], and Cliqz[9].

3.1.2 Using mobile apps with the Tor network

The following apps send their internet traffic over the Tor network, whether by using Tor's proxy interface, as Orbot does, or by providing a "system-wide VPN" interface as iCepa does.

3.1.2.1 Orbot Orbot [71] is an open-source app that allows other apps to send their internet traffic through it and then through the Tor network, thereby hiding the user's IP address. It includes an expert mode that allows users to make all network connections through Tor. However, it requires user caution as it does not provide any additional anti-tracking measures – it is trivial for apps or websites to track users in most web browsers and many other apps even if the IP address is hidden.

3.1.2.2 iCepa iCepa [34] is a new experimental app for connecting iOS apps to the Tor network. iCepa is in alpha.

3.1.3 Communications

From peer to peer messaging apps, to email clients, there are a number of different applications that people can use to anonymize their communications:

3.1.3.1 Briar Briar [6] is an open source peer-to-peer messaging app for Android. Briar sends its messages through the Tor network, preventing third parties from observing which parties are communicating with each other or revealing their IP addresses.

3.1.3.2 Ricochet Ricochet [78] is a desktop app that offers peer-to-peer messaging, without exposing information about who is communicating. All messages are sent wholly inside the Tor network. Ricochet uses Tor onion services to connect users to one another, eliminating the central server that most messaging apps use.

3.1.3.3 ChatSecure ChatSecure [7] is an open source iOS app offering optional Tor support that hides the device's IP address and allows messages to bypass restrictive firewalls.

3.1.3.4 TorBirdy TorBirdy [97] is an extension for Thunderbird, a desktop email client, that is developed and maintained by members of the Tor Project. TorBirdy sends email from Thunderbird over the Tor network. This means that the email servers a message passes through do not learn the user's IP address. It also hides email metadata to protect users' privacy. TorBirdy is in beta.

3.1.4 Secure operating systems

Currently there are a number of Linux-based operating systems which aim to provide anonymity and privacy to their users. These OSes send their internet traffic through the Tor network. These include Tails (The Amnesic Incognito Live System) [83], Whonix [105], and Qubes [76].

3.1.5 Monitoring tools

3.1.5.1 OONI Probe OONI Probe [68] is an app that runs on Mac OS X, Linux, Android, and iOS that allows users to run tests to determine if their internet connection is being censored or otherwise experiencing interference. OONI is a Tor project that monitors and reports on censorship around the world. See Section 5.2.1 for more information.

3.1.5.2 Haven Haven [32] is an Android app that uses device sensors to monitor a physical space. It detects motion, sound, vibration and light, and watches for unexpected guests and intruders. It offers a Tor onion service feature that allows the operator to connect to the device remotely and view its event logs and captured media. Haven is currently in beta.

3.1.6 File sharing

There are a few applications currently available which allow anonymous file sharing over the Tor network:

3.1.6.1 OnionShare OnionShare [58] is a desktop app that allows users to send files to one another over the Tor network. This prevents observers from seeing who is sending and receiving the files.

3.1.6.2 GlobaLeaks

GlobaLeaks [30] is software that enables people and organizations to set up their own secure, anonymous whistleblower platforms. When a user creates a new deployment, GlobaLeaks sets up a Tor onion service. People submitting documents can use Tor Browser to maintain anonymity when connecting to the platform.

There are a wide variety of GlobaLeaks deployments around the world for purposes such as investigative journalism, anti-corruption activism, transparency activism, and anti-crime activism.

3.1.6.3 SecureDrop

SecureDrop [81] is server software that allows whistleblowers to send files anonymously and securely to journalists. It functions as a Tor onion service and requires whistleblowers to use Tor Browser to submit materials. Over the past few years, it has become very common for major news organizations to deploy SecureDrop. The New York Times, The Associated Press, The Guardian, and ProPublica are just a few of the organizations that use SecureDrop.

3.1.6.4 OpenArchive

OpenArchive is a new app for Android for uploading and sharing media, including video. It sends its traffic over the Tor network to the Internet Archive and other archives. OpenArchive looks promising, but we need to examine whether Open Archive solves the problems that actual users have and assess its usability at accomplishing these goals.

3.2 Applications for users under censorship

We conducted a survey of the landscape of apps popular among users behind censorship, and thought about how we can serve those users, considering both safety and usability. Understanding what apps are popular with these users can help us understand what would be useful to build, and what apps we should try to make work with Tor. Internet censorship in China is particularly extensive, so if we consider solutions for users there, the lessons we learn may also apply in other contexts.

We began with the assumption that most users in most censored places would want to be using popular Android apps, since Android is the world's most popular mobile operating system and users in most places will have the same usage habits as one another. Popular apps include messaging apps, social media apps, app stores, and video streaming and sharing apps [45].

Many of these very popular apps such as WhatsApp, Telegram, Signal, Facebook, Twitter, Netflix, Twitch, and Periscope are blocked in China [85]. In China, TenCent [86] provides its own apps for messaging, social media, photo sharing, music streaming, ebooks, e-sport streaming, and more [87]. It is widely understood that Chinese government surveillance of these apps is pervasive [31].

All of these types of services could benefit from a Tor-based offering to protect privacy and fight censorship. There are quite a few mobile VPNs that claim to do this, but many don't do it well or safely. Virtual private networks (VPNs) are very popular in China for circumventing censorship, but Chinese authorities have taken steps to ban their use [103], with varying degrees of success.

An obvious challenge is how to make apps and their updates available in places where standard app stores such as Google Play and F-Droid are themselves censored.

Another important consideration is that people in China might not switch to Tor to connect to a website or app because that website or app, as well as Tor's documentation, lacks sufficient localization in Mandarin or Cantonese. How do we address this problem with localization in mind? Do we simply assume that people will generally switch over and add Chinese content back to the larger web? Do we accept that we may only be serving English-speaking users in China to some degree? Realistically we only have control over our own products, and making translation apps usable through Tor, but it's an important thing to bear in mind when trying to answer the question "can we make this popular app work with Tor for users in China?"

4 Performance Under Resource Constraints

Connections through the Tor network are usually slower than direct connections between a user and a destination on the internet. A big part of this can be explained by not enough resources being available on Tor relays, but other factors are also at play [17], and can be particularly severe for users with bad internet connections [3]. In addition, load on the network itself, including load created by an attacker or load targeted at particular relays [72], can affect the performance of the whole system. We need to better understand all of the factors that contribute to poor performance for our users.

4.1 Performance in resource constrained access networks

In the more developed world, recent advances in wired broadband networks and cellular networks have led to faster speed when connecting to the internet. But everyone has not seen these improvements. Connections over old DSL infrastructure have hardly improved in over a decade, and some people can't even get cellular data service. There is a large gap between the fastest and the slowest service—a gap that continues to deepen [33].

In the developed world, people usually had wired internet access first and then got mobile (cellular) internet devices later. But some less developed countries are skipping the construction of wired broadband networks altogether. This is cost-effective because a single cell phone tower can provide service to hundreds of customers.

According to the Egyptian Ministry for Communications and Information Technology, there were 29.84 million internet users in Egypt in 2016 [35], or about 37.8% of the population. That number breaks down into 28.65 million mobile internet subscribers and 4.6 million ADSL subscribers (the primary wired broadband technology in Egypt), with some presumable overlap between mobile and wired broadband users. The story is similar in other parts of the less developed world.

Mobile internet access has some unique challenges beyond the existing challenges that users of wired internet face. These generally relate to the variability of available bandwidth and delay.

Performance issues encountered in access networks are usually related to bandwidth, latency (the delay between when a data packet is sent and when it is received), data packet loss, and variations in the delays between data packets (jitter). In some cases there may also be issues with out-of-order delivery of data packets or intermittent connectivity.

Tor uses the *transmission control protocol* (TCP) for connections from the Tor client to the guard relay, connections between the relays in the circuit, and the connection from the exit to the destination. TCP provides both *congestion control* and *flow control*. Congestion control mechanisms manage the amount of traffic that enters the network to keep it from being overloaded. Flow control mechanisms limit the amount of traffic sent to the amount that the receiver is able to handle.

Ways to measure the maximum throughput—the rate of successful message delivery—for a TCP connection over a given access network are well understood [11], however this is only one part of the Tor circuit. There are multiple TCP connections in a Tor circuit, which makes measurement more difficult. Initial work on measuring the performance of Tor from an end-user perspective is detailed in Section 5.1.4.

When relays or bridges are under an attack intended to create a denial of service, bandwidth is reduced, data packet loss rises, and latency and jitter increase. Until connectivity is completely lost, the Tor protocol and the pluggable transports used with Tor should ideally be able to degrade gracefully by adapting to the changing network conditions.

4.2 Improving client performance

We plan to improve client performance both for users who have bad network connections and also to ensure that performance degrades gracefully when the network is under attack.

4.2.1 End-to-end performance

Network characteristics such as latency, bandwidth, data packet loss, and jitter are highly interrelated. For example, in TCP connections such as those used by Tor, increased latency can slow the time it takes until TCP allows the available bandwidth to be fully used. Data packet loss and out-of-order delivery of data also increase latency, as it increases the time before the software that receives the data can put it back together correctly.

Tor has traditionally relied on end-to-end flow control to avoid overloading clients and servers, but has not had a good solution for controlling congestion at the Tor relays. There are many locations in a Tor relay where data must be queued before it can be sent, and this can increase the latency in the network. These locations include network hardware, operating system interfaces and the Tor software.

4.2.1.1 Congestion control Researchers have explored alternative designs for congestion and flow control in order to decrease the delays introduced by uncontrolled congestion in Tor. One example [3] proposes to restrict the amount of data that is traveling through the network at any given time to a small, fixed amount per client. Another idea [3] is to vary that per-client amount based on observed network performance on the Tor circuit.

A third example [3] uses a fresh approach to congestion and flow control inspired by standard techniques from Asynchronous Transfer Mode (ATM) networks. Here, Tor relays explicitly inform their neighbors of how much traffic they can accept before becoming congested. This reduces unnecessary delays and memory consumption.

Analysis has shown that the first two approaches offer up to 65% faster web page response times relative to Tor's current design. However, they are overly conservative in the amount of traffic they allow into the network, resulting in slower download times. In contrast, analysis of experiments with the third proposal show that web clients experience up to 65% faster web page responses and a 32% decrease in web page download times compared to Tor's current design—though this proposal removes per-stream (end to end) flow control, so it can't be deployed without further design work. These results show that there is clearly room for improvement, although further research is needed before we decide what to implement.

4.2.1.2 Improved socket management Congestion does not only occur within Tor relay software or within the networks that join them together; it can also occur in the operating system that the Tor software runs on. When the Tor relay software decides to send data on a Tor circuit, the operating system will sometimes queue the data before it sends it. Some analysis [39] has indicated that congestion occurs almost exclusively inside of the operating system's outgoing queues, dwarfing delays caused by the Tor software itself.

This work went on to develop a new algorithm, KIST, to manage which circuit's data to process at a given time. It exposes the operating system's queues to the relay software, allowing Tor to choose circuits whose data will be sent immediately, rather than being queued by the operating system. That is, when the operating system tells the Tor software that many sockets are writeable, but most of those sockets secretly already have a lot of data queued inside the operating system, it is not possible for the Tor software to make good queuing choices.

KIST reduced circuit congestion by over 30%, reduced network latency by 18%, and increased network throughput by nearly 10% in experiments. This improved algorithm has been included in the Tor relay software since version 0.3.2.

4.2.2 Resumable uploads

Some users may have access networks that provide intermittent connectivity. This can occur even with good infrastructure when, for example, a user switches between their cellular data connection and a local Wi-Fi connection. For web browsing, where many small requests (for web pages, for example) are made, this sort of dropped connectivity may be noticeable, but recovery happens relatively quickly. However, if a user is half way through uploading a large file and the connection drops, it can be very irritating. For users under duress, the situation may be more serious. They may have a limited timeframe during which they can upload a file and having to begin all over again may prevent them from doing it.

JavaScript libraries, such as Resumable.js [77], which uses the HTML 5 File API [44], provide resumable upload functionality for web applications to help address this problem. In the future, a helper function could be added to Tor Browser to assist in resuming large uploads or downloads in the event that connectivity is interrupted. If transfers resume automatically this may even be transparent to the user.

4.2.3 Striping connections

Bridges providing access to the Tor network via pluggable transports often have lower bandwidth available than regular relays. To help in load-balancing across available bridges and avoid wasting capacity, and also to improve end-user performance, user traffic can be split across multiple bridges. Conflux [2] is one such approach to dynamic traffic-splitting that assigns traffic to a Tor circuit based on its measured latency.

Experiments have shown an improvement of approximately 30% in expected download time for web browsers that use multiple Tor bridges and for streaming video applications. This method introduces tradeoffs between users' anonymity and performance. As with congestion control however, there is clearly room for improvement. Further research is needed to decide on a path forward.

4.3 Performance while under attack

There are several ways to mount a Denial of Service (DoS) attack against the Tor network. This section is an overview of the range of possible ways the network could be attacked to cause resource exhaustion.

The following is heavily inspired by the 2015 Tor technical report "Denial-of-service attacks in Tor: Taxonomy and defenses" [51]. Section 6.2 describes some recent attacks on the Tor network.

4.3.1 Types of DoS attacks

4.3.1.1 CPU consumption Most of Tor's running time is spent performing expensive cryptographic computation. It requires a lot of processing power on the CPU (central processing unit—the computer's main processor) to accomplish these tasks. If an attacker can cause relays or clients to use a lot of processing power on other things, it could render Tor unable to accomplish its real tasks.

Countermeasures:

- Use more efficient cryptographic algorithms.
- Rewrite parts of the software so that more tasks can run on multiple processors at a time.
- Modify our protocols to be more resistant to attacks that use up processing power.
- Improve the code that schedules data to be sent from relays so that one user's traffic is less likely to interfere with another's.
- Instrument the code so that we can better detect abnormal patterns of processor use.

4.3.1.2 Memory consumption Tor relies on having adequate computer memory available to function well. Exhausting the available memory on a server running a relay leads to performance degradation and sometimes crashes.

Countermeasures. There are many places in the Tor code that could benefit from memory optimization: using smaller objects in memory, cleaning up more frequently, optimizing access to shared or common data, and so on.

4.3.1.3 Disk space consumption

These days, we rarely see servers—or even clients—with very low disk space, but disk space is still very relevant for certain parts of Tor. For example, directory authorities store various data about all of the relays in the network on disk and use it to create the Tor consensus.

Countermeasures. Detect and respond to low disk space events. Tor may not be the cause of available disk space becoming low, but it is important that we ensure that Tor can keep running, even with no available space at all.

4.3.1.4 Bandwidth consumption

Attacks that disrupt Tor's ability to move data between the client, the relays in a circuit, and a destination by consuming a lot of bandwidth are dangerous and when combined with other attacks can lead to de-anonymization [53, 20].

The simple version of this attack is expensive to mount because it requires the amount of bandwidth that the attacker is trying to consume on the target. This could be enough to interfere with one relay or many relays.

However, there are amplification attacks that, at very low cost, can force the target relay to use more bandwidth than the attacker expends. These are rare, but high-impact [42].

Countermeasures:

- Implement better cell scheduling. We currently use the KIST scheduler, which is better than the earlier simple round-robin design, but KIST aims to work well under normal conditions, and a scheduling approach that remains robust to intentional congestion remains an open research problem.
- Improve our bandwidth measurement system. There is currently work underway to replace our old, poorly maintained bandwidth scanner, TorFlow [73], with a new Simple Bandwidth Scanner (SBWS) [99]. PeerFlow [43], a system proposed in the academic literature, is a possibility for the future.
- Create a new Tor circuit and migrate a client's traffic to it if performance on the old circuit is poor.
- Optimize the onion services protocol to avoid bandwidth consumption attacks. Proposal 255 [101], which would allow load balancing an onion service across a pool of servers, is one possibility.

4.3.1.5 Network resources exhaustion

TCP connections are specified with a port and IP address. There are 65535 available ports. If an attacker opens, and keeps open, connections to most or all of the ports on a relay, it will become unusable.

Countermeasures. We currently have denial of service countermeasures in place in the Tor relay software that aim to prevent such an attack.

We could also explore using UDP, another transport protocol that doesn't suffer from this potential problem.

It might also be possible to use IPv6, which has a much larger range of addresses, to mitigate this problem.

4.3.2 Future directions

To improve Tor's resilience against Denial of Service attacks, we should favor defenses that address multiple types of attacks simultaneously over defenses that simply address one at a time.

Therefore, we should look for general solutions and process improvements, rather than piecemeal responses to individual attacks as they are discovered.

The technical report cited at the beginning of this section lists a series of recommendations that are still relevant today. However, it's worth highlighting one strategy in particular that has increasingly become the focus of our development work in the past year or so: modularity. Rewriting the Tor code to separate different functionality into different modules will allow much finer grained control of how resources are used. It will also help prevent interference between modules.

5 Network and Censorship Measurement

Ongoing, robust network measurement is essential in order to respond to censorship events, to adapt Tor Browser and other apps to respond to changing network conditions, and to validate changes to the Tor network software.

Section 5.1 describes the information that the Tor Metrics team collects about Tor users and the Tor network; Section 5.2 describes information that the Open Observatory of Network Interference (OONI) [27] and the Tor Networks team collect about censorship of the Tor network.

5.1 Tor Metrics public datasets

Tor Metrics [47] archives historical data about the Tor ecosystem, collects data from the public Tor network and related services, and helps develop novel approaches to safe, privacy preserving data collection.

Tor relays and bridges collect aggregated statistics about the bandwidth they use and the numbers of connecting clients per country. The relays protect user privacy by discarding IP addresses and only reporting country information from a local database that maps IP address ranges to countries. The relays aggregate these statistics and send them to the directory authorities periodically.

CollecTor [10] archives the latest server descriptors, "extra info" descriptors containing the aggregated statistics, and consensus documents from the directory authorities. This archive is public, and researchers can use the metrics-lib Java library [91] to perform analysis of the data.

In order to provide easy access to visualizations of the archived historical data, the Tor Metrics website [90] contains a number of customizable plots to show user, traffic, relay, bridge, and application download statistics over a requested time period for a particular country.

5.1.1 Relay and bridge user counts

The number of Tor users is one of our most important statistics. It is vital to know how many people use the Tor network on a daily basis, whether they connect via relays or bridges, from which countries they connect, what protocols (transports) they use, and whether they connect via IPv4 or IPv6.

Because Tor is an anonymity network, we cannot collect identifying data to learn the number of users. So we don't actually count users, but rather requests to the directories or bridges that clients make to update their list of relays in the network. We estimate user numbers from this information.

The result is an average number of concurrent users, estimated from data collected over a day, but not how many distinct users there are. It is not possible to know whether the same set of users stays connected over the whole day, or if that set leaves after a few hours and a new set of users arrives. However, our main interest is finding out if usage changes, so it is not critical to estimate absolute user numbers.

Relay users are users that connect directly to a relay in order to connect to the Tor network; bridge users connect to a bridge as an entry point into the Tor network.

5.1.2 Safe counting and PrivCount

Recent advances in privacy-preserving metrics collection such as private set-union cardinality [22] and PrivCount [41] can help to improve our user metrics.

Private set-union cardinality is useful in many settings. It can determine how many unique users are using Tor, how many unique users connect via a particular version of client software, and how many unique destinations are contacted. It could also be used to determine how users regularly connect to the network (e.g. over mobile connections, through proxies, etc.) and how the network is used (e.g. the length of time that users spend on the network in a single session).

PrivCount implements a privacy-preserving scheme for distributed measurement using secure multiparty computation. It provides a method of collecting network-wide aggregate statistics without revealing the totals provided by each individual relay. By adding carefully calculated noise, this scheme can also provide differential privacy guarantees.

Very recent work [49] has enhanced PrivCount and Private set-union cardinality and used them to measure numbers of Tor users. The results show a much higher number of users than we have previously estimated: closer to eight million daily users rather than the early count of two million daily users. We need to closely examine the methodologies used in this work to validate its findings.

5.1.3 Reliable geolocation

In producing metrics relating to countries, for example the top countries by daily users shown in Table 1, Tor Metrics uses IP geolocation databases. We currently use the MaxMind GeoLite2 City database [28], but a strong reliance on a single source of this data is undesirable. For calculating metrics relating to relays—where IP addresses are already public—we also look up autonomous system (AS) information in these databases to better understand the diversity of the network (as discussed further in Section 6.1).

One alternative for country code lookups is to use a database from IP2Location [38]. We have not yet explored the benefits of making this change and would be unlikely to be able to distribute this database as we currently do with the MaxMind database.

Country	Mean daily users
United States	367814 (17.70 %)
United Arab Emirates	257893 (12.41 %)
Russia	244418 (11.76 %)
Germany	153335 (7.38 %)
France	105256 (5.06 %)
Indonesia	95731 (4.61 %)
Ukraine	77508 (3.73 %)
United Kingdom	62134 (2.99 %)
India	45891 (2.21 %)
Netherlands	42741 (2.06 %)

Table 1: The top-10 countries by estimated number of directly-connecting clients between the 17th July and the 15th October 2018. These numbers are derived from directory requests counted on directory authorities and mirrors. Relays resolve client IP addresses to country codes locally using an IP geolocation database.

One alternative for AS number resolution is RIPEstat [79] which uses BGP feeds to provide near-live information. In a comparison study by Tor Metrics on the 1st July 2018 [36], there were 269 relays where there was disagreement between RIPEstat and the latest MaxMind database and 7889 in agreement ($\kappa = 0.979$ excluding relays for which either MaxMind or RIPEstat had no AS number). There were 101 relays for which MaxMind did not return an AS number and 2 relays for which RIPEstat did not return an AS number.

The MaxMind database is distributed to users as part of the Tor software, and it can be used for example to disable or prefer the use of exit relays in specific countries. So it may be dangerous for users if they get mixed information about the autonomous system numbers or country codes assigned to relays. It may be equally dangerous to incorrectly assign country codes, but without ground truth to compare to, it is not possible to say whether a switch would improve this situation or not.

We hope to analyze the different databases and feeds available to us to determine which options best fit our requirements.

5.1.4 Network performance

The performance that Tor users experience depends on many factors and is the subject of current research. In order to evaluate progress in improving Tor's performance, we need to continuously measure how fast Tor really is for our users.

In June 2009 we began longitudinal measurements of Tor performance using the Torperf [98] tool. It used a Tor client with the same path selection algorithms as an ordinary user would use and performed downloads of a 50 KB file, a 1 MB file, and a 5 MB file. The initial findings from these measurements [46] and the ongoing measurements, visualized in Figure 4, provided a feedback loop for developers and those looking after the network to understand the effects that changes could have for the key performance characteristics: throughput and latency.

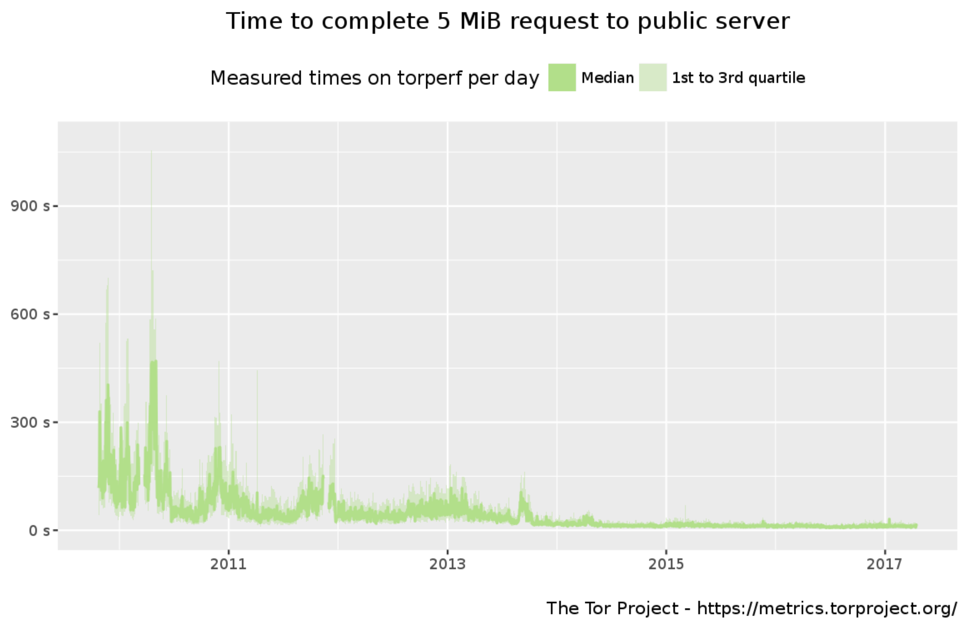


Figure 4: Overall performance when downloading a 5 MB static file over Tor from a server on the public internet between July 2009 and April 2017. The graph shows the range of measurements from first to third quartile, and highlights the median. The slowest and fastest quarter of measurements are omitted from the graph.

OnionPerf [57] is a modern rewrite of Torperf that includes more fine-grained detail in the measurements. OnionPerf has the ability to measure the performance of onion services from a user perspective in addition to the traditional measurement using a server on the public Internet. In April 2017, Tor Metrics began using OnionPerf to perform measurements of the Tor network, replacing Torperf. We currently convert OnionPerf's output to a format compatible with Torperf, so Tor Metrics does not yet handle all the new data that is captured.

We have, however, introduced new latency visualizations to the Tor Metrics website using OnionPerf data. For web browsing and other interactive applications, latency can be as important as throughput in improving the quality of end-user experience. Figure 5 shows the average latencies measured for circuits since Tor Metrics began running OnionPerf.

The Tor Metrics OnionPerf measurements are currently performed using vantage points in 3 data centers of a single network operator. In the future it is important that these vantage points are diversified to enable a view of the network that is closer to that of a real user. More realistic vantage points could include residential (DSL or DOCSIS) or cellular (3G or LTE) access networks. Further, to understand the performance for users in denied countries, performance measurements should be carried out using pluggable transports.

5.2 Detecting censorship of Tor

The Tor network offers online anonymity, privacy, and censorship circumvention. Human rights defenders rely on the Tor network to secure their communications, circumvent the blocking of

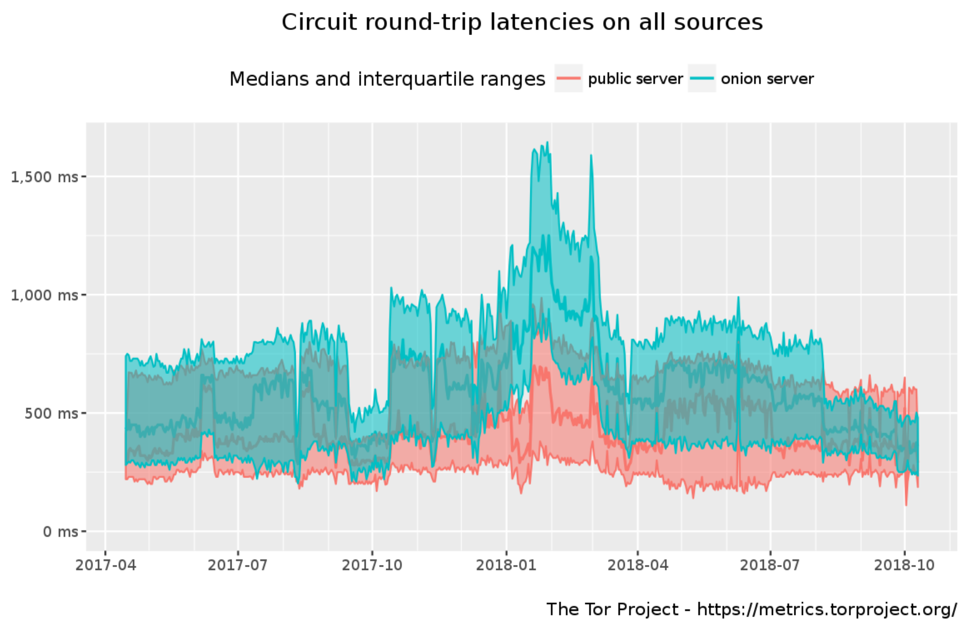


Figure 5: Round-trip latencies of circuits used for downloading static files of different sizes over Tor, either from a server on the public internet or from a version 2 onion server. Round-trip latencies are measured as the time between sending the HTTP request and receiving the HTTP response header. The graph shows the range of measurements from first to third quartile, and highlights the median. The slowest and fastest quarter of measurements are omitted from the graph.

social media apps during protests, and to securely report on corruption and other abuses of power. As a result, the Tor network has become a target of censorship by several governments around the world. In such countries, users can circumvent the blocking and connect to the Tor network through the use of bridges. In some cases though, bridges are blocked too.

The Open Observatory of Network Interference (OONI) [27] performs active measurement to detect censorship. Through the Tor Metrics described in the previous section, we track the expanding usage of Tor and Tor bridges globally. These metrics allow us to infer when access to the Tor network is blocked, particularly when we see spikes in Tor bridge usage. In addition to our measurement projects, we also receive alerts on Tor blocking from our community members.

In most countries around the world, the Tor network appears to be accessible. In recent years, however, we have (mainly) observed the blocking of the Tor network in the following countries: Egypt, Ethiopia, China, Iran, Turkey, and Venezuela.

5.2.1 Open Observatory of Network Interference

Over the last several years, we have been monitoring the blocking of the Tor network through our censorship measurement software, OONI Probe [68], which includes tests [69] designed to measure the reachability of the Tor network and of Tor bridges from the vantage point of the user. Every month, hundreds of thousands of measurements are collected and published [59] from thousands of networks in more than 200 countries [66]. These measurements enable us to track the blocking of the Tor network around the world.

The “vanilla Tor" test built in to OONI Probe is the primary means of detecting censorship of the Tor network. The vanilla Tor test attempts to make a connection to the Tor network. If the test successfully establishes a connection within a predefined number of seconds (300 by default), then Tor is considered to be reachable from the vantage point of the user. But if the test does not manage to establish a connection, then the Tor network is likely blocked within the tested network. This test was central to the analyses in the upcoming sections.

There is also a “Tor Bridge Reachability" test that examines whether Tor bridges work in tested networks. This test runs Tor with a list of bridges and if it’s able to connect to them successfully, we conclude that Tor bridges are not blocked in the tested network. If the test, however, is unable to make a connection, then the Tor bridges are either offline or blocked.

OONI Probe requires volunteers to run measurements and those volunteers must be located in the target countries. To engage volunteers, OONI has established a Partnership Program [67] to collaborate with local digital rights organizations on censorship measurement research. Over the last several years, OONI has established 25 partnerships, many of which involve digital rights organizations in countries where Tor is blocked, such as Egypt and Iran. We will continue to expand our community engagement activities to monitor the blocking of Tor and to make it more resilient to censorship.

5.2.1.1 Egypt Following the blocking of hundreds of media websites [21], over the last year, more and more Egyptians began to use censorship circumvention technologies. Egyptian ISPs then started blocking numerous circumvention tools, including the Tor network.

Last July, OONI published a research report [19] documenting Tor blocking. This study includes an analysis of all network measurements [61] collected from Egypt between January 2017 to May 2018. These measurements suggest that access to the Tor network was blocked,

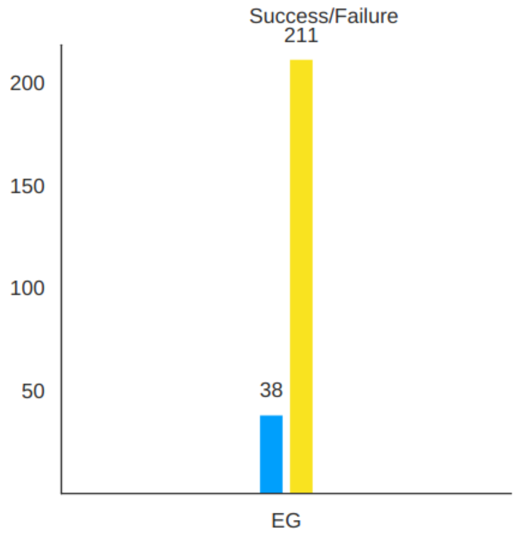


Figure 6: Success and failure rates for vanilla Tor measurements performed by OONI Probe in Egypt between June and November 2017.

since the tests weren't able to make connections to the Tor network within 300 seconds. More than 460 measurements showed that connections to the Tor network consistently failed, strongly suggesting that access to it was blocked.

Tor bridges appear to be blocked by some ISPs in Egypt as well. Vodafone, in particular, appears to be blocking the default *obfs4* bridges that are shipped as part of Tor Browser. (Anecdotally, we hear reports that private obfs4 bridges still work, but we don't have OONI test results to confirm this conclusion.) All measurements collected from state-owned Telecom Egypt, on the other hand, show that the built-in *obfs4* bridges work.

This is not the first time that we have observed Tor censorship in Egypt. In 2016, when Egyptian ISPs started to block media sites, we found that the Tor connection process was being disrupted via the blocking of requests to directory authorities.

Figure 6 illustrates the total number of tests that successfully connected to the Tor network versus those that failed between June 2017 and November 2017 from seven distinct autonomous system numbers (ASNs) in Egypt. 211 tests failed and only 38 test succeeded in connecting to the Tor network.

Recent OONI measurements suggest that access to the Tor network, when not using a private obfs4 bridge, remains blocked in Egypt.

5.2.1.2 Ethiopia During a wave of protests in 2016 [104], numerous Ethiopian media outlets and political opposition websites were blocked. WhatsApp was censored as well, along with several circumvention tool sites, including the Tor website.

In collaboration with Amnesty International, we published a research report [109] documenting these censorship events based on the analysis of OONI network measurements collected from Ethiopia. While we found that access to our site (torproject.org) was blocked, we did not find the Tor software itself being blocked in Ethiopia during the testing period. To enable Tor

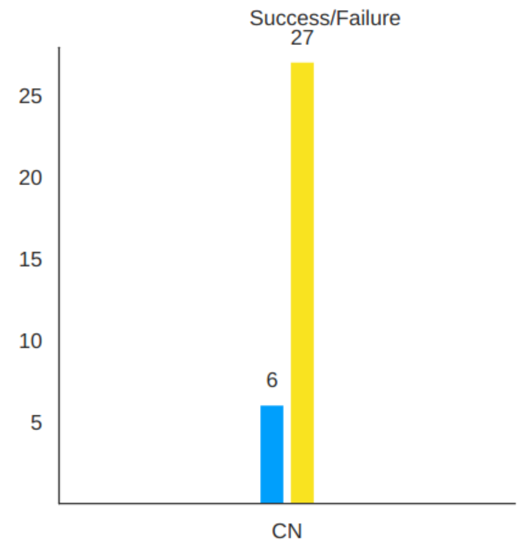


Figure 7: Success and failure rates for vanilla Tor measurements performed by OONI Probe in China between June and November 2017.

usage (despite our site being blocked), we shared information on how to download Tor Browser from an alternative site [29].

Over the last year, however, access to the Tor network has been blocked as well. Most OONI measurements collected from Ethiopia [62] in 2017 show that attempts to establish connections to the Tor network were unsuccessful, strongly suggesting that access is blocked.

Under a new Prime Minister [52] who is from the Oromia region—which has been protesting [104] against marginalization and persecution by authorities over the last few years—many reforms have taken place in Ethiopia over the past several months. These include the unblocking of hundreds of websites, which OONI reported[84] on in June.

Access to the Tor Project website, however, remains blocked. Similarly, the most recent measurements suggest that access to the Tor network remains blocked as well. Bridge reachability tests have never been run in Ethiopia, and so it remains unclear if access to Tor bridges is blocked.

5.2.1.3 China

China is known for its pervasive internet censorship apparatus, which also involves the blocking of numerous circumvention tools. Most OONI measurements [60] collected from local vantage points strongly suggest that access to the Tor network is blocked, since most tests fail to establish connections. Between June 2017 to November 2017, most measurements were unsuccessful, as seen in Figure 7.

In recent months, Tor reachability has been tested across multiple networks in China. Table 2 summarizes the findings from recent measurements by autonomous system.

Access to the Tor network appears to be blocked in most networks, since almost all measurements from those ASNs consistently fail to establish connections. It's worth noting, though, that the Tor network appears to be accessible from some Chinese networks, such as China Telecom (AS 4812) and the China Education & Research Network Center (AS 4538). Bridge reachability

Autonomous System	Vanilla Tor measurement result
Shanghai Mobile Communications (AS 24400)	Failure
China Telecom (AS 4812)	Success
Guangdong Mobile Communications (AS 9808)	Failure
China Education & Research Network Center (AS 4538)	Success
China Unicom (AS 4808)	Failure
China Telecom Backbone (AS 4134)	Failure
China Mobile Communications (AS 56040)	Failure

Table 2: Recent OONI Probe vanilla Tor measurement results by autonomous system for Chinese networks

tests haven't been run in China in recent years, limiting our ability to evaluate whether they work locally or not.

5.2.1.4 Iran The breadth and depth of internet censorship in Iran is pervasive. Thousands of OONI Probe measurements [63] collected from 60 local networks in Iran over the last several years confirm the blocking [108] of more than 800 domains. These include media outlets, opposition sites, pro-democracy sites, human rights sites, the blogs of Iranian political activists, as well as numerous circumvention tool sites, including torproject.org.

We also found the Tor network to be blocked in many networks in Iran, but accessible in some. While the domain of Tor's bridge database site was blocked (being a sub-domain of torproject.org), we found that some Tor bridges work in some networks in Iran.

Recent OONI measurements suggest that Tor remains blocked in some networks, like IranCell (AS 44244), but is accessible in other networks, like Aria Shatel (AS 31549). In general, internet censorship in Iran does not appear to be deterministic, as we've observed the dynamic blocking and unblocking of services over time. Censorship measurement is therefore required on an ongoing basis.

5.2.1.5 Turkey Direct connections to the Tor network appear to be blocked in Turkey as well. Between June 2017 and November 2017, most tests attempting to establish connections to the Tor network failed, as can be seen in Figure 8, based on measurements [64] collected from six distinct ASNs in Turkey.

Recent measurements collected from TurkNet (AS 12735) show that the Tor network is accessible, while measurements collected from Tellcom Iletisim (AS 34984) indicate that it's blocked, since most tests failed to establish connections to the Tor network. This ISP also appears to be blocking access to Tor's website, but the site is accessible from other networks, such as Vodafone (AS 15897). Bridge reachability tests have never been run by volunteers in Turkey, limiting our ability to assess whether Tor bridges work locally.

5.2.1.6 Venezuela A few months ago, one ISP in Venezuela blocked access to the Tor network as well. Following months of increased censorship, particularly targeting a number of local media websites and currency exchange sites, state-owned CANTV (AS 8048) blocked access to the Tor network, possibly to prevent people from getting around censorship [8].

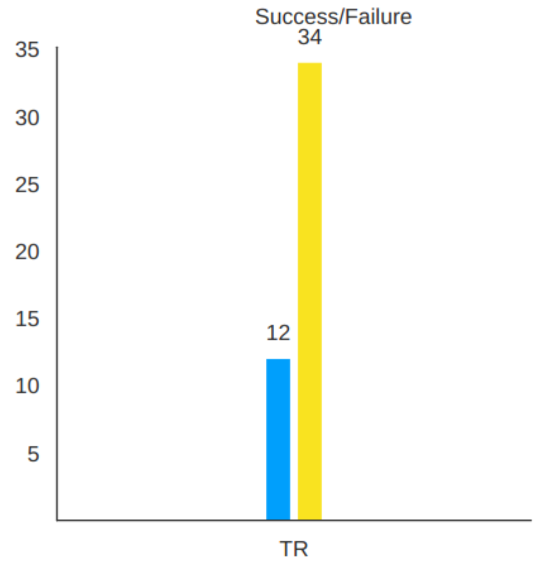


Figure 8: Success and failure rates for vanilla Tor measurements performed by OONI Probe in Turkey between June and November 2017.

All OONI measurements [65] collected from Venezuela up until 6th June 2018 were successful, showing that the Tor network was accessible then. On 20th June 2018, however, Tor testing on CANTV started to fail. Most other measurements collected from 20th June 2018 onwards (collected from the same network on an almost daily basis) failed as well, strongly suggesting that CANTV blocked access to the Tor network. According to OONI's scans in mid-August 2018 from CANTV, connections to 74% of well-known IP address and port combinations of the Tor network were blocked on the reverse path.

A large number of obfs4 bridges were blocked as well. Bridge reachability tests run from CANTV in late June 2018 show a failure rate of around 94% to known Tor bridges. Not all of these failures are necessarily caused by blocking, as some bridges might be offline or unreachable. The high percentage of connection failures, though, strongly suggests that well-known bridges are being blocked. Repeated testing in August 2018 also presented a high percentage: 88% of running bridges were unreachable from a CANTV vantage point. It's worth noting, though, that private bridges seemed to work. All testing to private Tor bridges resulted in successful connections, regardless of the type of bridge. This includes obfs4 bridges and bridges that don't use pluggable transports—so we conclude that the blocking was based on IP address, and not some more sophisticated DPI-based approach.

Over the last few months, CANTV has unblocked access to the Tor network. Further testing on 2nd October 2018 revealed that around 97% of public Tor nodes were reachable from the vantage point of CANTV. While the precise date of unblocking is unclear, Tor Metrics data seen in Figure 9 suggest that Tor may have been unblocked on 30th August 2018, since we observe a spike in Tor usage on that day.

The Tor Project website has remained accessible in CANTV (and other networks) all along,

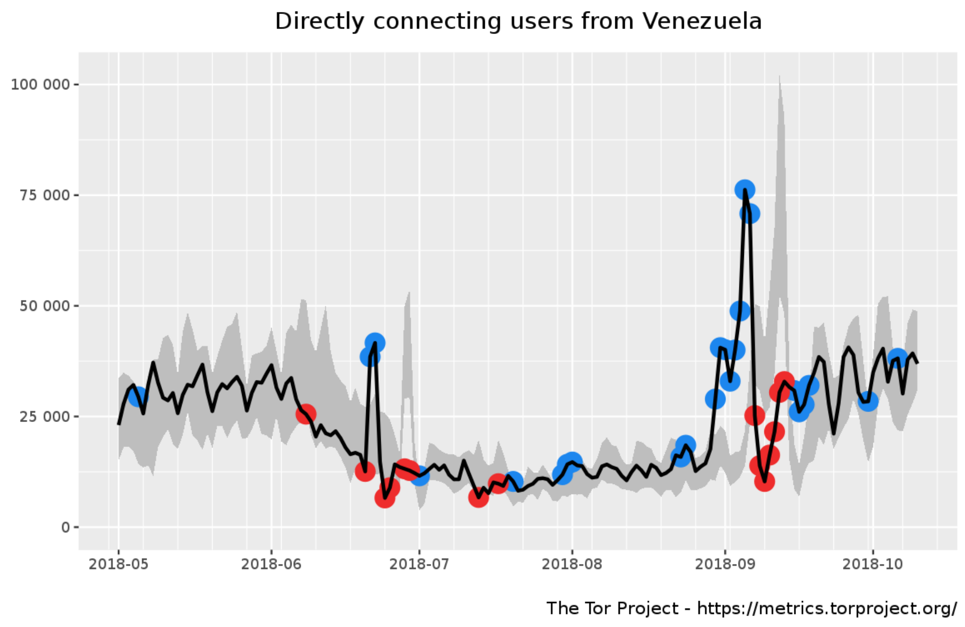


Figure 9: Number of directly connecting Tor users from Venezuela between 1st May and 10th October 2018. The darker grey range indicates the expected user counts. Blue and red spots indicate data points that fell either above or below the expected range.

even during the time that direct Tor network connections and default obfs4 bridges were blocked.

5.2.2 Tor Metrics anomaly detection

Tor Metrics has an “early warning system" [12] to indicate anomalies in the counts of directly connecting Tor users. The detector is based on a simple model of the number of users per day, per country. That model is used to assess whether the number of users we observe is typical, too high, or too low. In a nutshell, the prediction on any day is based on activity of previous days, locally as well as worldwide.

This system is integrated into the Tor Metrics website and can annotate visualizations with any abnormalities that are detected. When a country puts new censorship in place this can appear in our user data as an increase in directly connecting users in that country. As an example, Uganda introduced a “social media tax" [100] on July 1st 2018. This can be seen in Figure 10 for that date as an unexpected rise in users.

Similarly, when a country introduces censorship of the Tor network this can appear in our user data as a drop in directly connecting users. Many networks in Venezuela blocked access to the Tor network between June and August 2018 [102] and the effects of this can be seen in Figure 9.

Another system [107] developed and maintained by academic researchers, detects ongoing per-country anomalies in the user counts. It identifies contiguous anomalous periods, rather than daily spikes or drops, and allows anomalies to be ranked according to deviation from

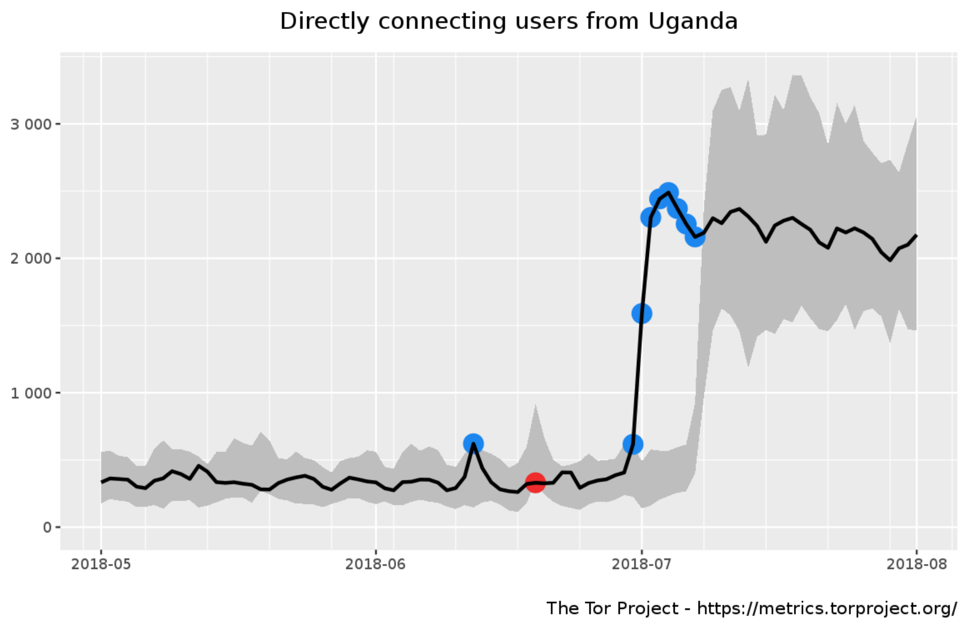


Figure 10: Number of directly connecting Tor users from Uganda between 1st May and 1st August 2018. The darker grey range indicates the expected user counts. Blue and red spots indicate data points that fell either above or below the expected range.

expected behavior. This system is currently run continuously and reports daily to a mailing list [37] but is not yet integrated with Tor Metrics.

5.2.3 Events timeline

Tor Metrics maintains a timeline [92] of events that might affect user metrics, to aid in the interpretation of data and visualizations. This timeline is—by necessity—manually maintained. Where possible each entry in the timeline is accompanied by links to relevant visualizations, discussions, and news articles. We may create entries in response to reports from users or news outlets, automated anomaly detection, analysis of data collected by OONI and many other sources.

Some example entries are shown in Table 3. When viewing visualizations on the Tor Metrics website, relevant entries will be shown under the graph for some visualizations.

5.2.4 Future directions

There is significant work to be done to get us from knowing when access to the Tor network is blocked, to knowing which pluggable transports work in which censored countries, to using our censorship measurement data to inform development of new pluggable transports. We don't currently have data about whether bridges are blocked in many of the countries we've studied, let alone which pluggable transports might be blocked. We need to develop more sophisticated tests than OONI Probe's vanilla Tor and bridge reachability tests to gather this data.

Date	Entry	Links
2018-07-01 to present	A social media tax takes effect in Uganda. The government pressures ISPs to block VPNs.	User graphs, BBC article, AllAfrica article
2018-07-28 to present	Another jump of relay users in Turkey, from 5k to about 30k.	User graphs
2018-07-29 to 2018-08-08	Protests over road safety in Dhaka, Bangladesh. Reports of mobile network throttling and blocking of Facebook.	New York Times article, Daily Star article

Table 3: Selected entries from the Tor Metrics Timeline that can provide context when interpreting the user count data and visualizations.

The type of testing we need to do may not fit into the OONI model—OONI aims to deploy a simple, easy to use tool to its volunteers and some of the Tor pluggable transports are relatively large and unwieldy. We might need to explore other ways of deploying our tests and collecting data. For example, we might release a modified version of Tor Browser that runs through the included pluggable transports, testing to see if connections succeed, and then uploading the results.

OONI might deploy new tests that not only establish a connection to the Tor network, but also perform some activity, to ensure that the test traffic is representative of that of a real user. A connection may be allowed to succeed but then be disconnected shortly after, leaving Tor unusable from a user perspective with no evidence of this being collected by OONI

Accurate and widespread measurement is essential for us to be able to respond to censorship. The data that Tor Metrics and OONI collect gives us some visibility into both ongoing censorship and discrete censorship events, but we need to do much more.

6 Onion Routing Network Defense

6.1 Tor network health

The health of the Tor network is crucial to all of the services Tor provides, including the ability of users to bypass censorship. This section details the health of the Tor network in terms of capacity, diversity, and performance. Section 6.1.1 describes the current state of the network, including some statistics and basic information on network health.

Section 6.1.2 describes our ongoing fight against malicious relays and includes our detection methods and our process to remove a relay from the network.

6.1.1 The state of the network

6.1.1.1 Numbers The Tor network is currently composed of about 6400 relays run by volunteers around the world. As Figure 11 shows, the number of relays has been quite steady during 2018.

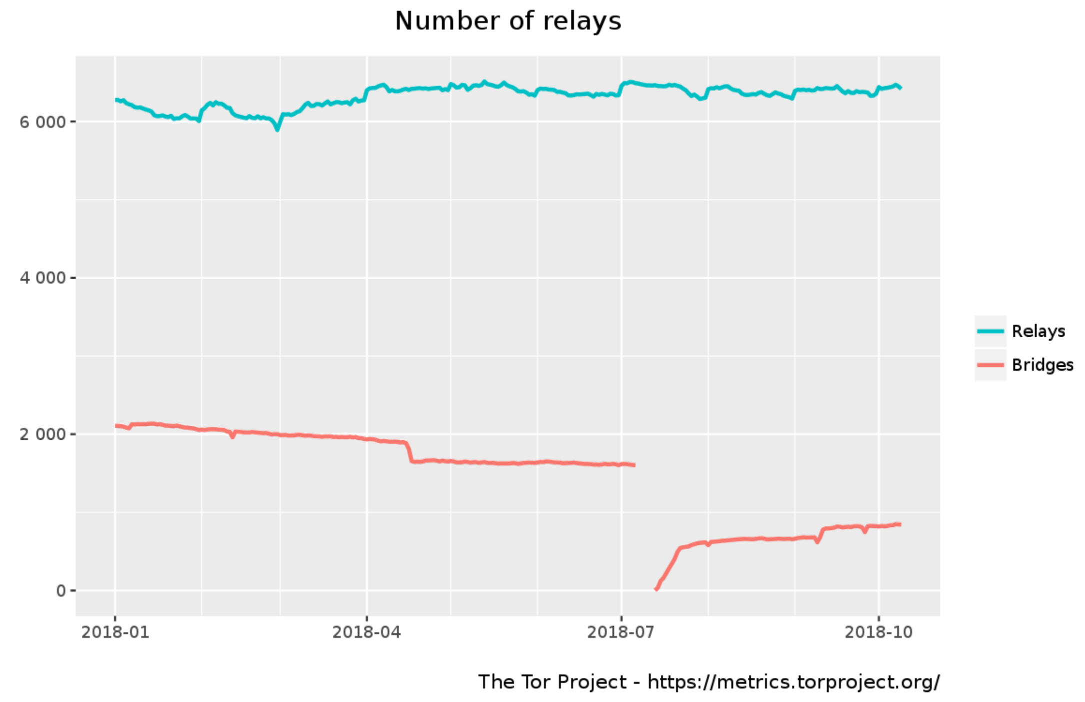


Figure 11: Number of running relays and bridges in the public Tor network. For relays, only those that have been verified as reachable by the directory authorities are included. For bridges, only those that announce themselves to the bridge authority are included (that is, private bridges are not included).

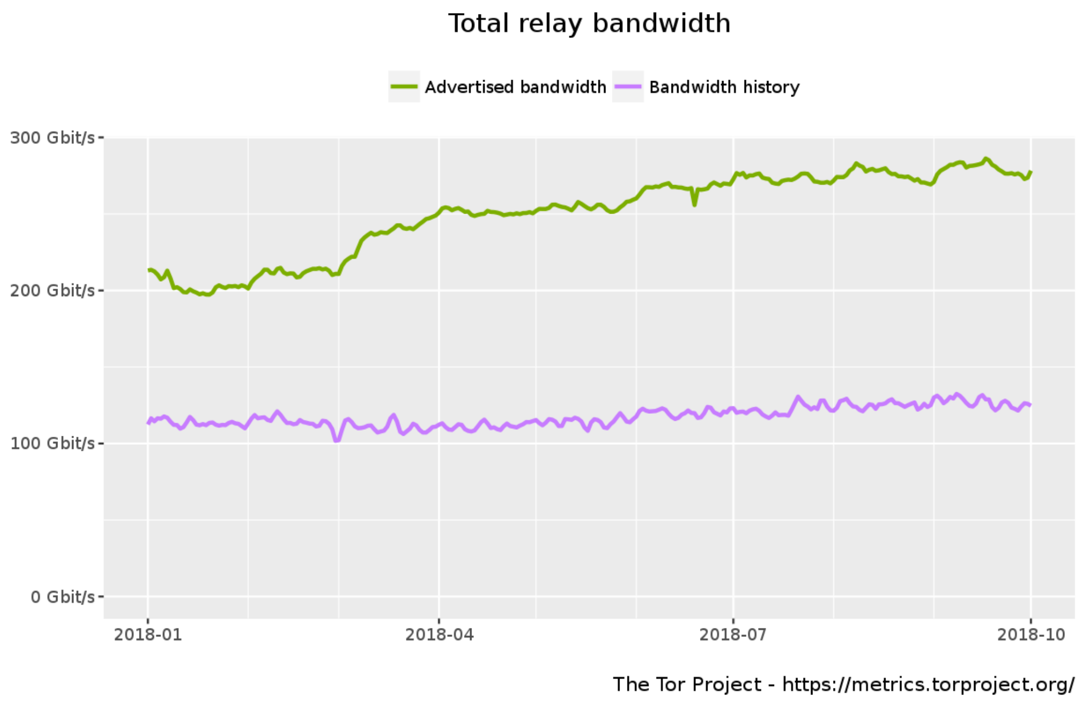


Figure 12: The sum of advertised and consumed bandwidth across all relays in the network. The advertised bandwidth is the peak volume of traffic, both incoming and outgoing, that a relay has observed itself able to do based on recent data transfers. The consumed bandwidth is the volume of traffic, both incoming and outgoing, that the relay has reported it has transferred.

This graph includes bridges, which are relays that are not listed in the public directory. In early July we transitioned to a new bridge authority, which is a special relay that aggregates bridge information. Currently there are slightly under 1000 bridges in the network.

The capacity of the Tor network is hovering around 275 GBit/s and around 125 GBit/s are actually in use. Figure 12 shows an increase in available bandwidth since March 2018.

As shown in Figure 13, by our measurements the network is currently seeing an estimated 2 million users each day. Since March, this number has been quite steady. It is important to understand that this count makes the conservative assumption that each of these users is online for the entire day. If every user is instead online only a few hours each day, our estimate could be low by a factor of ten. Early reports from new research [49] show a much higher estimate and we plan to take a close look at their methodology.

Notice the surge in users at the beginning of the year. This was due to a denial of service attack that we discuss in Section 6.2. We can ignore this for now.

Finally, Figure 14 shows the number of users using a bridge to enter the network. The current estimate shows around 55,000 daily users. Again, disregard the surge of users during the middle of the year; this was also the result of a denial of service attack.

6.1.1.2 Diversity Diversity is an important factor for the safety and anonymity provided by the Tor network. We can measure diversity in a variety of ways, but for this discussion, we

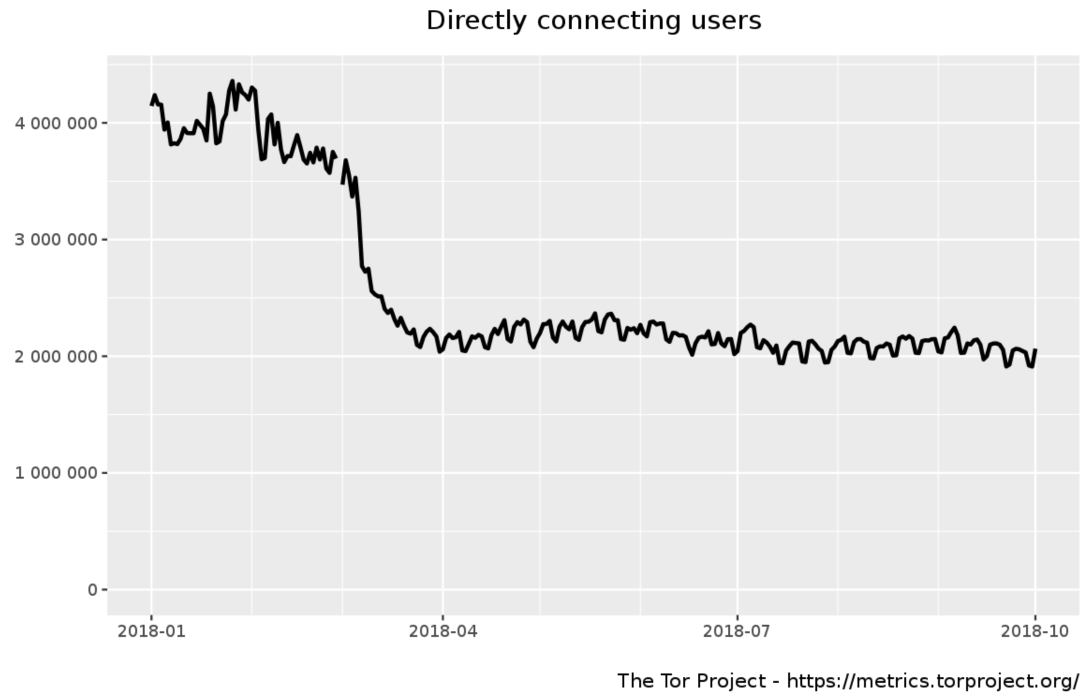


Figure 13: Estimated number of daily directly-connecting clients; this excludes clients connecting via bridges. These estimates come from counting the number of directory requests reported by relays.

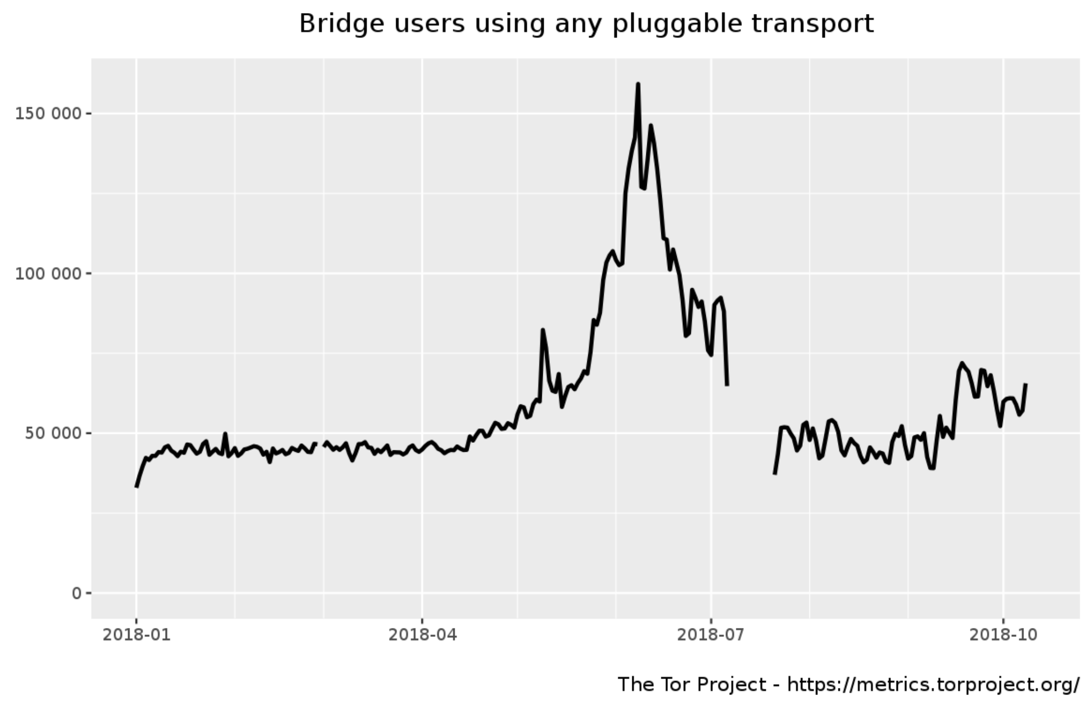


Figure 14: Estimated number of daily clients connecting via bridges. These numbers come from the number of directory requests reported by bridges.

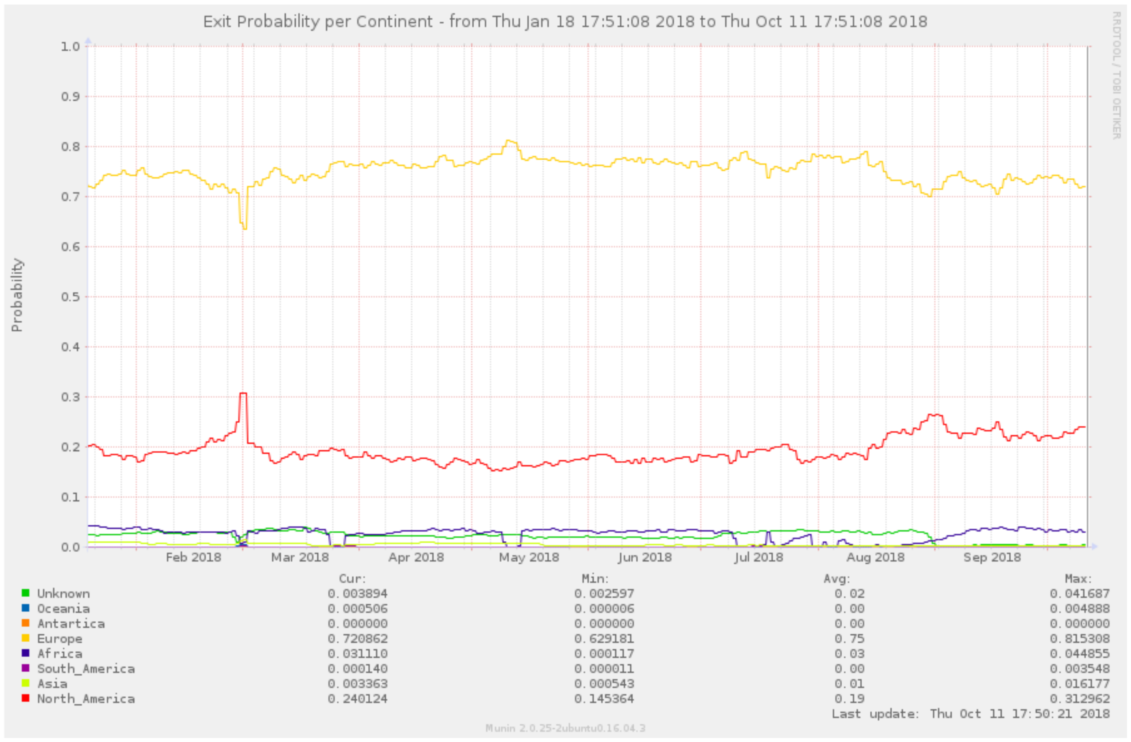
focus on diversity of the operating systems (OSes) that relays run on and the diversity of the physical locations of relays.

The following simple example shows the importance of operating system diversity: a software vulnerability that affects all Windows computers won't threaten the security of relays that run on Linux. So in terms of security, having relays that run on a wide variety of OSes improves the resilience and safety of the Tor network.

Unfortunately, the vast majority of relays (over 90%) are currently running on the Linux OS. The rest are on variants of the BSD OS; very few are on Windows. This is far from ideal for the network, and there is work to do to make sure Tor is easy to install and configure on many platforms. One of our current projects includes outreach and improved documentation to make it easier to set up relays on BSD OSes.

Geographic diversity is important because it improves the anonymity the network can provide, as well as its ability to help bypass censorship. If someone watching network traffic, such as a government actor, can observe both the connection to the guard relay and traffic exiting the Tor network, they may be able to compromise the anonymity of the user. If the guard relay and the exit relay are in different jurisdictions, there is much less chance of this.

Also, it is easier to censor a connection if, for example, an ISP can observe a user connecting to the Tor network as well as the exit relay and determine the user's web destination. If the exit relay is in a different jurisdiction than the user, there is less chance that this type of censorship can succeed. In general, spreading relays across different jurisdictions lessens the chance that a single observer can see the entire path through the Tor network.



	Cur:	Min:	Avg:	Max:
🟩 Unknown	0.003894	0.002597	0.02	0.041687
🟦 Oceania	0.000506	0.000006	0.00	0.004888
🟧 Antartica	0.000000	0.000000	0.00	0.000000
🟨 Europe	0.720862	0.629181	0.75	0.815308
🟦 Africa	0.031110	0.000117	0.03	0.044855
🟪 South_America	0.000140	0.000011	0.00	0.003548
🟩 Asia	0.003363	0.000543	0.01	0.016177
🟥 North_America	0.240124	0.145364	0.19	0.312962

Last update: Thu Oct 11 17:50:21 2018

Figure 15: Exit probability by continent based on resolving relay addresses using an IP geolocation database.

Figure 15 shows the probability that a Tor connection will exit the network in each continent. As you can see, there is a high concentration of exit relays on the European continent.

We use IP geolocation databases to determine where the exit relays are located, and these are often not completely accurate, but we see that users will almost always exit the network either in North America or Europe. This is far from ideal and something we are working to improve. We've recently hired a relay advocate who helps support current relay operators and does outreach to attract new ones. In addition, as part of our global south initiative, our community liaison has been helping volunteers set up relays in those regions.

6.1.2 Malicious relays

This section describes the types of malicious relays we see in the network and how we deal with them.

6.1.2.1 Rejecting relays Anyone on the Internet can run a relay which means that we need to be aware that malicious actors can join anonymously and disrupt the network—sometimes in ways we haven't imagined.

The nine Tor directory authorities come to a consensus every hour about which relays in the network can be used and publish this list. If a majority of directory authorities agree that a relay is behaving maliciously, it is removed from the list.

The Tor community created a *bad relay* team that is in charge of detecting malicious relays and proposing reject rules to the directory authorities. The directory authorities, in turn, decide whether to apply the rules so they really have the final word.

That team has a series of tools to probe the network in order to detect malicious behavior by relays. Regularly, all of the 6400+ relays are probed and tested for various things. Here are some of the current tests:

- Stripping SSL certificate
- Re-writing/Redirecting plaintext Web page
- TLS Downgrade
- Bitcoin address rewrite
- Exit re-injecting traffic into the network
- Malicious onion service directory

There is, of course, much more that we could be doing, and we are always looking for volunteers to help with this. In an ideal world, the Tor Project would have funding to hire a *network health engineer* to monitor the network more closely and improve our detection methods.

6.1.2.2 Malicious onion service directories A relay that has been online for 96 hours or more can become an *onion service directory*. Because of a design flaw in version 2 of the onion service protocol, such a relay can learn addresses of onion services in the network. A long-lived relay can learn many onion addresses over time.

This could enable the relay to launch a series of attacks in order to de-anonymize onion service users. We put great effort into designing version 3 of the protocol, which fixes this problem. But we still, and will for years to come, have version 2 services running that are affected by this issue.

For this reason, we have a scanner that detects onion service directories that collect .onion addresses and visit them. Every week we reject 5 to 10 relays from the network that are doing this.

6.1.3 Current outlook and future directions

The Tor network is healthy in terms of capacity. We have a healthy number of relays that are spread out over many countries around the world, which is crucial for protecting anonymity and evading censorship.

The number of Tor users is also important to the health of the system. One can't be anonymous alone—good anonymity requires a crowd. As the number of Tor users continues to grow, the anonymity provided by the network becomes stronger. By working towards making Tor more accessible, more user friendly, and more widely deployed on different devices (for example mobile), safety is improved.

The Tor Project and its community have built tools and processes to monitor, detect, and reject malicious relays, making the network safer. Onion service version 3 is a great example of years of cooperation between the development team, researchers, and our community to improve safety.

There is, of course, a lot more that we can do. Getting more exit relays running in more places around the world is an important goal, along with adding more OS diversity to the network. Improving our network monitoring is an ongoing goal. The health of the network really depends on collaboration among everyone in our community.

6.2 Denial of service

This section describes the denial of service attacks that the Tor network faces. We'll use a recent example to demonstrate how important it is that our community and monitoring work together.

6.2.1 Notable attacks

In 2015, Nick Mathewson, Tor maintainer, released a technical report [51] detailing different denial of service attacks on the Tor network along with possible defenses that could be implemented. See section 4.3 for highlights from this report.

Over the years, the Tor Project and community have dealt with many attacks of this kind, and they usually come as a surprise. This section highlights some of the attacks we have faced and the lessons they have taught us.

6.2.1.1 Sefnit botnet On the 20th of August 2013, the number of users connected to the network rose to abnormal levels [13] (see Figure 16). Initial assessments of some relays showed a large number of clients trying to connect to onion services.

Although the network was still working fine, relays experienced higher than normal use of their processors. This was traced to TAP [15], the protocol for establishing communication that the Tor client used. NTor [14], a new, more secure, and better performing protocol, was about to be released in a new version of the client software.

Tor developers changed the relay software so that it prioritized the NTor protocol. Old clients using TAP had less priority and were phased out.

Fortunately, by the end of 2013, the rise in users was identified by a Microsoft researcher as part of the Sefnit botnet. It was removed by Microsoft anti-virus at that point.

6.2.1.2 The sniper attack In February 2014, a paper was presented at the Network and Distributed Systems Security symposium titled **The Sniper Attack: Anonymously Deanonymizing and Disabling the Tor Network** [42].

It describes a way to remotely make a Tor relay ineffective using very few resources. One of the most important aspects of this attack is that it can be used very efficiently to deanonymize users, especially onion service users.

This is a concrete example of how a very cheap DoS could be used to mount other types of attacks. The academic literature is years ahead of practice in terms of attacks, so this is currently theoretical. Until we see it in practice, we don't know what its actual effects might be.

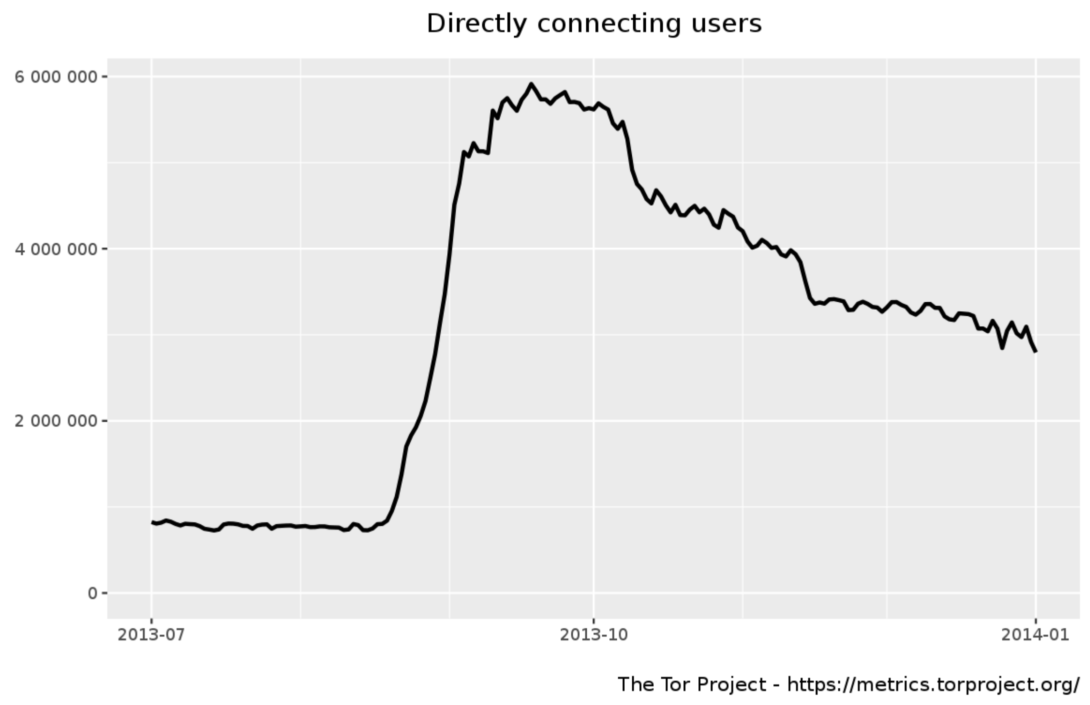


Figure 16: Estimated number of users between August 2013 and January 2014.

The sniper attack showed us that taking down a relay could lead to deanonymizing specific onion services.

6.2.1.3 DoS attack using onion services In mid December 2017, relay operators started to report to the tor-relays mailing list that they were seeing spikes in traffic to their relays and unusual amounts of memory and processing power being used.

This can sometimes happen in normal circumstances. For example if your relay becomes the guard relay for the Facebook onion service, you'll see a surge in resource usage in order to handle all of the users trying to reach it.

However, by the end of the month, the number of reports was unusually high, which led us to investigate further. We also noticed that, at the same time, there was a huge rise in the number of Tor users (see Figure 17).

After days of investigation, we realized that the entire network was being bombarded by onion service requests. The sheer number was causing relays to crash, to be shutdown by their operators to conserve resources, or at minimum, to be rendered very slow or unusable.

We analyzed connection patterns to some of the affected relays and noticed that there were 3 types of onion service connections that were coming to the relays in large numbers:

1. Client rendezvous requests
2. Service rendezvous requests
3. Tor2web rendezvous requests

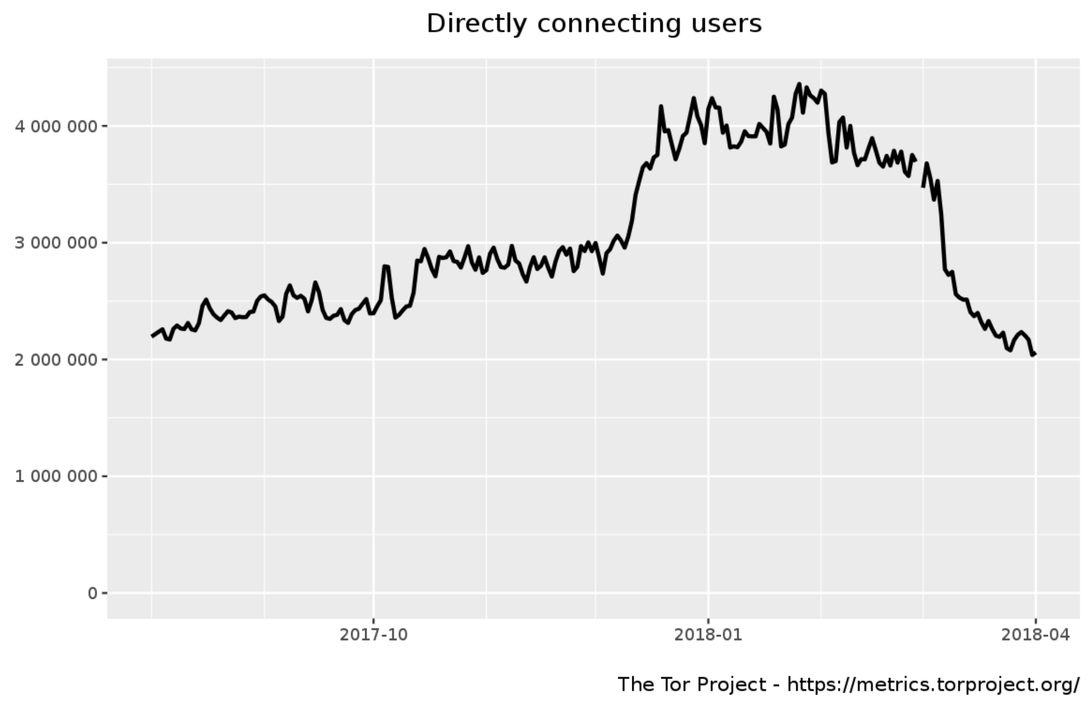


Figure 17: Estimated number of daily users between August 2017 and March 2018.

The attacker created hundreds of thousands of client onion service requests (1) to specific onion services, which created a large number of rendezvous circuits (2), making the problem significantly worse.

We believe the Tor2web requests (3) were unrelated, although they started around the same time.

Tor2Web is a gateway that allows users to access onion services from a regular web browser. This speeds up connections for the user, but does not provide anonymity. The so called “DarkWeb scanners" attempt to scan all onion services. They use Tor2web to increase their efficiency and thus scan the onion space faster. This causes excessive memory and processing power to be used on the relays that connect with the onion services.

In the face of this denial of service attack (DoS) we quickly rolled out 3 defenses which we implemented in all stable versions of Tor.

- Limit concurrent connections
- Limit the rate at which circuits can be created
- Block Tor2web connections

We observed that all of the malicious requests were coming from roughly 500-600 different IP addresses located within the networks of well known hosting providers. This is probably because the cost of mounting a large DoS attack from many IP addresses is much higher than only using relatively few addresses.

Because thousands of requests were coming from the same IP address, limiting the number of concurrent connections to guard relays from the same address was an effective defense. In addition, after the limit on concurrent connections was reached, we limited the rate at which guard relays could contact other relays to form a circuit through the Tor network. We also blocked connections from Tor2web.

Once the majority of relays were running the updated software, the performance of the network recovered quickly. After a few months, the attack stopped. To this day, these defenses are still in place, protecting the Tor network from such abuse.

6.2.1.4 Key takeaways These episodes took us by surprise and forced some Tor developers to drop everything to investigate and work on defenses. But they produced some key takeaways.

- Large numbers of circuit requests create pressure not only on the network, but also on the Tor software. DoS attacks often highlight weaknesses in the application that can cause it to become unusable or to crash. Identifying these weaknesses prior to an attack is crucial to making Tor more resilient.
- The onion service design is clearly susceptible to DoS attacks. We have begun thinking about how we can solve these problems, but this is ongoing work.
- Listening to the Tor community is critical. Because Tor developers engage with the relay operators, we were able to notice that an attack was happening more quickly than if we had relied only on our own measurements. Working with our community has been instrumental in defending against most of the attacks we've experienced over the years.
- DoS attacks always have unpredictable effects, not only on the network but on the Tor software itself. These are very dangerous attacks that can quickly spiral out of control and we take them extremely seriously.

6.2.2 The future

There are several important points to keep in mind when we consider how to defend against future DoS attacks on the Tor network.

First, encouraging a wide and healthy research community is crucial. Researchers often notice possible attacks before they are actually seen in practice. Reacting to what researchers discover is important to keeping the network strong.

Second, we need to interact constantly with our user and relay operator communities. They have their fingers on the pulse of Tor network health and are our most efficient "monitoring device". When they report issues, it is because they care about the Tor network. It is our job to listen to and work with them to improve the safety and performance of the network.

Third, we need to improve our ability to understand what is going on inside the Tor software. When we are faced with a potential DoS attack, extracting relevant information quickly is critical. In order to do this we need proper instrumentation in the Tor software. We don't currently have this, but in the future, we need it.

Finally, we need the resources to defend against attacks. When these attacks occur, developers need to drop everything to investigate and work on fixes. The Tor Project needs to get to the point where it is free to work on what is important for the Tor network and our users. We need to be able to allocate resources towards defenses without jeopardizing our funding. Without the ability for developers to take the time necessary to defend against these attacks, the next one could be the one that takes down the network.

7 A Stronger Research Community

7.1 Tor's role in the research community

Just about every major computer security conference these days has a paper analyzing, attacking, or improving Tor. While fifteen years ago the field of anonymous communications was mostly theoretical, with researchers speculating that a given design should or shouldn't work, Tor now provides an actual deployed testbed. Tor has become the gold standard for anonymous communications research for three main reasons:

First, Tor's source code and specifications are open. Beyond its original design document [16], Tor provides a clear and published set of technical specifications [75] describing exactly how it is built, why we made each design decision, and what security properties it aims to offer. The Tor developers conduct design discussion in the open, on public development mailing lists, and the public development proposal process [50] provides a clear path by which other researchers can participate.

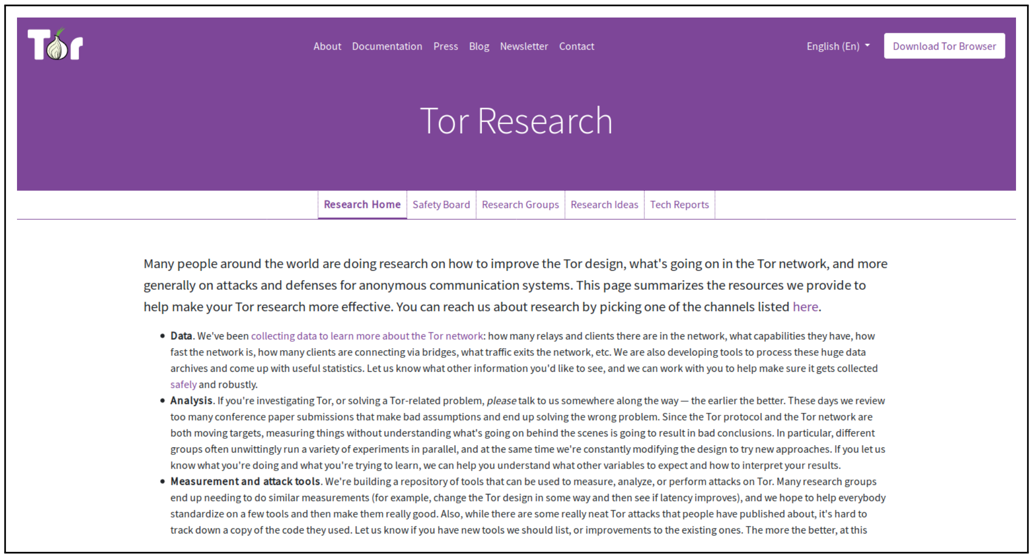
Second, Tor provides open APIs and maintains a set of tools to help researchers and developers interact with the Tor software. The Tor software's "control port" [74] lets controller programs view and change configuration and status information, as well as influence path selection. We provide easy instructions for setting up separate private Tor networks for testing [40]. This modularity makes Tor more accessible to researchers because they can run their own experiments using Tor without needing to modify the Tor program itself.

Third, real users rely on Tor. Every day millions of people connect to the Tor network and depend on it for a broad variety of security goals. In addition to its emphasis on research and design, the Tor Project has developed a reputation as a non-profit that fosters this community and puts its users first. This real-world relevance motivates researchers to help make sure Tor provides provably good security properties.

Tor attracts researchers precisely because it brings in so many problems that are hard to solve yet matter deeply to the world. How to protect communications metadata is one of the key open research questions of the century, and nobody has all the answers. Our best chance at solving it is for researchers and developers all around the world to team up and work in the open to build on each other's progress.

Since starting Tor, we've done over 100 Tor talks to university research groups all around the world, teaching grad students about these open research problems in the areas of censorship circumvention (which led to the explosion of pluggable transport ideas), privacy-preserving measurement, traffic analysis resistance, scalability and performance, and more.

The result of that effort, and of Tor's success in general, is a flood of research papers, plus a dozen research labs who regularly have students who write their theses on Tor. The original Tor design paper from 2004 now has over 4000 citations, and in 2014 Usenix picked that paper out of all the security papers published in 2004 to win their Test of Time award.



Tor Research

Research Home | Safety Board | Research Groups | Research Ideas | Tech Reports

Many people around the world are doing research on how to improve the Tor design, what's going on in the Tor network, and more generally on attacks and defenses for anonymous communication systems. This page summarizes the resources we provide to help make your Tor research more effective. You can reach us about research by picking one of the channels listed here.

- **Data**. We've been collecting data to learn more about the Tor network: how many relays and clients there are in the network, what capabilities they have, how fast the network is, how many clients are connecting via bridges, what traffic exits the network, etc. We are also developing tools to process these huge data archives and come up with useful statistics. Let us know what other information you'd like to see, and we can work with you to help make sure it gets collected safely and robustly.
- **Analysis**. If you're investigating Tor, or solving a Tor-related problem, *please* talk to us somewhere along the way — the earlier the better. These days we review too many conference paper submissions that make bad assumptions and end up solving the wrong problem. Since the Tor protocol and the Tor network are both moving targets, measuring things without understanding what's going on behind the scenes is going to result in bad conclusions. In particular, different groups often unwittingly run a variety of experiments in parallel, and at the same time we're constantly modifying the design to try new approaches. If you let us know what you're doing and what you're trying to learn, we can help you understand what other variables to expect and how to interpret your results.
- **Measurement and attack tools**. We're building a repository of tools that can be used to measure, analyze, or perform attacks on Tor. Many research groups end up needing to do similar measurements (for example, change the Tor design in some way and then see if latency improves), and we hope to help everybody standardize on a few tools and then make them really good. Also, while there are some really neat Tor attacks that people have published about, it's hard to track down a copy of the code they used. Let us know if you have new tools we should list, or improvements to the existing ones. The more the better, at this

Figure 18: Mock-up of redesigned Tor Research Portal.

7.1.1 Tor research resources

Resources for researchers are collected within the Tor Research Portal [94] which is currently undergoing a redesign. This portal serves as a landing point for anyone interested in doing Tor-related research. A mock-up of the redesigned portal can be seen in Figure 18.

7.1.1.1 Datasets We've been collecting data to learn more about the Tor network: how many relays and clients there are in the network, what capabilities they have, how fast the network is, how many clients are connecting via bridges, what traffic exits the network, etc. We are also developing tools to process these huge data archives and come up with useful statistics. These datasets are described in more detail in Section 5.1.

7.1.1.2 Measurement and attack tools We're building a collection of tools that can be used to measure, analyze, or perform attacks on Tor. Many research groups need to do similar measurements (for example, change the Tor design in some way and then see if communication delays are reduced), and we hope to help everybody standardize on a few tools and then make them really good. Also, while there are some really neat Tor attacks that people have published, it can be hard to track down the code they used in their experiments.

We need defenses too—not just attacks. Most researchers find it easy and fun to come up with new attacks on anonymity systems. We've seen this lately in terms of improved congestion attacks, attacks based on remotely measuring latency or throughput, and so on. Knowing how things can go wrong is important, and we recognize that the incentives in academia aren't aligned with spending energy on designing defenses, but it sure would be great to get more people working on how to address the attacks.

7.1.2 Tor research safety board

The research safety board is a group of researchers who study Tor, and who want to *minimize privacy risks while fostering a better understanding of the Tor network and its users*. We aim to accomplish this goal in three ways:

- Develop and maintain a set of guidelines that researchers can use to assess the safety of their Tor research.
- Give feedback to researchers who use our guidelines to assess the safety of their planned research.
- Teach program committees about our guidelines, and encourage reviewers to consider research safety when reviewing Tor papers.

The board has seen early success in three ways. First, it has handled a dozen or so cases, where research groups wrote their research plans and a safety self-assessment and submitted them to the board. The board engaged in a discussion with the researchers to help them flesh out their assessment and to help them think through their safety plans.

Second, there are a small but growing set of papers out there on *privacy-preserving measurement*, which expand the toolbox of safe ways to do network-wide measurements.

And third, major conferences like PETS and Usenix Security are starting to ask authors about their safety board interactions when the paper includes some questionable or concerning methodology, and mainstream papers are starting to include an "Ethics of Data Collection" section or the equivalent [54].

To go with these successes, we have three things we want to do in the near term to position the research safety board for further success. First, we need to publish the details of more past cases, so future researchers can benefit from reading them. One of the main goals of the safety board is to provide historical examples so prospective researchers can get concrete ideas of how to assess safety. Authors generally agree to publish their interactions, but only after their paper has gone public—which means we need better ways of tracking the status of old safety board cases and noticing when we can make them public.

Second, we need to improve our workflow for noticing and responding to new cases. In the past we used a shared mailing list, but some people found it hard to interact with. Now we use our own HotCRP instance, but there are some gaps in the workflow for noticing and assigning new submissions, and we sometimes miss them. The fundamental problem here is that we have only busy professors involved. Maybe we need to expand the set of volunteers to include people who can better manage submissions and coordinate review.

Finally, we need to do more outreach to program committees. We've heard anecdotes from several program committee members who excitedly told us that they asked authors "what the Tor safety board thought"—and that's a fine first question, but it's misunderstanding the intended role of the safety board. We don't want to be the gatekeepers of Tor research. Ultimately, we want reviewers to be demanding a discussion of safety in the paper, not a discussion of the authors' interaction with the safety board.

8 Conclusion

Today, free and open communication on the internet is essential for the growth and maintenance of strong communities and societies. Threats to this open communication come from a variety of state and non-state censors. The Tor network is a robust and highly scaled communications network that protects communications metadata, and as such has an important role to play in combatting these denials of service.

In this report we provide a snapshot of the current state of several research areas in the Tor ecosystem: pluggable transports that help users evade censorship of the Tor network; an intuitive user experience that enables real users to accomplish their goals; expanding the set of applications, especially on mobile, that can be used safely on the Tor network; Tor performance on networks with poor resources or that are under DoS attack; measurement of network resources and of censorship; defense of the Tor network against attack; and strengthening the ties between the Tor Project and the academic research community.

Over the next several months, we will continue our work to make progress in these areas; future reports will describe that progress. In the pluggable transport area, Snowflake presents a promising direction. Our work on improving the user interface for setting up bridges and pluggable transports, along with the Guardian Project's work on getting Snowflake working on Android should make this technology more accessible for users. OpenArchive represents a positive step toward enabling users to upload video from censored regimes. Ongoing work on improving Tor performance and on protecting the Tor network from attack aims to make the network more robust. Improvements to network measurement, as well as adding more realistic tests to OONI Probe will help us detect which locations are censored and which pluggable transports work from censored locations. Finally, strengthening our relationship with the academic research community will ensure that we can continue to advance in these areas as well as many others.

9 Acknowledgements

This research was supported by NSF grants CNS-1526306 and CNS-1619454, and by DARPA contract HR0011-16-C-0056. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of NSF or DARPA.

References

[1] Adversary Lab. https://github.com/OperatorFoundation/AdversaryLab.

[2] Mashael Alsabah, Kevin Bauer, Tariq Elahi, and Ian Goldberg. The path less travelled: Overcoming Tor's bottlenecks with traffic splitting. In *Proceedings of the 13th Privacy Enhancing Technologies Symposium (PETS 2013)*, July 2013.

[3] Mashael AlSabah, Kevin Bauer, Ian Goldberg, Dirk Grunwald, Damon McCoy, Stefan Savage, and Geoffrey Voelker. DefenestraTor: Throwing out windows in Tor. In *Proceedings of the 11th Privacy Enhancing Technologies Symposium (PETS 2011)*, July 2011.

[4] American Library Association. The Freedom to Read Statement. http://www.ala.org/advocacy/intfreedom/freedomreadstatement.

[5] Brave. https://brave.com/.

[6] Briar: Secure messaging, anywhere. https://briarproject.org/.

[7] Chatsecure: Free and open source encrypted chat for ios. https://chatsecure.org/.

[8] Mariengracia Chirinos, Andrés Azpúrua, Leonid Evdokimov, and Maria Xynou. The State of Internet Censorship in Venezuela. https://ooni.torproject.org/post/venezuela-internet-censorship/.

[9] Cliqz. https://cliqz.com/en/.

[10] CollecTor. https://metrics.torproject.org/collector.html.

[11] B. Constantine, G. Forget, R. Geib, and R. Schrage. Framework for TCP Throughput Testing. RFC 6349 (Informational), August 2011.

[12] George Danezis. An anomaly-based censorship-detection system for Tor. Technical Report 2011-09-001, The Tor Project, September 2011.

[13] Roger Dingledine. How to handle millions of new Tor clients. https://blog.torproject.org/how-handle-millions-new-tor-clients.

[14] Roger Dingledine and Nick Mathewson. Tor Protocol Specification: The "ntor" handshake. https://gitweb.torproject.org/torspec.git/tree/tor-spec.txt#n1090.

[15] Roger Dingledine and Nick Mathewson. Tor Protocol Specification: The TAP handshake. https://gitweb.torproject.org/torspec.git/tree/tor-spec.txt#n1043.

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

[17] Roger Dingledine and Steven J. Murdoch. Performance improvements on Tor or, why Tor is slow and what we're going to do about it. Technical Report 2009-11-001, The Tor Project, November 2009.

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

[19] Mohammad El-Taher, Hassan Al-Azhary, Sarah Mohsen, Leonid Evdokimov, and Maria Xynou. The State of Internet Censorship in Egypt - Full Report. https://ooni.torproject.org/documents/Egypt-Internet-Censorship-AFTE-OONI-2018-07.pdf.

[20] Nathan Evans, Roger Dingledine, and Christian Grothoff. A practical congestion attack on Tor using long paths. In *Proceedings of the 18th USENIX Security Symposium*, August 2009.

[21] Leonid Evdokimov, Maria Xynou, Mohammad El-Taher, Hassan Al-Azhary, and Sarah Mohsen. The State of Internet Censorship in Egypt - Summary. https://ooni.torproject.org/post/egypt-internet-censorship/.

[22] Ellis Fenske, Akshaya Mani, Aaron Johnson, and Micah Sherr. Distributed measurement with private set-union cardinality. In *Proceedings of the 24th ACM Conference on Computer and Communications Security (CCS '17)*. ACM, 2017.

[23] David Fifield. meek. https://trac.torproject.org/projects/tor/wiki/doc/meek.

[24] David Fifield and Yawning Angel. Go PT library. https://gitweb.torproject.org/pluggable-transports/goptlib.git.

[25] David Fifield, Nate Hardison, Jonathan Ellithorpe, Emily Stark, Roger Dingledine, Phil Porras, and Dan Boneh. Evading censorship with browser-based proxies. In *Proceedings of the 12th Privacy Enhancing Technologies Symposium (PETS 2012)*. Springer, July 2012.

[26] David Fifield, Chang Lan, Rod Hynes, Percy Wegmann, and Vern Paxson. Blocking-resistant communication through domain fronting. *Proceedings on Privacy Enhancing Technologies*, 2015(2):46–64, 2015.

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

[28] GeoLite2 Free Downloadable Databases.
https://dev.maxmind.com/geoip/geoip2/geolite2/.

[29] GetTor.
https://www.torproject.org/projects/gettor.

[30] Globaleaks: The open-source whistleblowing software. https://www.globaleaks.org/.

[31] Angus Grigg. WeChat's privacy issues mean you should delete China's No. 1 messaging app .
https://www.afr.com/news/world/asia/wechats-privacy-issues-mean-you-should-delete-chinas-no1-messaging-app-20180221-h0wgct.

[32] Haven: Keep watch. https://guardianproject.github.io/haven/.

[33] Martin Hilbert. The bad news is that the digital access divide is here to stay: Domestically installed bandwidths among 172 countries for 1986–2014. *Telecommunications Policy*, 40(6):567 – 581, 2016.

[34] iCepa: iOS system-wide VPN based Tor client. https://github.com/iCepa/iCepa.

[35] ICT Indicators in Brief (Egypt). www.mcit.gov.eg/Upcont/Documents/Publications_2122017000_Flyer_new_December_2016_EN_21_02_2017.pdf.

[36] Improve AS number and name coverage (switch maxmind to RIPE Stat). https://trac.torproject.org/projects/tor/ticket/26585#comment:3.

[37] infolabe-anomalies - detected anomalies in tor user numbers. http://lists.infolabe.net/lists/listinfo/infolabe-anomalies.

[38] IP2Location. https://www.ip2location.com/.

[39] Rob Jansen, John Geddes, Chris Wacek, Micah Sherr, and Paul Syverson. Never been KIST: Tor's congestion management blossoms with kernel-informed socket transport. In *Proceedings of 23rd USENIX Security Symposium (USENIX Security 14)*, San Diego, CA, August 2014. USENIX Association.

[40] Rob Jansen and Nicholas Hopper. Shadow: Running Tor in a box for accurate and efficient experimentation. In *Network and Distributed System Security Symposium (NDSS)*, 2012. See also https://shadow.github.io.

[41] Rob Jansen and Aaron Johnson. Safely measuring Tor. In *ACM Conference on Computer and Communications Security (CCS)*, 2016. See also https://github.com/privcount.

[42] Rob Jansen, Florian Tschorsch, Aaron Johnson, and Björn Scheuermann. The sniper attack: Anonymously deanonymizing and disabling the Tor network. In *Network and Distributed System Security Symposium (NDSS)*, 2014.

[43] Aaron Johnson, Rob Jansen, Nicholas Hopper, Aaron Segal, and Paul Syverson. PeerFlow: Secure load balancing in Tor. *PoPETs*, 2017(2):74–94, 2017.

[44] Marijn Kruisselbrink. File API. W3C working draft, W3C, October 2017. https://www.w3.org/TR/2017/WD-FileAPI-20171026/.

[45] List of most-downloaded Google Play applications. https://en.wikipedia.org/wiki/List_of_most-downloaded_Google_Play_applications#Paid_applications_with_one_million_or_more_downloads.

[46] Karsten Loesing. Performance of requests over the Tor network. Technical Report 2009-09-001, The Tor Project, September 2009.

[47] Karsten Loesing, Steven J. Murdoch, and Roger Dingledine. A case study on measuring statistical data in the Tor anonymity network. In *Proceedings of the Workshop on Ethics in Computer Security Research (WECSR 2010)*, LNCS. Springer, January 2010.

[48] Daniel Luchaup, Kevin P. Dyer, Somesh Jha, Thomas Ristenpart, and Thomas Shrimpton. LibFTE: A Toolkit for Constructing Practical, Format-Abiding Encryption Schemes. In *USENIX Security Symposium (USENIX Security)*, 2014.

[49] Akshaya Mani, T. Wilson-Brown, Rob Jansen, Aaron Johnson, and Micah Sherr. Understanding tor usage with privacy-preserving measurement. In *Proceedings of the Internet Measurement Conference 2018, IMC 2018, Boston, MA, USA, October 31 - November 02, 2018*, pages 175–187, 2018.

[50] Nick Mathewson. The tor proposal process. https://gitweb.torproject.org/torspec.git/tree/proposals/001-process.txt, January 2007.

[51] Nick Mathewson. Denial-of-service attacks in Tor: Taxonomy and defenses. Technical Report 2015-10-001, The Tor Project, October 2015.

[52] Meron Moges-Gerbi and Chris Giles. Ethiopia's parliament swears in new prime minister. https://edition.cnn.com/2018/04/02/africa/ethiopia-new-pm-abiy-ahmed/index.html.

[53] Steven J. Murdoch and George Danezis. Low-cost traffic analysis of Tor. In *Proceedings of the 2005 IEEE Symposium on Security and Privacy*. IEEE CS, May 2005.

[54] Milad Nasr, Alireza Bahramali, and Amir Houmansadr. DeepCorr: Strong flow correlation attacks on tor using deep learning. In *ACM Conference on Computer and Communications Security (CCS)*, 2018.

[55] United Nations. Universal Declaration of Human Rights. https://www.un.org/en/universal-declaration-human-rights/.

[56] Onion Browser: An open-source, privacy-enhancing web browser for iOS. https://onionbrowser.com/.

[57] OnionPerf. https://github.com/robgjansen/onionperf.

[58] Onionshare. https://onionshare.org/.

[59] OONI Explorer. https://explorer.ooni.torproject.org/world/.

[60] OONI Explorer China. https://explorer.ooni.torproject.org/country/CN.

[61] OONI Explorer Egypt. https://explorer.ooni.torproject.org/country/EG.

[62] OONI Explorer Ethiopia. https://explorer.ooni.torproject.org/country/ET.

[63] OONI Explorer Iran. https://explorer.ooni.torproject.org/country/IR.

[64] OONI Explorer Turkey. https://explorer.ooni.torproject.org/country/TR.

[65] OONI Explorer Venezuela. https://explorer.ooni.torproject.org/country/VE.

[66] OONI Measurement Statistics. https://api.ooni.io/stats.

[67] OONI Partnership Program. https://ooni.torproject.org/get-involved/partnership-program/.

[68] Ooni probe. https://ooni.torproject.org/install/.

[69] Ooni tests. https://ooni.torproject.org/nettest/.

[70] Open design process. http://opendesignkit.org/process/.

[71] Orbot: Tor for Android. https://guardianproject.info/apps/orbot/.

[72] Vasilis Pappas, Elias Athanasopoulos, Sotiris Ioannidis, and Evangelos P. Markatos. Compromising anonymity using packet spinning. In *Proceedings of the 11th Information Security Conference (ISC 2008)*, September 2008.

[73] Mike Perry. TorFlow: Tor network analysis. In *Workshop on Hot Topics in Privacy Enhancing Technologies (HotPETs)*, 2009.

[74] The Tor Project. TC: A tor control protocol (version 1). https://spec.torproject.org/control-spec, June 2005.

[75] The Tor Project. Tor specifications. https://spec.torproject.org/, November 2018.

[76] Qubes OS: A reasonably secure operating system. https://www.qubes-os.org/.

[77] Resumable.js. http://www.resumablejs.com/.

[78] Ricochet. https://ricochet.im/.

[79] RIPEstat.
https://stat.ripe.net/.

[80] Bruce Schneier. Censorship in the Age of Large Cloud Providers.
https://www.schneier.com/essays/archives/2018/06/censorship_in_the_ag.html.

[81] Securedrop: The open-source whistleblower submission system. https://securedrop.org/.

[82] Snowflake. https://trac.torproject.org/projects/tor/wiki/doc/Snowflake.

[83] TAILS: The Amnesic Incognito Live System. https://tails.boum.org/.

[84] Berhan Taye, Maria Xynou, Leonid Evdokimov, and Moses Karanja. Ethiopia: Verifying the unblocking of websites. https://ooni.torproject.org/post/ethiopia-unblocking/.

[85] SLC Team. List of Websites and Apps Blocked in China [Updated Aug 2018].
https://startuplivingchina.com/list-websites-apps-blocked-china/.

[86] Tencent.
https://en.wikipedia.org/wiki/Tencent.

[87] Tencent social networks.
https://www.tencent.com/en-us/system.html.

[88] The design and implementation of the Tor Browser.
https://www.torproject.org/projects/torbrowser/design/.

[89] The Guardian Project. https://guardianproject.info/.

[90] Tor Metrics.
https://metrics.torproject.org/.

[91] Tor Metrics Library.
https://metrics.torproject.org/metrics-lib.html.

[92] Tor Metrics News.
https://metrics.torproject.org/news.html.

[93] Tor pluggable transport specification. https://spec.torproject.org/pt-spec.

[94] Tor Research Portal.
https://research.torproject.org/.

[95] Tor uplift project. https://wiki.mozilla.org/Security/Tor_Uplift.

[96] Tor ux research. https://trac.torproject.org/projects/tor/ticket/27010.

[97] Torbirdy. https://trac.torproject.org/projects/tor/wiki/torbirdy.

[98] Torperf.
https://gitweb.torproject.org/torperf.git/.

[99] Matt Traudt and juga. Simple Bandwidth Scanner technical details.
https://sbws.readthedocs.io/en/latest/specification.html.

[100] Uganda imposes WhatsApp and Facebook tax 'to stop gossip'. https://www.bbc.com/news/world-africa-44315675.

[101] Tom van der Woerdt. Tor Protocol Specification Proposal 255. https://github.com/torproject/torspec/blob/master/proposals/255-hs-load-balancing.txt.

[102] Venezuela blocks access to the Tor network. https://www.accessnow.org/venezuela-blocks-tor/.

[103] Vpns can still be used in china despite march 31 ban. https://www.zdnet.com/article/vpns-can-still-be-used-in-china-despite-march-31-ban/.

[104] What is behind Ethiopia's wave of protests? https://www.bbc.com/news/world-africa-36940906.

[105] Whonix: A High Security Method of Surfing the Internet. https://www.whonix.org/.

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

[107] Joss Wright, Alexander Darer, and Oliver Farnan. Detecting internet filtering from geographic time series. *CoRR*, abs/1507.05819, 2015.

[108] Maria Xynou, Arturo Filastò, Mahsa Alimardani, Sina Kouhi, Kyle Bowen, Vmon, and Amin Sabeti. Internet Censorship in Iran: Network Measurement Findings from 2014-2017. https://ooni.torproject.org/post/iran-internet-censorship/.

[109] Maria Xynou, Arturo Filastò, Amnesty International, and Girma Walabuma. Ethiopia: Evidence of social media blocking and internet censorship. https://ooni.torproject.org/post/ethiopia-report/.