MassBrowser:
Unblocking the Web for the Masses, By the Masses

Milad Nasr*, Anonymous*, and Amir Houmansadr
University of Massachusetts Amherst
{*milad,amir*}*@cs.umass.edu*
Project Website: https://massbrowser.cs.umass.edu/

Abstract

Existing censorship circumvention systems fail to offer reliable circumvention without sacrificing their users' QoS, or undertaking high costs of operation. We design a new circumvention system, called MassBrowser, with the objective of addressing such practical weaknesses of existing designs. Our system is based on a new design principle, called "the separation of properties," that states that circumvention systems should be tailored for circumvention as opposed to offering additional properties like anonymity. We combine various state-of-the-art circumvention techniques to make MassBrowser significantly resistant to blocking, while keeping its cost of operation small ($0.001 per censored client per month).

We have built and deployed MassBrowser as a fully operational system with end-user software for regular Internet users. A key part of MassBrowser's design is using non-censored Internet users to run volunteer proxies to help censored users. We perform the first user study on the willingness of typical Internet users in helping circumvention operators. We have used the findings of our user study in the design of MassBrowser to encourage wide adoption by volunteers; particularly, our GUI software offers high transparency, control, and safety to the volunteers.

*The first two authors made equal contribution.

1 Introduction

The Internet plays a crucial role in today's social and political movements by facilitating the free circulation of speech, information, and ideas; democracy and human rights throughout the world critically depend on preserving and bolstering the Internet's openness. Consequently, repressive regimes, totalitarian governments, and corrupt corporations regulate, monitor, and restrict the access to the Internet, which is broadly known as Internet *censorship*. The techniques commonly used to enforce censorship include IP address blocking, DNS hijacking, and TCP content filtering [14, 35, 37, 56] to block access to certain destinations or to prevent certain forms of content from being transmitted. To ensure compliance and to detect undercover political/social activists, repressive regimes additionally utilize advanced networking tools, including deep packet inspection (DPI), to prevent the use of the censorship circumvention technologies by their citizens [32, 33, 54, 66].

To restore the openness of the Internet, researchers have designed and deployed an arsenal of tools [12, 14, 15, 28, 29, 38, 43, 47, 57, 61, 63, 67] that help users bypass censorship. Such tools, known as *circumvention systems*, deploy a variety of techniques ranging from IP indirection to onion routing to traffic obfuscation [35, 56].

Key shortcomings of existing systems. Unfortunately, existing circumvention systems suffer from one or all of the following weaknesses: *(1) Eas-*

ily blocked: A majority of circumvention systems work by setting up *proxy* servers outside the censorship regions, which relay the Internet traffic of the censored users. This includes systems like Tor, VPNs, Psiphon, etc. Unfortunately, such circumvention systems are easily blocked by the censors by enumerating their limited set of proxy server IP addresses [54, 55, 64, 66]. *(2) Costly to operate:* To resist proxy blocking by the censors, recent circumvention systems have started to deploy the proxies on shared-IP platforms such as CDNs [41], App Engines [23], and Cloud Storage [11], a technique broadly referred to as *domain fronting* [18]. This mechanism, however, is prohibitively expensive [42] to operate for large scales of users. *(3) Poor QoS:* Proxy-based circumvention systems like Tor and it's variants [29, 40, 59] suffer from low quality of service (e.g., high latencies and low bandwidths). This is due to various factors such as the small number of proxies to clients, and the large volume of client traffic used to access voluminous content (like pirated movies). *(4) Hard to deploy:* Several circumvention systems proposed in the literature are impractical to be used at large scale due to various reasons. For instance, decoy routing systems [28,34,67] require wide adoption by Internet ISPs, and tunneling systems [29,31,40,59] can be disabled by third-party service providers they use for tunneling.

Our approach. The goal of this paper is to design a new circumvention system that offers practical circumvention by tackling the aforementioned shortcomings of circumvention systems. We base our design on a *new design principle* not considered by prior circumvention designs. Our principle, which we call the *separation of properties* principle, states that *the key property expected from an effective circumvention system is blocking resistance, and it does not need to provide other properties such as browsing privacy or anonymity.* Our real-world observation [13, 24] suggests that the majority of censored users are solely interested in blocking resistance, but not other properties like anonymity. For instance, typical censored users trust any open proxy or VPN provider just to get access to censored websites despite the trivial absence of anonymity and browsing privacy [24]. Therefore, we argue that a circumvention system needs to be designed in a way to optimize blocking resistance; bundling additional properties like anonymity is the main reason for the majority of weaknesses mentioned above. For censored users who need additional properties like anonymity, they can achieve those by cascading the circumvention system with other privacy-enhancing technologies like anonymity systems (and, consequently trading off QoS and cost to get those additional properties). In this paper, we demonstrate that designing a circumvention system based on this principle enables us to offer strong blocking resistance in addition to practical QoS and low cost of operation. For instance, the separation of properties principle allows us to run single-proxy circumvention connections, improving the QoS-cost tradeoff. It also enables us to limit the use of our circumvention system only for accessing circumvention content. This not only reduces congestion on the proxies (therefore improving the QoS-cost tradeoff), but also increases the potential number of volunteer proxies by minimizing the legal risks of running circumvention proxies (as witnessed for general purpose circumvention systems like Tor [51] [9]).

Contributions. We design a new circumvention system, called MassBrowser, that aims at addressing the weaknesses of prior designs, as discussed above. That is, MassBrowser aims at offering reliable blocking resistance while providing practical QoS and low operational costs. The *core idea* of MassBrowser is to use normal Internet users with access to the free Internet, which we call *Buddies*, as relays to proxy censored web traffic for censored users, which we call *Clients*. This will address the challenges of circumvention systems discussed above in different ways. First, the diversity, abundance, and dynamicity of the IPs used by the Buddies will make any attempt of IP enumeration by the censors prone to significant collateral damage (i.e., due to falsely blocking significant non-circumvention traffic). Particularly, normal Internet users connect from behind NATs, therefore blocking NATed Buddies has similar collateral damage impact on the censors as in the (impractically expensive) domain fronting systems [18] (i.e., to block a NATed Buddy, the censors will need

to block the Buddy's subnet). Second, MassBrowser combines various state-of-the-art circumvention techniques including CacheBrowsing [26] and Domain Fronting [18] to optimize the QoS of circumvention connections while minimizing the operational costs of circumvention. As shown in Section 8.2, we estimate the total cost of deploying MassBrowser to be no more than *$0.001 per active client per month*.

We have built a fully operational implementation of MassBrowser, with end-user graphical user interfaces for MassBrowser Client and Buddy users with minimal technical background. We have been testing MassBrowser's software for several months using volunteer clients from inside censored countries. MassBrowser will make a real-world impact only with wide adoption by volunteers who run MassBrowser Buddies. Therefore, a major challenge to MassBrowser's success is to facilitate and encourage wide-scale adoption by volunteer relays. Towards this, we perform *the first* user study on the willingness of Internet users in voluntarily helping circumvention technologies. The results of our user study suggest that *a significant fraction of Internet users are willing to run software that helps censored users—if they get guarantees on their safety and security*. Advised by this, we build MassBrowser to provide high levels of safety and security to the volunteer Buddies. Particularly, we design a user-friendly GUI software for Buddies that provides them with *transparency* and *full control* on how their computers are used to help censored users. For instance, our Buddy software enables a proxy operator to control the websites she feels comfortable proxying traffic to, as well as the volume of traffic she is willing to proxy for censored users.

Note that our implementation of MassBrowser supports *connecting through Tor* for users who need anonymity in addition to blocking resistance (at the expense of a lowered QoS comparable to Tor's QoS). More specifically, our Buddy software enables a volunteer to optionally become a Tor bridge as well. Therefore, existing Tor bridges can adopt MassBrowser as a pluggable transport [49]. We evaluate MassBrowser's cost of operation when used as a Tor pluggable transport, showing that it is drastically cheaper than meek [41], while they both offer similar blocking resistance properties (both meek and MassBrowser aim at increasing the censors' collateral damage by making use of shared IP addresses).

In summary, we make the following main contributions:

1. We have designed a new circumvention system, MassBrowser, with the objective of addressing practical weaknesses of existing designs, particularly, blocking resistance, QoS, and operational costs.

2. We have performed the first user study on the willingness of normal Internet users in helping circumvention systems.

3. We have implemented and deployed MassBrowser as a fully operational system. We have used the findings of our user study to build a usable GUI software for clients and volunteers. Our software is pending an IRB approval for public release.

2 Background on Circumvention Systems

Internet censorship continues to remain as the biggest global threat to freedom of speech, ideas, and information [20]. The censors deploy various technique to implement censorship [14, 35, 37, 56]. To help the censored users regain open access to the Internet, researchers and practitioners have designed and deployed an arsenal of tools known as *circumvention systems* [12, 14, 15, 28, 29, 38, 43, 47, 57, 61, 63, 67]. Censorship authorities utilize their censorship technology to prevent the use of such censorship circumvention technologies by their citizens [32, 33, 54, 66], i.e., they block circumvention systems. In the following, we overview the major classes of circumvention systems and their weaknesses.

Proxy-based Systems The most common approach used by circumvention systems is to run network proxies outside the censorship region, and use them to relay the traffic of censored users to the censored Internet destinations. Many in-the-wild circumvention systems such as Tor [16], Psiphon [50],

Anonymizer [8], and the VPN services [46,48] deploy circumvention proxies in different ways to help censored users. Some circumvention systems [8, 46, 58] use simple, single-hop proxies, while others [16,36,50] use more complex models for proxy deployment. Tor, in particular, has implemented various *pluggable transports* [47,49] to further hinder blocking through obfuscating the characteristics of Tor traffic.

Domain Fronting Domain fronting [18] is a more recent approach for setting up censorship resistant services. In this approach, the domain fronted service is hosted on network architectures that share IP addresses among various other services, particularly, content delivery networks (CDNs), App Engines, and Cloud Computing. Tor's domain fronted pluggable transport, called meek [41], relays users' Tor traffic through proxy servers hosted on CDNs and App Engines. CloudTransport [11] can also be considered as a domain fronting system, as it runs proxies on shared cloud storage services. Domain fronted services are harder to block by the censors, as blocking their shared IP addresses will also block the benign network services hosted on the same platform. For instance blocking a CDN-hosted circumvention proxy requires the censors to block all the websites hosted on the same CDN as well.

CacheBrowsing CacheBrowsing [26,68] is another recent circumvention approach that leverages CDNs. In CacheBrowsing, a censored client locates the censored content hosted on CDNs (using various bootstrapping mechanisms), and directly fetches the located censored content from their hosting CDNs with no need to use any proxies. In contrast to domain fronting, CacheBrowsing is significantly cheaper [26, 42] as the CDN service is paid for by the content provider not the circumvention system. On the other hand, CacheBrowsing can be used to unblock only the censored content hosted on CDNs.

Protocol Tunneling Several circumvention designs work by tunneling traffic through popular Internet services that are unlikely to be entirely blocked by the censors. For instance, FreeWave [29] tunnels circumvention traffic through VoIP services like Skype, and CovertCast [40] tunnels traffic through video streaming services. Alternatively, Rook [59] and Castle [25] tunnel traffic through gaming applications, and Sweet [31] tunnels through email communications. To block a tunneling circumvention system, the censors may have to block its oblivious hosting services as well, causing them collateral damage [22]. On the downside, tunneling circumvention systems offer impractical QoS (e.g., high latencies and low bandwidth) due to the limitations imposed by their hosting services.

Table 1: Weaknesses of major types of circumvention systems

Category	Easily blocked	Costly	Poor QoS	Deployability
Proxy-Based	●	◐	●	○
Domain Fronting	○	●	○	○
CacheBrowsing	○	○	●	○
Tunneling	○	◐	●	◐
Decoy Routing	○	◐	○	●

Decoy Routing Decoy routing is an alternative approach for censorship circumvention [28, 34, 45, 67] whose design is motivated by the ease of IP address blocking of traditional proxy-based circumvention systems. In decoy routing, censorship circumvention is implemented with help from a number of friendly Internet autonomous systems, called *decoy ASes*. Each decoy AS modifies some of its routers (e.g., its border routers) such that they deflect the Internet traffic of censored users to the blocked Internet destinations requested by the users. By design, decoy routing defeats IP address blocking, however, it is prone to particular routing-based blocking attacks known as RAD [30,44,52]. Requiring deployment by a number of in-the-wild ISPs is a major obstacle to the real-world deployment of decoy routing systems.

2.1 Weaknesses of Existing Systems

Existing circumvention systems, overviewed above, suffer from one or multiple of the following challenges (summarized in Table 1):

1) Easy to block: A majority of circumvention systems work by setting up *proxy* servers outside the censorship regions, which relay the Internet traffic of the censored users. This includes systems like Tor [16], VPNs [46,48], Psiphon [50], etc. Unfortunately, such circumvention systems are easily blocked by the censors who enumerate their limited set of

proxy server IP addresses [54,55,64,66]. While IP address filtering is the most commonly practiced mechanism to disable circumvention systems, the censors can also use more advanced techniques like traffic analysis and active probing to block circumvention systems [21, 27, 54, 55, 66].

2) Costly to operate: To resist proxy enumeration by the censors, recent circumvention systems have started to deploy some of their proxies on shared-IP platforms such as CDNs, App Engines, and Cloud services, a technique known as *domain fronting* [18]. Blocking a domain-fronted proxy causes the censors collateral damage; However, due to the prohibitively high costs of domain fronting [42], domain fronting is not used for circumvention proxying at scale (recent proposals suggest to use domain fronting only for circumvention signaling, but not for proxying [53]). CloudTransport [11] takes a similar approach by running proxies on cloud storage platforms, similarly suffering from high costs of operation.

3) Poor QoS: Proxy-based circumvention systems like Tor suffer from low quality of service (e.g., high latencies). This is due to various factors such as the small ratio of the number of proxies to clients, as well as the large volume of client traffic used to access voluminous non-censored content such as copyright violated multimedia shared using torrent applications. Tunneling circumvention systems like FreeWave [29], Sweet [31], and CoverCast [40] offer low bandwidth and high latencies to the clients as they are constrained by the quality of service provided by their hosting services. CDNBrowsing systems [26, 68] offer good latencies but can only be used to browse specific types of censored websites (unless combined with other circumvention solutions, as discussed in this paper).

4) Hard to deploy: Several circumvention systems proposed in the literature are impractical to be used at large scale due to various reasons. For instance, decoy routing systems [28, 34, 45, 67] require wide adoption by Internet ISPs, and tunneling systems [29,31,40] can be disabled by third-party service providers they use for tunneling.

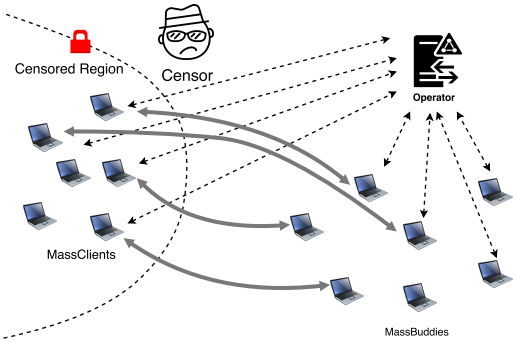

Figure 1: The basic overview of our tool, users in the censoring region try to connect to a set of helpers outside of the censoring region.

3 Sketch of our Approach

3.1 High-Level Design of MassBrowser

The *core idea* of MassBrowser is to use normal Internet users with access to the free Internet as relays to proxy censored web traffic for censored users. Figure 1 shows the high-level architecture of MassBrowser, which is composed of three main types of entities:

1. *Clients:* These are the censored Internet users residing inside censored regions. Clients use MassBrowser's client software to gain access to the censored Internet.
2. *Buddies:* A Buddy is a Internet user with access to the free Internet (e.g., living out of censored regions) who voluntarily installs MassBrowser's proxy software to help censored clients get around censorship by proxying their traffic.
3. *Operator:* This is MassBrowser's central management system that operates the whole system by taking various roles such as registering Clients and Buddies as well as mapping and connecting Clients and Buddies.

We will provide further details about these components and their interactions in the rest of the paper.

3.2 How MassBrowser Addresses Circumvention Issues

In the following, we summarize how MassBrowser aims at addressing the major circumvention issues discussed in Section 2.1. This will be further expanded later on.

1) Blocking resistance: As discussed above, proxy enumeration is the most common mechanism of blocking circumvention systems in the wild. We argue that the main cause of this weakness is twofold: first, the small number of proxy IP addresses, enabling enumeration and blocking by the censors; second, the proxy IPs used by circumvention systems are static and dedicated for circumvention, therefore, once identified they can get easily blacklisted by the censors with little collateral damage. Domain fronting shares proxy IPs with other services, however, the operational costs are prohibitive for large scale deployment, as discussed earlier. Our approach is to deploy a large number of proxies with help from volunteer users. Our volunteers are *typical* Internet users, as opposed to dedicated servers, and therefore they connect from NATed networks and move between networks. Therefore, the diversity, abundance, and dynamicity of the IPs used by our volunteer proxies will make any attempt of IP enumeration by the censors prone to significant collateral damage (i.e., due to mistakenly blocking non-circumvention traffic).

2) Cost of operation: Similar to domain fronting [18] and CloudTransport [11], MassBrowser makes use of shared IP addresses to defeat IP enumeration. By contrast, MassBrowser is significantly cheaper to operate as the voluminous circumvention traffic is proxied by volunteer proxies with shared IPs. MassBrowser's operator runs as a domain-fronted service to defeat blocking, but it only generate a tiny volume of traffic that results in negligible operational costs.

3) QoS: MassBrowser uses several complimentary techniques to provide a high QoS. First, supported by a user study, we use various social engineering techniques to attract a large number of typical Internet users as volunteers. Second, MassBrowser leverages CacheBrowsing [26] to minimize the traffic load on the volunteer proxies. Third, MassBrowser's design is based on the principle of "separation of properties," allowing it to further improve the QoS by using single-hop proxies and restricting the use of proxies for censored content only. Note that clients who need anonymity can use MassBrowser as a gateway to Tor (at the cost of lowered QoS). That is MassBrowser can be used as a pluggable transport for Tor [49].

4) Deployment feasibility: Unlike approaches like decoy routing systems [28, 34, 67] and tunneling systems [29,31,40], MassBrowser does not require cooperation/deployment from third-party Internet operators. Also, we have built a user-friendly GUI for both volunteers and clients to attract a larger number of adopters.

3.3 Comparison to Similar Systems

MassBrowser is not the first circumvention design to propose to use volunteer proxies by typical Internet users. In the following, we contrast MassBrowser to these alternative systems. Note that *none of these systems are deployed/released to the public* by the time of submitting this paper.

FlashProxy [17, 19]: FlashProxy is run with help from volunteer websites. A volunteer website loads a particular JavaScript on each of its visitors that turn them into proxies for censored clients. Even though a FlashProxy volunteer website presents a banner to its visitors informing them of its involvement with FlashProxy, the visitors have no way to opt out except by refraining from visiting that website. Therefore, FlashProxy does not appear to be favorable to benign websites. Additionally, the censors can retaliate by censoring (or attacking) the volunteer websites. Another issue of FlashProxy is that the proxies are short-lived, e.g., a visitor to a FlashProxy website is a proxy only during its visit to the website. Note that the FlashProxy project is deprecated.

uProxy [58]: uProxy is another system that uses volunteer Internet users as proxies for censored users. uProxy works as a plugin in the Chrome web browser, and uses the WebRTC protocol to connect a censored user to a volunteer proxy. uProxy does not have any central operator as in MassBrowser; instead, a uP-

roxy censored user is supposed to know a friend outside the censorship region to act as her proxy. In some sense, uProxy acts as a "private" proxy for a censored client, very much similar to how private VPNs work. Unfortunately, this is not a scalable solution as many censored users do not know a friend willing to help in the free Internet. MassBrowser, on the other hand, is designed as a generally accessible circumvention system. uProxy is not released to the public at the time of submission.

Snowflake [53]: Snowflake is the successor of the FlashProxy project and uses some of the core communication protocols of uProxy, e.g., its WebRTC communication schemes. Similar to FlashProxy, Snowflake converts the visitors of some volunteer websites to circumvention proxies by loading a JavaScript. Therefore, we argue that a major challenge to Snowflake is adoption by volunteer websites: a volunteer website may get the target of censorship or cyberattacks by the censors, and therefore we do not expect adoption by major websites. Note that deployment by low-visitor websites does not help as the number of proxies is proportional to the number of the volunteer website's visitors. Also, similar to Flashproxy, users in Snowflake have no way to opt out except by refraining from visiting the volunteer websites.

4 User Survey on Circumvention Participation

In addition to our MassBrowser, several recent proposals for circumvention systems work by using volunteer proxy operators [17, 53, 58], as introduced earlier. The key factor to the success of such systems is adoption by (a large number of) volunteers. We have conducted *the first* user study on the willingness of uncensored Internet users in helping censored users by running a circumvention software. We created an online survey questionnaire and distributed it among several groups of uncensored Internet users asking them about their interest and preferences in running circumvention software. In this section, we present the outcome of our survey.

Ethics. We received an IRB approval for our survey before distributing it among participants. The participants' data was collected and processed anonymously and voluntarily, and was stored on secure computer servers.

4.1 Survey Participants

We distributed our online survey among three groups of participants (a total of 76 participants):

1. `CS`: We advertised our survey among the members of a computer science department's social group. The participants are computer science students, faculty, and researchers working on various areas of computer science, but from the same institution.
2. `OSN`: We distributed our survey on an online social networking platform. We expect the participants to be from diverse ethnic/technical backgrounds, though we did not request their fine-grained background information to keep the survey anonymous.
3. `XCensored`: We distributed our survey among the members a group formed of people originally from a major censoring country, but currently living in the US. The members of the group come from diverse educational backgrounds (e.g., mostly non-CS).

Table 4 (Appendix B) shows the demography of the participants for each of the surveys. Note that providing the demographic information was optional, therefore we did not receive the demographic information from all participants.

While the number of our survey participants is not significantly large, we believe it gives us enough confidence to infer conclusions on the willingness of normal Internet users in fighting censorship. Note that we could recruit a much larger number of participants by conducting a paid survey (e.g., using Amazon Mechanical Turk); we refrained from doing so as we believe asking "paid" survey participants about their willingness to do a "voluntary" service would bias the survey's outcome.

4.2 Survey Format

The full survey questionnaire is included in Appendix E. We introduced the circumvention software as follows: "Suppose that there is a software called Helper that when you install on your laptop/desktop computer, it will assist the Internet users in censored countries to get around censorship". We also told the participants "Assume that someone you trust guarantees that the Helper software will not make any harm to your computer or your network. Also, the use of Helper is transparent to you and does not interfere with your work". Then, we asked each participant about their willingness in running the "Helper" circumvention software and their preferences, as described in the following.

4.3 Survey Results

Willingness to Install and Run the Software. We first asked each participant if they were willing to install and run the Helper software (introduced above)—for free. We told them that "running Helper does not cost you anything, but also does not earn you money." We also mentioned "you can completely control the use of Helper (as will be asked in the follow up questions)". If a participant expressed her unwillingness, we asked her if she would participate if she got paid.

Figure 2 shows the aggregation of the responses. As we can see, *a significant fraction of surveyed participants (75%) express their willingness to run the circumvention software completely voluntarily (e.g., for free), if they trust the software to be harmless.* This is very encouraging to volunteer-based circumvention systems: with adequate security and safety protections, one can expect a decent number of volunteer circumvention helpers. Also, about 36% of people who were unwilling to participate for free ($\approx 10\%$ of all participants) expressed their willingness to participate if they were paid.

Comparing the responses from different group of participants, we see that the `XCensored` participants are less willing to participate in running a circumvention software. We speculate this to be due to their family/business bonds with the censoring countries, e.g., they may have family members in the censored country and travel there frequently.

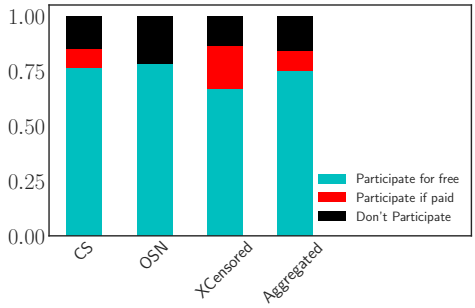

Figure 2: Survey participants' willingness to install and run a software that helps censored users.

Content Type Preferences. We asked the willing participants about the type of content they are willing to proxy for censored users. We warned them that "some censored users (whom you don't know) will use your computer to connect to censored Internet websites. So your Internet provider may assume that you are browsing those websites yourself. What kind of websites do you feel comfortable (and allow) to be proxied through your computer by censored users."

Figure 3 shows the results for various categories of censored content. We observe that only a small fraction of participants (about 12%) are willing to relay *any* type of traffic, while others have reservations on the type of traffic they proxy. We particularly see that video streaming websites are the least accepted and the News/Scientific websites are the most accepted categories.

Censored Country Preferences. We asked the willing participants if they preferred to help censored users from any specific countries. As shown in Figure 4, the majority of willing participants ($\approx 80\%$) had no particular preference on the ethnicity of the censored users they were helping.

Bandwidth Devotion Preferences. We asked the willing participants how much of their "unused bandwidth" they are willing to allocate to the circumvention software. As shown in Figure 5, the majority of participants are willing to devote substantial fractions of their "unused" bandwidth for circumvention. We particularly see that around 40% of participants

are willing to donate more that 50% of their unused bandwidth for circumvention.

4.4 Incorporating the Results Into MassBrowser

We incorporate the key findings of our survey in the design of MassBrowser, as described in the following.

- We observe that a significant fraction of our survey participants are willing to install and run a circumvention software to help censored users (given guarantees on their safety and security). We find this encouraging the design of emerging volunteer-based circumvention systems like MassBrowser. We also see that most of the willing participants do this for free, therefore we do not see a need for an incentivization mechanism for MassBrowser.
- Our participants were told that someone they trust guarantees their security and safety. Towards this, we have released MassBrowser's code as open source software, and we are undergoing third-party *code review* by a reputable organization.
- We observe that the willing volunteers have various reservations about how they are willing to proxy circumvention traffic, particularly the type and volume of the content they proxy. Therefore, we deploy MassBrowser in a way to enable its volunteers to adjust how the software runs on their computers. We believe that the sparsity of volunteers in popular systems like Tor is due to the lack of such control and guarantees, especially given recent incidents for Tor exit operators [9, 51].
- To enable participation from a wide spectrum of volunteers, we build a simple, user-friendly GUI interface for volunteer proxy operators.

5 MassBrowser's Threat Model

We assume that Clients are users located inside censoring regions who wish to gain access to blocked websites, however, Buddies are users residing in non-censoring regions with unrestricted access. Each censored region's network is monitored by a censor who is capable of observing all communication from her censored region to the outside. The censors are able to block or throttle any traffic between users inside the censored region and the outside network and will decide to do so based on the traffic's destination or blacklisted keywords and protocols which it detects in the traffic. Censors are also able to act as Clients or Buddies in order to gain information about the system and to disrupt the system to the best of their ability. However, we assume that censors are not capable of tampering with users' devices (e.g., installing monitoring softwares on their devices).

We assume the players in our system to be *rational*. A rational censor tries to minimize the costs and collateral damages incurred by its actions, such as blocking benign destinations or receiving objections from its users. Buddies are rational in that they are willing to help censored users as long as it does not pose any risk to themselves. For example, a Buddy will not let Clients use her device to deploy network attacks (e.g., port scan, sending spam email, etc.).

We also assume that the *censors do not penalize normal users for the sole act of using a circumvention software*, unless the websites accessed are directly related to major criminal offenses, e.g., child pornography, drug trafficking, etc. Although using circumvention tools is considered illegal in many censoring countries, penalizing Internet users solely for using a circumvention software is very rare in most countries [13] (instead, the censors have penalized people who *operated* a circumvention software for others [2, 5]). For example, as of 2017, Facebook has over 17 million users from Iran accounting for over 20% of the population [10], despite it having been blocked for more than 8 years. The fact that users are willing to provide public information on a blocked website confirms the negligible risk of using circumvention software in such countries. Our threat model assumes that the censored clients are aware of, and accept the (negligible) risks of using a circumvention software (so, a journalist may opt to use a different system).

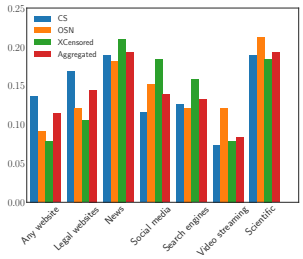

Figure 3: Participants' willingness to proxy different types of content.

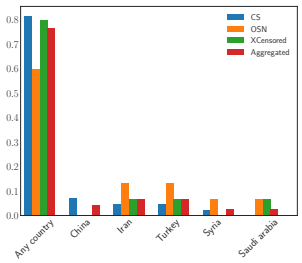

Figure 4: Participants' willingness to help censored users from different countries.

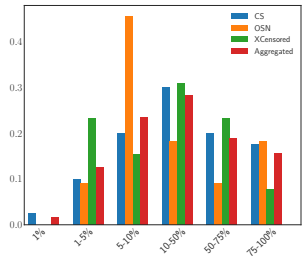

Figure 5: The fraction of unused bandwidth that participants are willing to devote to a circumvention software

5.1 The Separation of Properties Principle

We base our design on a **new design principle** not considered by prior academic circumvention designs. Our principle, which we call the *separation of properties* principle, states that *the key property expected from an effective circumvention system is blocking resistance, and it does not need to provide other properties such as browsing privacy and anonymity.* Observations suggest that the majority of censored users are merely interested in blocking resistance, but not necessarily other properties such as anonymity. A typical censored user would be content using any open proxy or VPN provider in order to gain access to it's desired censored websites despite the trivial absence of anonymity and browsing privacy within most of these systems.

We argue that a circumvention system needs to be designed in a way to provide blocking resistance only; bundling additional properties like anonymity is the main cause of the mentioned weaknesses of prior circumvention systems. A circumvention system should be targeted towards censored users who wish to gain open access to information such as news, social media, video sharing, and other such content otherwise inaccessible to them. For censored users who need additional properties like anonymity and privacy, such as activists, whistleblowers, and other cautious users, they can achieve those in the same fashion as users in non-censoring regions, i.e., by using anonymity systems such as Tor. A concerned censored user can achieve this goal by cascading a circumvention system with other privacy-enhancing technologies and anonymity systems (and, consequently trading off QoS and cost for those additional properties). In fact, *our implementation of MassBrowser supports tunneling traffic through Tor for users who need anonymity*, in which case MassBrowser will serve as a Tor pluggable transport.

We will show that designing a circumvention system based on this principle enables us to offer a high blocking resistance with a practical balance between QoS and cost of operation. Particularly, the separation of properties principle allows us to run single-proxy circumvention connections, which improves the QoS-cost tradeoff. Also, the principle allows us to constraint the use of our circumvention system for accessing censored content only. This not only reduces congestion on the proxies (therefore improving the QoS-cost tradeoff), but also increases the potential number of volunteer proxies by minimizing the legal risks of running circumvention proxies as witnessed for general purpose circumvention systems like Tor [9, 51].

6 MassBrowser's Design Decisions

In this section, we detail our design decisions aimed at addressing each of the issues discussed in Section 2.1.

6.1 Blocking Resistance

We combine the following mechanisms to provide a high blocking resistance in MassBrowser.

Use of shared, dynamic proxy IPs to resist IP enumeration As MassBrowser proxies are run by typical Internet users, blocking them is costly and prone to collateral damage. First, a typical Buddy volunteer will have a NAT IP address, therefore sharing a public IP address with other users/services in the same network. For instance, a Buddy connecting from a coffee shop will share a public IP with other users in the area (we will describe how MassBrowser enables connections despite NAT). Additionally, a typical Buddy will frequently change IP addresses, e.g., by moving across networks, amplifying the collateral damage. Second, by employing various social engineering techniques, described later, we hope that MassBrowser will attract a large number of volunteer proxies making IP enumeration unreliable and costly (in addition to its high collateral damage).

Traffic Obfuscation and Encryption All MassBrowser communications are encrypted to prevent deep-packet inspection. Additionally, MassBrowser deploys traffic obfuscation mechanisms to remove protocol fingerprints and prevent censors from detecting MassBrowser traffic based on traffic patterns like packet timings and sizes.

Domain Fronting the Operator MassBrowser's Operator runs as a domain fronted service [18]. As discussed earlier, a domain fronted service runs behind a network infrastructure with shared IPs (e.g., CDNs), therefore blocking it will cause significant collateral damage to the censors. Although domain fronting is a relatively expensive technique, the costs of domain fronting MassBrowser's Operator is very low due to the small volume of control traffic generated by the Operator, as shown in Section 8.2.

6.2 Addressing Cost and QoS Issues

As discussed earlier in Section 3, existing circumvention systems suffer from either low QoS or high cost of operation (or both). We argue that the main reason for the poor QoS/high cost of existing circumvention systems is twofold: 1) the large volume of traffic generated by the clients, and 2) the small number of proxies compared to clients. We take the following two complimentary approaches to enable high QoS censorship circumvention with lost costs of operation.

6.2.1 Minimizing the volume of proxied traffic

We use the following complimentary techniques to minimize the traffic load on MassBrowser proxies.

Targeting censored content only Existing circumvention tools like Tor, VPNs, etc., are designed to offer not just censorship circumvention, but also other properties like anonymity. Therefore, in order to use such tools, users are required to relay all of their traffic through the proxies, regardless of whether the traffic being proxied is indeed censored or not. As a result, a large fraction of the proxied traffic is for non-censored content which could have been obtained directly, with higher QoS and without imposing extra costs on the circumvention system. We base the design of MassBrowser on the "separation of properties" property introduced in Section 5. Therefore, we limit the use of MassBrowser Buddies for circumvention only, i.e, any traffic proxied through the Buddies is traffic which could not have been directly obtained by the users.

CacheBrowsing MassBrowser uses a recent circumvention technique called CacheBrowsing [26, 68] to further minimize the load on the proxies. As introduced in Section 3, in CacheBrowsing a client directly fetches a censored object hosted on CDNs from the hosting CDN's edge servers, without using any proxies. However, a limitation of CacheBrowsing is that it can only retrieve censored content which is hosted on a CDN and accessible through HTTPS. We integrate CacheBrowser into the client software of MassBrowser's clients. That is, a MassBrowser client will fetch the CDN-hosted censored content directly from CDNs, and only use MassBrowser Buddies for the censored content not hosted on a CDN. As shown in Section 8.1, this saves a lot of bandwidth on the relays.

Strategic proxy assignment to prevent DoS by Sybils In MassBrowser, clients discover Buddies and

connect to them with help from the Operator, i.e., by connecting to the Operator. MassBrowser's Operator uses a proxy assignment mechanism, described later, to prevent the censors from learning a large fraction of Buddies. Note that, the censors can *not* block the discovered (NATed) Buddies, but can overload them to consume their circumvention capacities.

6.2.2 Incentivizing volunteers

The QoS of a proxy-based circumvention system critically depends on the number of its proxies. We use the following approaches to increase the number of volunteer proxies. We envision a large fraction of MassBrowser Buddies to be from typical Internet users with little technical background. We therefore design a GUI-based client software for Buddies to offer a user-friendly experience, transparency, and full, fine-grained control. Our Buddy GUI offers the following features.

Imperceptible operation Our Buddy GUI runs imperceptible and does not interfere with the volunteer's normal activities. The volunteer user will only need to perform a one-time installation and setup of the relay software, and may then let it operate without any further changes.

Transparency on usage Our Buddy GUI offers the volunteer with information on how the proxy is being used.

Enable relays to limit proxied bandwidth The relay software enables a volunteer Buddy operator to specify how much bandwidth she is willing to donate to MassBrowser. Even a small donated bandwidth can help MassBrowser clients due to the bandwidth minimization mechanisms discussed above.

Enable relays to whitelist destinations Our MassBrowser relay software enables a volunteer to only proxy traffic to certain destinations she is comfortable with. A major hindering factor to proxy traffic for others is the potential legal consequences of relaying traffic to controversial destinations. In MassBrowser, relays whitelist the categories of destinations they are willing to proxy traffic to, e.g., a relay can decide to relay traffic only to major news websites.

Optional economic incentives Future versions of MassBrowser may incorporate economic incentives for volunteers, either as the form of a service like Bitcoin mining by clients, or monetary compensation. We leave the investigation of incorporating such economic incentives with MassBrowser to future work.

6.3 Privacy and Anonymity

Our threat model results in privacy properties for MassBrowser that are different than prior academic solutions like Tor. According to our *separation of properties* principle, a user may or may not require each of the two distinct properties, anonymity and censorship resistance. The former can be achieved through a system such as Tor, however Tor can easily be blocked by a government wishing to do so (e.g. using public Tor directories) and has poor QoS for users who do not need the anonymity.

MassBrowser is solely a censorship circumvention system, it is designed to be fast and hard to block but it does not provide anonymity guarantees. Only if a censored user also desires anonymity, he or she may do so by accessing an anonymity network (i.e, Tor) through MassBrowser. MassBrowser is made fully compatible with Tor and can be easily used with TorBrowser similar to using any other pluggable transport. Essentially, MassBrowser will act as a Tor bridge for gaining access to the Tor network. Each Buddy may also choose to allow or disallow Tor traffic.

Please also refer to Section 9 for a detailed discussion of privacy in MassBrowser.

6.4 Deployment

Unlike some of the previous circumvention systems like decoy routing [28, 34, 67] and tunneling systems [25,29,31,59], MassBrowser's operation does not rely on any third-party operators like autonomous systems and services providers. We have implemented and deployed an operational version of MassBrowser, and have received and IRB exemption for it by our university's research compliance office.

7 MassBrowser's Implementation Details

In Section 3, we introduced the high-level design of MassBrowser, and in Section 6 we discussed MassBrowser's design decisions. In this section, we describe the detailed implementation of MassBrowser. MassBrowser is composed of three main parties, as introduced in Section 3.1: Clients are censored Internet users who use MassBrowser to bypass censorship, Buddies are volunteer-run proxies who proxy traffic for the Clients, and the Operator is the central entity who maintains and oversees the MassBrowser system.

Please see the details of our code and implementation in Appendix D.

7.1 Connecting Users Behind NAT

The MassBrowser network is composed of users located behind NATs and without publicly accessible IP addresses. As a result, connecting to Buddies cannot be done by conventional means and requires the use of *NAT traversal* techniques [39, 65]. Typical NAT traversal techniques, however, may not be applicable for all transport protocols depending on the type of NAT the user is behind, i.e., how the NAT maps local to global IP addresses. Matthews et al. [39] do a detailed analysis of different NAT deployments in the Internet and how NAT traversal techniques apply to them. We categorize MassBrowser users (i.e., both Clients and Buddies) into three groups based on the type of the NAT they have.

TCP Reachable users are those with whom it is possible to initiate a TCP connection to, either directly or via some existing NAT traversal technique.

UDP Reachable users are the users whom we are not able to initiate TCP connections with, but are still able to send UDP packets to them via some NAT traversal technique. These users reside behind *Restricted NATs* as defined by Wing et al. [65].

Unreachable users are the users who are located behind NATs that prevent the use of any NAT traversal technique. Wing et al. [65] classify these NATs as *Symmetric NATs*.

In order to establish a connection between a Client and a Buddy, either the Client or the Buddy must initiate the TCP connection with the other party. The NAT restrictions imposed on each of the users is one of Operator's deciding factors in matching Clients and Buddies. It is also used to determine which party should initiate the connection, as depicted in Table 2. As can be seen, when both of the parties are reachable, the Client initiates the connection. When both uses are UDP reachable, MassBrowser's software tunnels a TCP connection through an established UDP tunnel. If none of the parties are reachable, a MassBrowser connection can not be established between the parties, so the Operator will not match two unreachable parties.

Table 2: The party who initiates a MassBrowser connection depending on the parties' type of NAT.

		Buddy		
		TCP-Reach	UDP-Reach	Unreach
Client	TCP-Reach	Client	Buddy	Buddy
	UDP-Reach	Client	Client	Buddy
	Unreach	Client	Client	✗

7.2 Assigning Buddies to Clients by the Operator

The Operator is in charge of coordinating Client and Buddy communications and providing Clients with online Buddies to use as relays. The Operator assigns Buddies to Clients with the following considerations.

Buddy destination whitelists Buddies may whitelist destinations they are willing to proxy traffic to based on their content types. The Operator actively maintains Buddy whitelist settings. When queried for new Buddies, the Operator will respond with Buddies that allow the intended destinations in their whitelists.

Buddy loads The MassBrowser system is a heterogeneous network composed of machines with varying processing powers and network bandwidths. The Operator approximates a Buddy's available resources based on its bandwidth limit settings, observed quality of service history, and the number of currently active Clients assigned. This is used to balance the load on Buddies when assigning Buddies to new Clients.

Parties' NAT types The Operator also considers the NAT types of users in matching Clients and Buddies. We described the matching details above in Section 7.1.

Sybil attack protection As discussed in Section 9, a censor can *not* block the Buddies that she obtains from the Operator, nor can she identify their clients (since Buddy IPs are NATed). However, a resourceful censor may overload the identified Buddies in order to consume their circumvention capacity (i.e., DoS the Buddies). Note that this will be a costly DoS attack due to the symmetry between the load on the attacker and victim. Nonetheless, our Operator can deploy standard Sybil protection mechanisms against such an expensive DoS attack. We have particularly borrowed and implemented the Sybil protection mechanism used by rBridge [62]. This mechanism uses a reputation system for clients in order to provide them with new proxy information. Similar to rBridge, we have deployed an invitation-based mechanism for accepting new clients.

7.3 Minimizing Load on Proxies

Existing popular proxy-based circumvention tools proxy all of the users traffic through proxy servers regardless of whether the content being proxied is in fact censored or not. In contrast, MassBrowser views each request individually and decides how to best handle the request independent of other incoming requests. Figure 6 (Appendix C) shows this process. MassBrowser only relays the request through Buddies if it identifies the requested content to be censored for the Client. Furthermore, if the requested resource is CacheBrowsable [26], MassBrowser will fetch the content (either fully or partially [68]) directly from CDNs imposing no load on the Buddies.

To perform such per-request targeted proxying, the MassBrowser Client must have a means of identifying censored and CacheBrowsable URLs. For this purpose, the Operator actively maintains a database of MassBrowser-supported websites along with detailed information about the different resources and requests of each website. The MassBrowser Client software also keeps a regularly-synced local version of this database containing information for user's websites of interest.

Note that the Operator itself is deployed as a "domain fronted" service. All communications between the Client and the Operator, such as those required for updating the local database, requesting Buddies and NAT traversal, will be domain fronted and thus unblockable by the censors.

7.4 Encryption and Traffic Obfuscation

All communication between Clients and Buddies must be encrypted in order to resist DPI attacks deployed by the censors. Both parties will encrypt their messages using a symmetric cipher with a shared secret key that they share through the Operator. Our implementation currently uses AES 256 for Client-to-Buddy encryption.

We also implement traffic obfuscation to protect MassBrowser's traffic against traffic analysis attacks [21, 27, 60]. Particularly, we build a custom implementation of the obfsproxy [47] Tor pluggable transport tailored to work with our MassBrowser implementation. The obfuscation algorithm removes identifiable traffic patterns, making the Client-Buddy protocol seem like benign peer-to-peer traffic.

7.5 Communication Sessions in MassBrowser

We define a MassBrowser *session* to be a connection between a Client and a Buddy. Upon receiving a request from the browser, the Client checks whether the request can be handled with any of the currently active sessions the Client has, i.e., whether any of the connected Buddies will accept the request in their whitelisted categories. If no such session is found, the Client will need to ask the Operator to assign it a new session with a suitable Buddy that will accept the request.

The Operator will select a Buddy to assign to the Client and will notify both parties to establish a new session. Each session has the following attributes:

1. *Allowed content types* This is the list of content types that the Client is allowed to obtain

through this session.

2. *Shared Keys and Cipher Suite* All communications between the Client and Buddy are encrypted with a shared key and cipher suite shared through the Operator.
3. *Obfuscation method* In order to prevent fingerprinting attacks on the Client-Buddy communication protocol, the Operator may instruct the users to use one of the available obfuscation algorithms if the censoring region is known to deploy DPI attacks.
4. *Connection initiator* Based on the type of NAT the users are located behind, the Operator will instruct one of the users to initiate the connection with the other using an appropriate NAT traversal technique, as described earlier.
5. *Expiration time* Each session is only valid within a defined time period. The Client will have to ask to renew the session if he wishes to continue usage beyond the expiration time. This is to perform load balancing on Buddies over time.

The Operator will send the details of a new session to the Client and Buddy. The party who has been selected as the connection initiator will then attempt to establish a connection with the other party. The receiving party will keep the session in a list of pending sessions until either the connection is established or the session expires. Each session can only be used once, and both parties will notify the Operator once the session connection has been established. Figure 7 (Appendix C) shows the messages involved in establishing a session, and how traffic is relayed between Clients and Buddies.

8 Performance Evaluation

8.1 Buddy Bandwidth Contribution

Our analysis of the top 1000 Alexa website homepages [7] finds the average size of each webpage to be 2.4 MB. We found 41% of the generated traffic by these pages to be CacheBrowsable (note that some most webpages are only partially CacheBrowsable [68]). Therefore, in order to load a page through MassBrowser the client will only need to proxy ≈ 1.4 MB through the Buddies. The Akamai State of the Internet Connectivity Report [6] estimates the Internet bandwidth of an average user living in the United States in 2017 to be 18.7 Mbps. Assuming volunteers will provide MassBrowser with 25% of their unused bandwidth, an average Buddy in the United States will contribute 4.7 Mbps when not using the internet, which translates into a page load every 2.5 seconds. Also, recall that in MassBrowser, the bandwidth of Buddies is solely used for loading censored Client pages.

8.2 Costs of Operation

Ensuring low operational cost is one of the primary design goals of MassBrowser. The (bulky) circumvention traffic of MassBrowser clients is handled by volunteer Buddies. Therefore, the only operational cost of MassBrowser is imposed by running the Operator. Recall that the Operator is deployed as a domain fronted service, i.e., hosted on a CDN, in order to allow unblockable access to the censored users. In this section, we show that while domain fronting is known to be prohibitively expensive for proxying [42], it imposes little costs on MassBrowser as it is only used for its control traffic.

There are three factors that contribute to the Operator's operational costs:

1) Number of Client-Requested Sessions Per Day: Each session established between a Client and a Buddy is capable of serving any volume of traffic to different destinations as long as they satisfy the content type restrictions imposed by the Buddy. Therefore, it is unlikely that a Client will require more than a few active sessions at any given time. Our evaluation of a typical Client shows that 20 sessions per day is sufficient for typical web browsing.

2) Size of Session Objects: Upon creation of a new session between a Client and a Buddy, the Operator will need to exchange some protocol messages to the two parties. The exchanged information is composed of a 500 byte fixed-size segment containing details about the IP addresses, ports, NAT types, connection initiator, secret key, and the session expiration date, along with a variable-size segment listing the content types that will be accepted on the session (each content type takes 12 bytes). Therefore, the

overall traffic load on Operator for each session is ≈ 1000 bytes.

3) Size of Database: The Operator maintains a database containing information on how to browse different censored websites supported by the Buddies. While the number such unique domains for every website could be high, the database stores the domains in regex format, combining groups of similar domains with identical censorship information into single entries. The majority of the websites have at most 50 entries in Operator's database; given that each entry is around 1KB, each website will use at most 50KB in the database.

Based on these factors, we predict Operator's operational costs on the Amazon AWS.

Cost of Running the Operator Servers: We estimated every user to request 20 sessions per day. For 10,000 users this requires 200,000 requests which would amount to an average of 2 requests per second. An AWS EC2 *t2.micro* instance, costing at about \$0.015 an hour, will be sufficient for handling this load of requests generated by 10,000 users. *The monthly cost will amount to \$0.0011 per user.*

Cost of Deploying on CDNs: We have hosted the Operator on Amazon Cloudfront CDN. Amazon Cloudfront charges based on the volume of traffic, and the locations of the CDN edge servers used. Note that Operator's communications with Clients are *not* latency sensitive; therefore, it suffices for the Operator to use a cheap CDN service (we use a service with \$0.01 per GB). As estimated above, each user will request 600 sessions per month, for which the Operator will need to send 600 KB of control data to the Clients; this costs *\$0.00006 per user each month.* The user will also need to sync her local database with Operator, resulting in a one-time 50 KB data transfer for each supported website, which costs \$0.0000005 per user for every website.

Comparing costs with meek: Meek [41] is a Tor pluggable transport that relays Tor traffic through domain fronted proxies to evade censorship. In order to operate, meek must proxy all of the users' traffic through CDN servers. As a result, unlike MassBrowser the costs of operating meek is in direct correlation with the user's bandwidth usage. As we saw in the previous analysis, we estimate the cost for a MassBrowser user with 600 sessions per month to be \$0.00006 each month using Amazon Cloudfront CDN regardless of the types of websites browsed (e.g., video streaming, news, etc.). If we assume each session to be just for one website load and each website to have an average of 2.4 MB (as we measured), then the same user using meek over Amazon Cloudfront CDN will cost $600 * 0.0024 * 0.01 = \$0.014$, which is over 200 times the cost of the user on MassBrowser. Note that in real life each session will be used to browse multiple websites and may require higher traffic (e.g., for video streaming), therefore, the cost gap will be even greater in favor of MassBrowser.

Table 3: Average page load of different website over Tor, using MassBrowser as bridge for Tor and MassBrowser.

Website	**Tor (s)**	**MassBrowser + Tor (s)**	**MassBrowser** (s)
Google.com	19.6	20.3	2.6
Youtube.com	27.3	25.6	6.3
Facebook.com	27.4	30.4	6.6
Baidu.com	7.5	10.1	1.7
Wikipedia.com	29.5	22.3	1.1

8.3 MassBrowser as a Tor Transport

As mentioned before, MassBrowser can be used as a Tor pluggable transport, i.e., a client who needs anonymity can connect to a Buddy who allows Tor traffic. We measured the time to load the top 100 Alexa websites with Tor, using MassBrowser as bridge for Tor and MassBrowser without Tor. We browsed each website 50 times over each setting and compute the average time to load the websites. Table 3 presents the load times for different websites. On average loading each website on Tor takes more than 16 seconds than using MassBrowser. Using MassBrowser as a Tor bridge does not significantly change the load times compared to using Tor with no pluggable transport; therefore, MassBrowser's added latency on Tor is negligible.

9 Discussion of Privacy and Resilience

In this section, we discuss the privacy and censorship resistance properties of MassBrowser's components. Please refer to Appendix A for potential questions not discussed here.

9.1 Client Privacy

A. Privacy against Buddies In MassBrowser, a Buddy has the same privacy threats to its Clients as the threats imposed by a network observer (e.g., ISPs and transit routers) on typical Internet users

Anonymity against Buddies: As discussed earlier, providing client anonymity is not a design goal for MassBrowser based on the separation of properties principle. Therefore, a Buddy can learn the destinations being accessed by her connected Clients —this is similar to how a typical network observer (like an ISP) can learn browsing patterns for typical Internet users. Note that, like a normal Internet user, a MassBrowser client needing anonymity can use an anonymity system like Tor—through MassBrowser —(i.e., by connecting to Buddies that support Tor as one of their allowed destinations).

Confidentiality from Buddies: A Buddy will not be able to see the communication content for HTTPS destinations, which includes the majority of services hosting sensitive user data like social networking websites and search engines. A proxy, however, will be able to see a client's communication content to an HTTP destination. This is not any different than how network observers (e.g., ISPs) can learn

Surveillance by the censor-run Buddies: A powerful organization that runs numerous Buddies for user surveillance is not different than a nation state or ISP with access to Internet routers. Real-world observations over the years have shown that censoring governments tend not to penalize their users for the sole act of circumventing censorship. A Client using MassBrowser is only able to evade censorship for a select number of supported websites and these websites exclude any which are known to have legal consequences for the users.

Identification by censors who know Buddies The Buddies obtained by a censoring client from Operator can not be used to learn any information about the Clients who use these Buddies. This is because different Clients connecting to the same Buddy will make connections through different IP address and port combinations due to NAT.

B. Privacy against Operator Unlike traditional circumvention tools like Psiphon, Anonymizer, and Lantern, in MassBrowser the operator of the circumvention system is separate from the parties relaying traffic. Therefore, the Operator would not be able to directly observe user traffic and would only occasionally learn categories of content accesses by Clients.

9.2 Buddy Privacy

Privacy against Clients A Client using a Buddy will only learn the *NATed* IP address of that Buddy, but no other information. As Client-Buddy assignment is performed by the Operator, the Client can not choose the Buddy to connect to.

Privacy against Operator The Operator will have access to the Buddy's settings such as their whitelisted content types and specified bandwidth limits. The Buddy's IP address will also be exposed to the Operator, however similar to the Clients, this is the *NAT* IP address of the Buddy, which is also visible to any other web service the Buddy connects to regularly.

9.3 Censorship Resistance

IP blocking the Operator In MassBrowser, the Operator has the same protection against IP blocking as domain fronting systems. That is, being run on public CDNs, the censors will have to block a whole CDN in order to block the Operator.

IP blocking Buddies The IP enumeration techniques that censors practice against traditional circumvention systems like Tor [54, 66] will *not* work against MassBrowser Buddies. This is because the censors can only obtain the NAT IPs of the Buddies; blacklisting such IPs will have similar collateral damage as blocking domain fronting systems.

Traffic fingerprinting All MassBrowser communications between Clients and Buddies are obfuscated (and encrypted) using a tailored variant of obfsproxy [47] to prevent known traffic fingerprinting attacks [21, 27, 60]. All Client traffic to the Operator is protected with domain fronting.

DoS attacks through censor's Sybils As discussed before, a censor who obtains a Buddy from the Operator can not block that Buddy, nor can he identify the Buddy's clients. However, a resourceful censor may overload the obtained Buddies to consume their available circumvention capacities. Note that such an attack is not a strong DoS attack, as the load on the attacker and victim is symmetric (asymmetry is the key property of real-world DoS attacks). Nonetheless, our Operator deploys standard Sybil protection mechanisms as explained earlier.

References

[1] Amazon Cloudfront CDN. https://aws.amazon.com/cloudfront.

[2] Behind China's VPN Crackdown, A 'Game Of Cat And Mouse' Continues. https://www.npr.org/sections/alltechconsidered/2017/08/04/541554438.

[3] Django Web Framework. https://www.djangoproject.com/.

[4] Electron Framework. https://electronjs.org/.

[5] Iranian regime arrests VPN software seller. https://www.ncr-iran.org/en/news/human-rights/13125-iranian-regime-arrests-vpn-software-seller.

[6] Akamai. State of the Internet Connectivity Report, Q1 2017. https://www.akamai.com/us/en/about/our-thinking/state-of-the-internet-report/global-state-of-the-internet-connectivity-reports.jsp.

[7] Alexa. Top Websites. https://www.alexa.com/topsites.

[8] Anonymizer. https://www.anonymizer.com/.

[9] Dark net raids were overblown by police. http://www.bbc.com/news/technology-29987379.

[10] Azali, M. Infographic: Facebook Usage Statistics in Iran. http://techrasa.com/2017/08/16/infographic-facebook-usage-statistics-iran/.

[11] Brubaker, C., Houmansadr, A., and Shmatikov, V. CloudTransport: Using Cloud Storage for Censorship-Resistant Networking. In *PETS* (2014).

[12] Burnett, S., Feamster, N., and Vempala, S. Chipping Away at Censorship Firewalls with User-Generated Content. In *USENIX Security* (2010).

[13] Penalties For Using VPN In Various Countries. https://www.vpnunlimitedapp.com/blog/penalties-for-using-vpn/.

[14] Defeat Internet Censorship: Overview of Advanced Technologies and Products. http://www.internetfreedom.org/archive/Defeat_Internet_Censorship_White_Paper.pdf, 2007.

[15] Dingledine, R., and Mathewson, N. Design of a Blocking-Resistant Anonymity System. https://svn.torproject.org/svn/projects/design-paper/blocking.html.

[16] Dingledine, R., Mathewson, N., and Syverson, P. Tor: The Second-Generation Onion Router. In *USENIX Security Symposium* (2004).

[17] Fifield, D., Hardison, N., Ellithorpe, J., Stark, E., Boneh, D., Dingledine, R., and Porras, P. Evading censorship with browser-based proxies. In *Privacy Enhancing Technologies* (2012), Springer, pp. 239–258.

[18] Fifield, D., Lan, C., Hynes, R., Wegmann, P., and Paxson, V. Blocking-resistant Communication through Domain Fronting. In *PETS* (2015).

[19] FlashProxy. http://crypto.stanford.edu/flashproxy/.

[20] Freedom on the Net 2017. https://freedomhouse.org/sites/default/files/FOTN_2017_Final.pdf, 2017.

[21] Geddes, J., Schuchard, M., and Hopper, N. Cover Your ACKs: Pitfalls of Covert Channel Censorship Circumvention. In *CCS* (2013).

[22] China's GitHub Censorship Dilemma. http://mobile.informationweek.com/80269/show/72e30386728f45f56b343ddfd0fdb119/.

[23] GoAgent proxy. https://code.google.com/p/goagent/.

[24] Everyone's Guide to By-Passing Internet Censorship for Citizens Worldwide. http://www.nartv.org/mirror/circ_guide.pdf.

[25] Hahn, B., Nithyanand, R., Gill, P., and Johnson, R. Games without frontiers: Investigating video games as a covert channel. In *Security and Privacy (EuroS&P), 2016 IEEE European Symposium on* (2016), IEEE, pp. 63–77.

[26] Holowczak, J., and Houmansadr, A. CacheBrowser: Bypassing Chinese Censorship without Proxies Using Cached Content. In *The 22nd ACM Conference on Computer and Communications Security (CCS)* (2015).

[27] Houmansadr, A., Brubaker, C., and Shmatikov, V. The Parrot Is Dead: Observing Unobservable Network Communications. In *S&P* (2013).

[28] Houmansadr, A., Nguyen, G., Caesar, M., and Borisov, N. Cirripede: Circumvention Infrastructure Using Router Redirection with Plausible Deniability. In *CCS* (2011).

[29] Houmansadr, A., Riedl, T., Borisov, N., and Singer, A. I Want My Voice to Be Heard: IP over Voice-over-IP for Unobservable Censorship Circumvention. In *NDSS* (2013).

[30] Houmansadr, A., Wong, E. L., and Shmatikov, V. No Direction Home: The True Cost of Routing Around Decoys. In *NDSS* (2014).

[31] Houmansadr, A., Zhou, W., Caesar, M., and Borisov, N. SWEET: Serving the Web by Exploiting Email Tunnels. In *PETS* (2013).

[32] How Iran Censors The Internet. http://www.popsci.com/technology/article/2013-03/how-iran-censors-internet-infographic.

[33] Iran Reportedly Blocking Encrypted Internet Traffic. http://arstechnica.com/tech-policy/2012/02/iran-reportedly-blocking-encrypted-internet-traffic.

[34] Karlin, J., Ellard, D., Jackson, A., Jones, C., Lauer, G., Mankins, D., and Strayer, W. Decoy Routing: Toward Unblockable Internet Communication. In *FOCI* (2011).

[35] Khattak, S., Elahi, T., Simon, L., Swanson, C. M., Murdoch, S. J., and Goldberg, I. SoK: Making sense of censorship resistance systems. *Proceedings on Privacy Enhancing Technologies 2016*, 4 (2016), 37–61.

[36] Lantern. https://getlantern.org/.

[37] Leberknight, C., Chiang, M., Poor, H., and Wong, F. A Taxonomy of Internet Censorship and Anti-censorship. http://www.princeton.edu/~chiangm/anticensorship.pdf, 2010.

[38] Mahdian, M. Fighting Censorship with Algorithms. In *Fun with Algorithms* (2010).

[39] Matthews, P., Mahy, R., and Rosenberg, J. Traversal using relays around nat (turn): Relay extensions to session traversal utilities for nat (stun).

[40] McPherson, R., Houmansadr, A., and Shmatikov, V. CovertCast: Using Live Streaming to Evade Internet Censorship. In *Privacy Enhancing Technologies (PETS)* (2016).

[41] meek Pluggable Transport. https://trac.torproject.org/projects/tor/wiki/doc/meek.

[42] [tor-project] summary of meek's costs, march 2017. https://lists.torproject.org/pipermail/tor-project/2017-April/001097.html.

[43] Moghaddam, H., Li, B., Derakhshani, M., and Goldberg, I. SkypeMorph: Protocol Obfuscation for Tor Bridges. In *CCS* (2012).

[44] Nasr, M., and Houmansadr, A. Game of decoys: Optimal decoy routing through game theory. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security* (2016), ACM, pp. 1727–1738.

[45] Nasr, M., Zolfaghari, H., and Houmansadr, A. The waterfall of liberty: Decoy routing circumvention that resists routing attacks.

[46] Nobori, D., and Shinjo, Y. VPN Gate: A Volunteer-Organized Public VPN Relay System with Blocking Resistance for Bypassing Government Censorship Firewalls. In *NSDI* (2014), pp. 229–241.

[47] A Simple Obfuscating Proxy. https://www.torproject.org/projects/obfsproxy.html.en.

[48] Perta, V., Barbera, M., Tyson, G., Haddadi, H., and Mei, A. A glance through the VPN looking glass: IPv6 leakage and DNS hijacking in commercial VPN clients. *Proceedings on Privacy Enhancing Technologies 2015*, 1 (2015), 77–91.

[49] Tor: Pluggable Transports. https://www.torproject.org/docs/pluggable-transports.html.en.

[50] Psiphon. http://psiphon.ca/.

[51] Access Now and EFF Condemn the Arrest of Tor Node Operator Dmitry Bogatov in Russia. https://goo.gl/nNmP86.

[52] Schuchard, M., Geddes, J., Thompson, C., and Hopper, N. Routing around decoys. In *ACM CCS* (2012).

[53] SnowFlake Pluggable Transport. https://github.com/keroserene/snowflake.

[54] Ten Ways to Discover Tor Bridges. https://blog.torproject.org/blog/research-problems-ten-ways-discover-tor-bridges.

[55] How Governments Have Tried to Block Tor. https://svn.torproject.org/svn/projects/presentations/slides-28c3.pdf.

[56] Tschantz, M. C., Afroz, S., Paxson, V., et al. SoK: Towards Grounding Censorship Circumvention in Empiricism. In *Security and Privacy (SP), 2016 IEEE Symposium on* (2016), IEEE, pp. 914–933.

[57] Ultrasurf. http://www.ultrareach.com.

[58] uProxy. https://www.uproxy.org/.

[59] Vines, P., and Kohno, T. Rook: Using video games as a low-bandwidth censorship resistant communication platform. In *Proceedings of the 14th ACM Workshop on Privacy in the Electronic Society* (2015), ACM, pp. 75–84.

[60] Wang, L., Dyer, K. P., Akella, A., Ristenpart, T., and Shrimpton, T. Seeing Through Network-Protocol Obfuscation. In *ACM CCS* (2015).

[61] Wang, Q., Gong, X., Nguyen, G., Houmansadr, A., and Borisov, N. CensorSpoofer: Asymmetric Communication Using IP Spoofing for Censorship-Resistant Web Browsing. In *CCS* (2012).

[62] Wang, Q., Lin, Z., Borisov, N., and Hopper, N. rBridge: User Reputation Based Tor Bridge Distribution with Privacy Preservation. In *NDSS* (2013).

[63] Weinberg, Z., Wang, J., Yegneswaran, V., Briesemeister, L., Cheung, S., Wang, F., and Boneh, D. StegoTorus: A Camouflage Proxy for the Tor Anonymity System. In *CCS* (2012).

[64] Wilde, T. Knock Knock Knockin' on Bridges' Doors. https://blog.torproject.org/blog/knock-knock-knockin-bridges-doors, 2012.

[65] Wing, D., Matthews, P., Mahy, R., and Rosenberg, J. Session traversal utilities for NAT (STUN).

[66] Winter, P., and Lindskog, S. How the Great Firewall of China Is Blocking Tor. In *FOCI* (2012).

[67] Wustrow, E., Wolchok, S., Goldberg, I., and Halderman, J. Telex: Anticensorship in the Network Infrastructure. In *USENIX Security* (2011).

[68] Zolfaghari, H., and Houmansadr, A. Practical censorship evasion leveraging content delivery networks. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security* (2016), ACM, pp. 1715–1726.

A Potential Questions

Why cannot the Great Firewall blacklist millions of IP addresses? They can (at potentially high computation overhead) but this will cause them significant collateral damage due to false positive blockings. MassBrowser IP addresses are shared IP addresses that are used by benign non-circumvention traffic like VoIP, gaming, etc.

What's the difference between MassBrowser and Tor? MassBrowser and Tor are apples and oranges! MassBrowser is a system to defeat IP blocking—very much similar to meek and other Tor pluggable transports. In fact, as we have discussed, MassBrowser can be used as a Tor pluggable transport.

Even though your survey shows high interest from volunteers to deploy MassBrowser, in practice one DMCA complaint could be enough to scare all volunteers away. A key strength of MassBrowser compared to other volunteer-run circumvention systems is that the volunteers will not relay *any* controversial traffic. Volunteers only relay traffic to totally legal destinations like News and social networks, or act as a bridge to get to Tor. Also, each volunteer decides the destinations she proxies to (see Figure 8).

Roger Dingledine always argues that anything that's weaker than Tor will have inadequate security. How do you counter this? Please refer to our discussion of the "separation of properties" principle. Most of the users in China and Iran only need blocking resistance and do not need anonymity. For users who need anonymity they can use MassBrowser as a pluggable transport to Tor. When used as a pluggable transport, MassBrowser offers similar blocking resistance features to meek at a significantly lower cost of operation.

What if the censors intercept the communication between clients and buddies by applying SSL or TLS splits between country border and domain front Operator server to intercept, modify, and drop communication between the clients and operator? No circumvention system, including Tor, will work against a censoring adversary with such capabilities.

MassBrowser's certificate installation on client seems intrusive? Absolutely not! This is a local certificate. The certificate is generated locally on the user's own machine and never leaves the machine. The user is warned never to share the certificate, and advised to revoke the certificate if he or she no longer wishes to use MassBrowser. Also, our code is undergoing a code review by a third-party.

What if the censor blocks all p2p traffic? This will likely impact many MassBrowser connections, but also a significant number of legitimate p2p traffic like VoIP, gaming, file sharing, etc.

Do all Buddies have to be NATed? No! MassBrowser will work fine for Buddies who are not behind NATs as long as they are not blocked. For Buddies with static public IP addresses, MassBrowser's resistance against blocking is similar to other proxy-based circumvention systems. However, MassBrowser will only assign such Buddies to Clients with sufficient reputation as described in Section 7.2.

Will there be enough bandwidth available through Buddies to make the system practical? MassBrowser's design makes it require significantly less bandwidth than a system like Tor. This is because **1)** Tor traffic must pass through three hops

in the network, **2)** only a small number of the websites browed by a user will pass through the MassBrowser network, and **3)** only a small portion of a website's contents will pass through MassBrowser's network (due to MassBrowser's deployment of CacheBrowsing).

What if a nation-state (like UAE) penalizes its citizens just for using a circumvention system? Then MassBrowser will not be the right solution for the users of such countries. Fortunately, for major censoring governments, including China, Turkey, and Iran, there is almost no instance of such punishments.

B Survey Demography

Table 4 shows the demography of our survey participants.

Table 4: The demography of survey participants

		CS	OSN	XCensored	Aggregated
Gender	Male	76%	64%	73%	73%
	Female	19%	14%	26%	20%
	Not answered	4%	21%	0%	7%
Location	USA	89%	28%	100%	80%
	Europe	2%	50%	0%	10%
	Asia	6%	7%	0%	5%
	Other	0%	0%	0%	0%
	Not answered	2%	14%	0%	5%
Age	18-30	78%	57%	73%	73%
	30-40	12%	28%	26%	18%
	Above 40	8%	7%	0%	6%
	Not answered	0%	7%	0%	1%
Proficiency	High	86%	85%	53%	80%
	Medium	10%	7%	46%	17%
	Low	0%	0%	0%	0%
	Not answered	2%	7%	0%	3%

C MassBrowser Design Graphs

Figure 6 shows how MassBrowser client software decides how to handle a client request. Figure 7 shows the messages involved in establishing a session, and how traffic is relayed between Clients and Buddies.

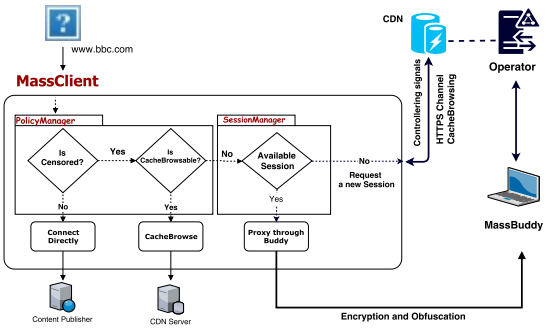

Figure 6: Optimizing proxying load in the MassBrowser client

D MassBrowser Code

We have fully implemented MassBrowser as an end-user software, and it is currently in the beta release state with users evaluating it. Our current implementation of MassBrowser supports Mac, Windows, and Linux operating systems, which is available at https://massbrowser.cs.umass.edu/. In the following we give details of our system implementation.

D.1 The Operator server

We have coded Operator server mostly in Python with the Django web framework [3]. We have hosted our Operator server on Amazon CloudFront CDN [1], therefore it is a domain fronted service and can not be blocked. Our Operator's API is accessible through both standard HTTP requests and WebSockets, though we refrain from using WebSocket connections for the Client in order to prevent introducing protocol fingerprints.

As previously mentioned, the Operator maintains a database of supported websites along with per-region censorship and CacheBrowsing information for all domains in the websites. To do so, the Operator has a *probing* component that regularly crawls the supported websites to identify domains and update its information.

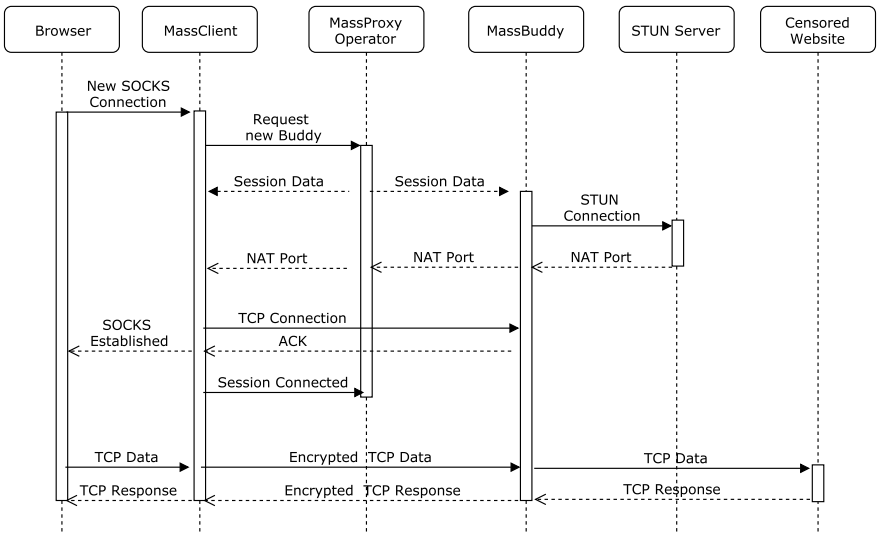

Figure 7: The session protocol between MassBrowser Clients and Buddies. In this example it is assumed the Client is selected as the connection initiator.

D.2 Buddy Software

We have coded our Buddy software in Javascript ES6 using NodeJS with a graphical user interface developed with the Electron framework [4]. In addition to the GUI interface, or Buddy software is also available as a command-line application for expert volunteers. The Buddy actively maintains a WebSocket connection to the Operator, and will be notified of newly created sessions on this channel.

The Buddy software allows volunteers to have full transparency and control over their desired settings including bandwidth limits, destination whitelists and Client blacklists (Figure 8 displays a snapshot of a Buddy volunteer configuring her destination whitelists through the GUI). The Buddy software runs with minimal interference from the user. It is able to run in the background while providing an easily accessible switch for disabling the Buddy's activities on the users demand.

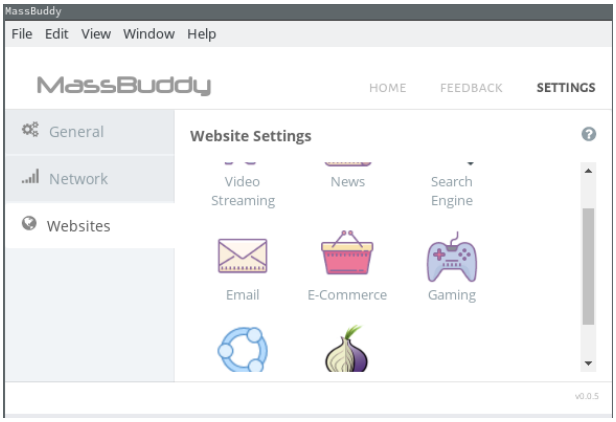

Figure 8: The settings page in the MassBrowser Buddy software allowing the user to select it's allowed content types

D.3 Client Software

We have implemented our Client software with NodeJS with an Electron based GUI. A client application, e.g., a web browser, can connect to the Client software via a SOCKS proxy. On the first run, the Client software will walk the user through a setup wizard which will assist them in configuring their preferred browsers to use MassBrowser. The current implementation of Client software provides a setup wizard for the Firefox browser only, but an

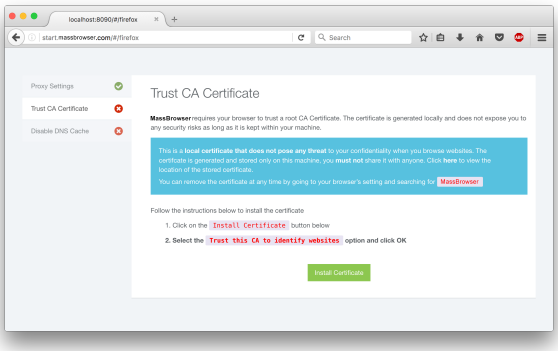

Figure 9: The Client setup wizard. This page is asking the user to trust the local MassBrowser root certificate and providing users with details on how to keep their connections safe.

expert client can set up any web browser to use the Client software. Figure 9 displays the Client setup wizard for Firefox.

The MassBrowser Client software requires to see each individual request, even when encrypted with TLS. In the normal case, the proxied TLS requests would not be visible to the Client software since it does not own the website certificates. To enable the interception of TLS connection by Client, the setup wizard adds a *locally created root certificate* to the client's browser during the initial setup. Note that the root certificate does not leave the client's computer, and therefore the client is secure as long as she does not share the certificate with others (Figure 9 shows how the user is informed during the setup). Client uses this certificate to "locally" man-in-the-middle MassBrowser's TLS connections to perform load optimizations like CacheBrowsing.

E Complete User Survey

The following is online survey we used in our study.

- Are you willing to voluntarily install and run Helper on your personal laptop/desktop (so you help censored Internet users)? Assume that running Helper does not cost you anything, but also does not earn you money. Also, assume that you can completely control the use of Helper (as will be asked in the follow up questions).
  - Yes
  - No
- Are you willing to install and run Helper on your personal laptop/desktop (so you help censored Internet users) if you get paid?
  - Yes
  - No
- What fraction of your unused Internet bandwidth are you willing to allocate to Helper (the unused bandwidth is the bandwidth you are not using anyways)?
  - 1%
  - $1 - 5\%$
  - $5 - 10\%$
  - $10 - 50\%$
  - $50 - 75\%$
  - $75 - 100\%$
-
- When you install the Helper software, some censored users (whom you don't know) will use your computer to connect to censored Internet websites. So your Internet provider may assume that you are browsing those websites yourself. What kind of websites do you feel comfortable (and allow) to be proxied through your computer by censored users?
  - I am OK with all websites
  - I am OK with all legal websites
  - I want to be more specific with my choices
- Which categories would you allow censored users to browse through your computer (assume that all categories use the same bandwidth)?
  - News pages (CNN, FoxNews, etc)

  - Social media (Facebook, Twitter, Instagram)
  - Search engines (Google, Bing)
  - Video sharing and streaming (YouTube, Vimeo, etc)
  - Scientific websites
- Users from which censored countries are you willing to help?
  - Any Country
  - China
  - Iran
  - Syria
  - Turkey
  - Saudi Arabia
- What is your age? (Optional)
  - 18-30
  - 30-40
  - Above 40
  - Prefer not to answer
- What is you gender? (Optional)
  - Male
  - Female
  - Prefer not to answer
- How woud you rate your computer proficiency? (Optional)
  - High
  - Medium
  - Low
  - Prefer not to answer
- Where do you live? (Optional)
  - USA
  - Europe
  - Asia
  - Other
  - Prefer not to answer