Metis: Data collection and reduction of overhead for censorship circumvention

Audrey Randall, Eric Wustrow

May 7, 2018

Abstract

Internet censorship is a swiftly growing phenomenon in many countries, with China, Iran, Syria, Russia, and many others drastically increasing their censorship programs in recent years. Existing censorship circumvention tools are slow and inconvenient to use, partly because of the lack of available data on which user requests will need to avoid being tampered with and which will not. Metis aims to provide an efficient algorithm to detect or predict which requests are subject to censorship, avoid censorship attempts when necessary, and aggregate global data on which websites get blocked. At the same time, Metis protects its users by implementing differential privacy techniques. By accurately predicting which requests do not require circumvention technology, Metis can reduce the bandwidth required to avoid censorship, improve the latency experienced by users of circumvention tools, and provide a means of collecting data on censorship across the globe.

1 Introduction

Internet censorship is defined as "the control or suppression of what can be accessed, published, or viewed on the Internet." [1] Censorship in the context used here refers primarily to that performed by national governments, whether through direct means or by influencing regulators and Internet service providers. It can be performed for moral, political, religious, economic, or other reasons. Government-sponsored censorship is not restricted to the denial of access to online resources - it can also take the form of surveilling people who attempt to access those resources, or tampering with these resources in more subtle ways, such as by slowing down or interrupting the connection or modifying the websites' contents. It is important to note that even the threat of surveillance can be sufficient to cause a change in user behavior: people who believe that they are being watched, and that there will be severe consequences for acting against the watcher's will, are much less likely to do so. The fear of being monitored exists whether the threat is consistently enforced or not.

Various watchdog organizations attempt to track and publish results on censorship. One such group is Freedom House, an organization founded in 1941

by Eleanor Roosevelt and Wendell Willkie, in order to "defend human rights and promote democratic change." [2] The group analyzes numerous factors in support of this goal, one of which is Internet freedom. They publish an annual report called "Freedom on the Net" which summarizes changes in censorship and surveillance online by country over the past year. The most recent reports have not been optimistic.

In 2016, "Freedom on the Net" reported that 67% of all Internet users lived in countries where criticism of the government, military, or ruling family was subject to censorship. During that year, authorities made arrests based on the contents of the arrestees' social media posts in 38 countries. Furthermore, 27% of all Internet users lived in countries where people have been arrested for publishing, sharing, or merely "liking" content on Facebook. [3]

2017 did not see an improvement. "Freedom on the Net" reported a seventh consecutive year of decline in Internet freedom around the world. More countries saw disruptions to mobile internet services, especially in places populated by religious and ethnic minorities. There was also an"increase in physical and technical attacks" on human rights defenders and journalists. [4]

"Freedom on the Net" also ranks countries based on their tendancy to abuse the online freedoms of their citizens. For the third year in a row, China topped the list. [4] For this reason among others, China's censorship system is the focus of the censorship circumvention and measurement system described in this paper. It is necessary at this point to briefly describe the events that led to China's censorship program, in order to understand why it is so extensive. Ever since the market-economy reforms of Deng Xiaoping in the late 1970s and 1980s, China has been walking an ideological tightrope. [5] While opening China to greater outside trade and foreign investments boosted the economy and raised the standard of living across the country, it also allowed ideas dangerous to the Chinese Communist Party's stability to enter. Deng Xiaoping is famously quoted as saying, "If you open a window for fresh air, you have to expect some flies to blow in." [6] When the Internet arrived a decade or so later, it was full of the "flies" he had been referring to. In order to keep track of how Chinese citizens were using the Internet, the Golden Shield project was started in 1998 to monitor China's domestic Internet. The Great Firewall of China followed shortly thereafter, in 2003, as the program for censoring unwanted foreign sites that is active in China today. Its purpose is to filter Internet requests based on both their destination (domain) and their content. [6] Contrary to expectations, the purpose of such censorship does not appear to be to to silence criticism of Chinese leaders or government policies. [7] Instead, content that would encourage people to act collectively is blocked, regardless of the purpose of the collective action. Social media posts that express anger or disgust with the Chinese government are normally left alone, while posts that would promote group action, even action in support of or unrelated to the government, are taken down. Censored posts are usually removed within 24 hours, although some may remain for up to five days. The purpose of the program seems to be, based on statements from current officials and the type of content blocked, to maintain the stability of the regime. [7]

Censorship based on the source rather than the contents of social media posts falls into a few broad categories. Social media that the Chinese government find difficult to monitor, such as end-to-end encrypted tools like WhatsApp, are blocked entirely, as are sites that allow downloading VPNs and Tor. Search engines that allow access to information the Chinese government does not wish discussed, such as Google, are also blocked outright. Additionally, websites that are objected to on moral or political grounds, such as sites that promote political ideologies that are incompatible with communism or pornography, are blocked as well. Interference of this type seems to be largely accomplished by deep packet inspection of all TCP packets that pass through state-controlled (or state-influenced) Internet Service Providers (ISPs). If a keyword that the censor has flagged to be blocked is found, a TCP reset packet is sent to both endpoints of the connection. [8] Censorship of this type is, for the most part, what current censorship circumvention systems are designed to address.

1.1 Existing Circumvention Tools

Tools such as Tor and Psiphon are already in widespread use in China. [9] [10] Psiphon uses SSH tunneling and VPN technology to ensure that a censor cannot read the true destination of a request, and therefore cannot decide to block it or not based on the domain of its destination. [10] Tor uses a technique called "onion routing." A packet gets encapsulated in several layers of tunnel, and then bounced around through several servers, called relays, before reaching its destination. Each layer of encryption gets stripped off one by one at each relay. This allows the relays to only know the identity of the relay that sent them the packet and the relay which is the next step on the path, which provides anonymity to the user. However, these tools do have flaws. For example, the Tor network is vulnerable to enumeration attacks. Tor must make the list of entry relays, called "guard relays," public. Guard relays must be highly trusted, since they have access to the identity of the client, so they are difficult to change quickly. Censors can successfully cut off Tor users by discovering all of the guard relays and blocking traffic destined for them. The Tor Project's solution to this problem was to invent "bridge nodes," which are essentially unlisted guard nodes, and distribute information about only three of them at a time, but China's censorship program has turned out to have sufficient resources to discover and block bridge relays as well. The result of these innovations is to create an arms race between censors and circumvention tools.

Another problem shared by most anti-censorship technology is that of speed. These circumvention systems are very slow. This is partly due to the fact that ALL traffic, regardless of whether or not it is destined to be censored, gets routed through these tools. An anti-censorship tool that could predict whether or not a request will need circumvention could provide a significant improvement in usability over existing technologies. Such a capability would require both detecting when requests are being tampered with, and an ability to predict which requests will be blocked based on previous behavior - essentially, a database of which sites are blocked in China. This is the purpose of Metis.

2 Metis

Metis aims to provide two improvements over existing circumvention software: first, to improve latency by predicting which requests will require avoiding censorship, and second, to collect a database of information about which websites are censored. This will serve to both improve the functionality of Metis itself and provide up-to-date information on the Chinese censorship program to researchers and the general public.

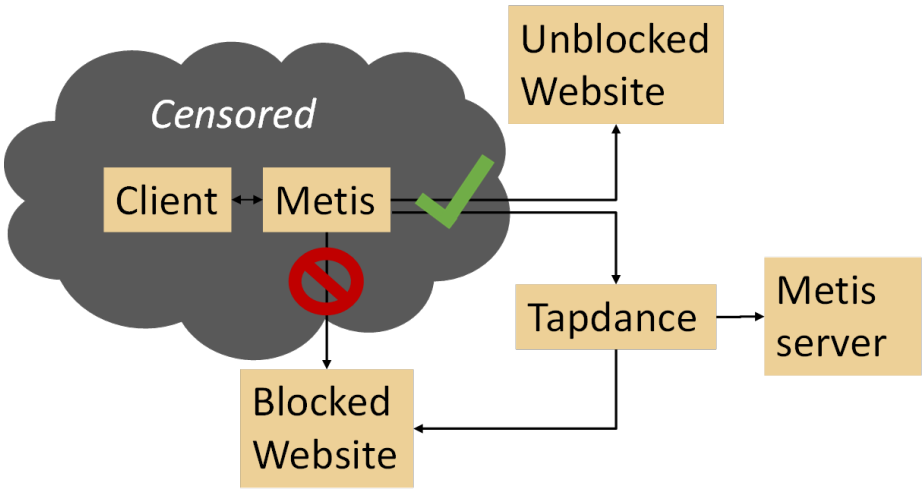

Figure 1: The purpose of Metis is to route requests sensibly based on how they can be most quickly and successfully completed.

Metis consists of a client and a server (Figure 1). The client sits as a proxy behind the user's browser, and decides how to route each request that the browser sends it. If the destination domain of the request is unknown to the client, Metis will attempt to make the request through regular means, by connecting directly. It will then monitor the connection for signs of censorship, such as:

- TCP reset packets (RSTs)
- Broken pipe errors
- Suspicious redirects
- Timeouts

If a symptom of censorship is detected, Metis will add the website to a "temporary" list of blocked websites, and reattempt to make the connection through a circumvention tool, as shown in Figure 2. At the moment, Metis uses the circumvention tool TapDance for this purpose. If an error is encountered

when the connection is made through TapDance as well as through normal channels, Metis assumes the error is not symptomatic of censorship, and removes the domain from the "temporary" blocked list. If an error is NOT encountered during the attempt made through TapDance, the website is removed from the "temporary" list of blocked domains, and added to the "permanent" list of blocked domains. This list is stored by each client and periodically sent to the Metis server, which aggregates the reports from all of the clients and assembles a "master blocked list," which then gets redistributed to the clients. The use of a "temporary" and "permanent" blocked list was adopted from a similar technique used by the Lantern censorship circumvention software, as was the list of symptoms of censorship. [11]

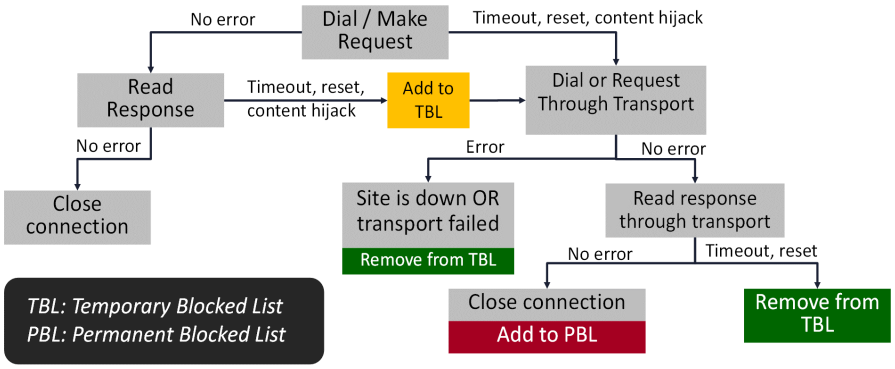

Figure 2: Metis's routing algorithm

2.1 TapDance

TapDance is currently the only circumvention technology that Metis utilizes. It was chosen primarily for its robustness against enumeration attacks. It utilizes end-to-middle proxying, which is a system where network operators in other countries than those implementing a censorship program agree to deploy software that allows censorship avoidance. Packets routed through these networks have a decoy destination, but are tagged with a steganographic request to be routed toward a different destination entirely. TapDance "stations" are deployed in these networks to detect the presence of tagged packets and route them toward their covert destinations. From a censor's point of view, it is impossible to distinguish TapDance packets from packets that are truly destined for the decoy sites, because the tag is encrypted with the public key of the station. Without the station's private key, it is impossible to distinguish this tag from the already-encrypted message it is appended to, so it is impossible to detect and block TapDance packets using deep packet inspection. Additionally, any site that would be reached by going through a participating ISP can serve as a decoy site, so blocking all possible decoy sites would create untenable collateral

damage if enough ISPs were to participate in hosting TapDance stations. [12] While Tapdance is not yet widely deployed, it is to be hoped that in the future, it will prove to be a remarkably robust circumvention tool. For the moment, it certainly has sufficient capacity to be used to test the efficacy of Metis.

2.2 Differential Privacy and Google's RAPPOR

Sending lists of domains that Metis clients have deemed to be censored to the Metis server poses a potential privacy risk to users. It would be unwise to do so in a format that reveals to the server the identity of the client as well as the exact websites that client has accessed. To alleviate this risk, Metis implements a form of differential privacy to encode client reports. In its simplest form, differential privacy allows the collection of aggregated, noisy data, without allowing the collector to know with certainty the exact value of any particular response. The classic example of differential privacy is the Communist Party thought experiment (which has no relation to the Chinese Communist Party). Say a surveyor wishes to ask how many people in a country are members of the Communist Party, but their respondents are unwilling to tell the truth for fear of repercussions. The surveyor may then instruct the respondents to flip a coin before they answer the question and conceal the outcome of the coin flip. If the coin came up "heads," the respondent should answer truthfully. If the coin came up "tails," the respondent should answer in the affirmative, no matter what the truthful answer is. In this way, the surveyor would be unable to prove even if they wanted to that any particular respondent was a member of the Communist Party. However, the surveyor would also be able to calculate, using basic statistics, what percentage of the respondents were truly members of the party to some degree of certainty.

The classic Communist Party problem deals only with survey answers that have a binary form. In order to report arbitrary strings such as website domains, a much more sophisticated differential privacy method must be used. Metis implements an algorithm developed by Google called RAPPOR, or "Randomized Aggregatable Privacy-Preserving Ordinal Response." [13] RAPPOR operates by encoding each string into its own Bloom filter, then adding noise to the filter in a probabilistic fashion. One noisy filter becomes one report. Since the server knows how each client generates its noise, it can estimate which bits were set before the noise was added, but only for a sufficiently large number of reports on the same string. In order to decode the reports, a set of strings to search for must be known in advance, and the number of strings that can be detected is reliant on the number of reports that the server has received. This limits the Metis server to searching for only the top n most popular websites, where n is determined by the number of reports available for the server to analyze. The implication of this limitation is that the master blocked list stored by the server will become more accurate over time, but will never be able to detect a certain amount of infrequently accessed websites, since these will never be reported on frequently enough to create a detectable signal in the randomized reports.

In more detail, RAPPOR guarantees both differential privacy and protection

against longitudinal attacks. It does so by adding noise in two steps instead of one. After each domain gets put into a Bloom filter, which will not change over the lifetime of the client, the first layer of noise is added to create the Permanent Randomized Response (PRR). This response is memoized and does not change — each domain has a PRR that is set at the time the domain is first encountered. Let B be the Bloom filter of size k associated with a particular domain, where B_i is the i$^{\text{th}}$ bit in the filter, $0 < i < k$. The PRR for that domain is then generated with the following formula:

$$\text{PRR}_i = \begin{cases} 0 & \text{with probability } \frac{1}{2}f \\ 1 & \text{with probability } \frac{1}{2}f \\ B_i, & \text{with probability } 1 - f \end{cases} \tag{1}$$

where f is a tunable parameter. The PRR is never sent across the wire. Once the PRR has been generated and stored, it is used and re-used to generate the Instantaneous Randomized Response (IRR), which changes for each instance of the report. This system guarantees that even if an attacker could aggregate and decode an infinite number of reports, the only value they would learn would be that of the PRR, which does not allow them to discover the value of the original Bloom filter with any certainty. This avoids loss of user privacy, and guarantees to the user that the server cannot ever detect which websites they specifically visited even if it should attempt to do so. However, with enough user reports, the server will be able to detect which websites are consistently visited and consistently considered blocked.

Assume that the IRR is a bit array of size k, where IRR_i is the i$^{\text{th}}$ bit in the array, $0 < i < k$. The IRR is generated with the following formula:

$$P(\text{IRR}_i = 1) = \begin{cases} q, & \text{if PRR}_i = 1 \\ p, & \text{if PRR}_i = 0 \end{cases} \tag{2}$$

where q and p are tunable parameters.

The implementation of RAPPOR published in Erlingsson et al. uses the parameter values $f = 0.5$, $q = 0.75$, and $p = 0.5$. We have left the values of p and q unchanged, but reduced the value of f to 0.2, in order to achieve greater detection accuracy on infrequently reported domains. The generation of the PRR and IRR may be visualized in Figure 3, which was taken from Erlingsson et al. Note that the PRR is called B' in that figure, and the IRR is labeled "Report sent to server."

A further note to make on the RAPPOR algorithm is that collisions between strings in bits of the Bloom filter are problematic. That is, if two domains hash to the same bit in the Bloom filter, the chances of identifying either domain after the noise has been added is much lower. To solve this problem, RAPPOR introduces "cohorts," or groups of reporters that use different sets of hash functions to generate their Bloom filters. Erlingsson et al. warns that if the number of cohorts m is too high, insufficient signal is produced to reconstruct the Bloom filters at the server. Similarly, if m is too low, the risk of collisions

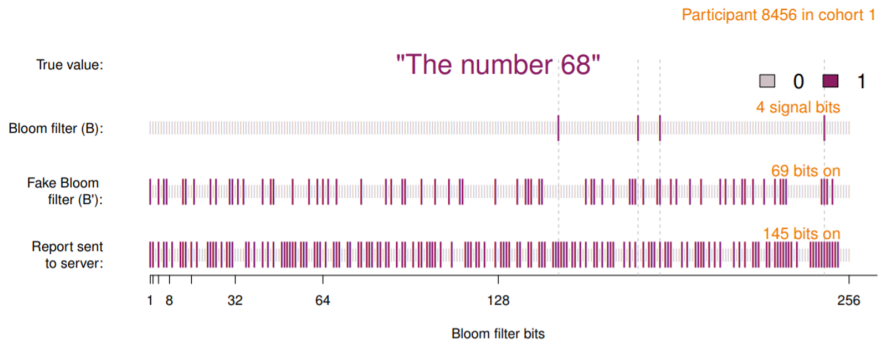

Figure 3: The generation of the PRR and IRR from the Bloom filter.

is not sufficiently reduced. Erlingsson et al. used a value of $m = 64$, and Metis left that number unchanged.

Decoding the IRRs once a sufficient number have arrived at the server is done by the following process:

1. First, given the collected IRRs, estimate how many times each bit was set in each original Bloom filter, before the noise was added. This will allow us to estimate the distribution of the original set bits, which in turn will allow us to predict which domains caused those bits to be set. For example, if bits 2 and 4 get set in the Bloom filter for "xyzw.com,", and we estimate that bits 2 and 4 were set far more often than other bits in the reports we have collected, we may predict that "xyzw.com" was reported frequently.

   Let t_{ij} be the number of times each bit i within cohort j was truly set. The estimate of original bit distribution is given by the following formula:

   $$t_{ij} = \frac{c_{ij} - (p + \frac{1}{2}fq - \frac{1}{2}fp)N_j}{(1-f)(q-p)}$$

   Place all t_{ij}'s into a vector called Y. This will become the response vector when LASSO regression is used to detect the domains that were reported.

2. Create the design matrix, X. This shall be a matrix of size $(km*M)$, where M is the number of domains that will be searched for. Each column corresponds to a specific domain. For every domain in the search space, determine what the bits of its Bloom filter would be for every cohort. Then, let each bit in the Bloom filter be an entry in X, such that each column is a stack of the Bloom filters for each cohort for the corresponding domain.

3. Use LASSO regression to determine which domains were strongly correlated with the estimated distribution of set Bloom filter bits. In order to

determine a reasonable value for the LASSO regularization parameter, λ, we used cross validation and a grid search. [14] [15] [16]

4. Confirm the domains selected by the LASSO regression were truly reported by performing a regular least-squares regression using the selected domains to estimate counts of set bits, their standard errors and their p-values.

5. Compare the p-values to a Bonferroni-corrected confidence level of $0.5/M$ to determine which reported frequencies are statistically different from 0.

We ran into a series of interesting challenges while implementing RAPPOR. First, the choice of p, q, and f is somewhat arbitrary. It is possible to find a value of ϵ, as described by Cynthia Dwork's paper on the original definition of differential privacy, for given values of the parameters. [17] However, the definition of a "reasonable" value of ϵ is still under debate, and indeed is an ongoing area of research. Values in the literature range from 0.01 to 10. [13] Given that we only chose to vary f, it was easy to record the effects of that variation on the LASSO regression relationship we generated to select which domains were probably reported. As expected, since the greater the value of f the more noise was added to the PRR, our ability to accurately detect which domains were reported for a given number of reports fell of steeply as f rose.

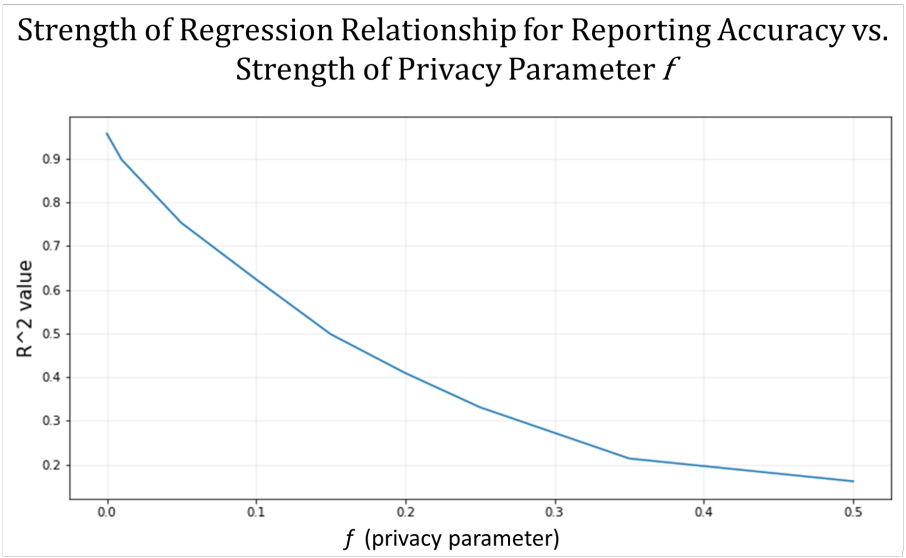

Figure 4: By modifying the privacy parameter f, it is possible to decrease the decoding time and increase the decoding accuracy.

We also ran into challenges when performing the LASSO regression itself. The accuracy of this method as a means of variable selection depends heavily on the choice of the regularization parameter, λ. (The scikit-learn library which

we utilized to perform this function calls this parameter α, since the keyword "lambda" is already reserved in Python.) The purpose of λ is to penalize the size of the coefficients computed by the regression, in order to decrease overfitting. This effect is achieved by minimizing the usual loss function for linear regression subject to the constraint that the product of λ and the L1-norm of the coefficients are added to it:

$$\min_{\beta \in \mathbb{R}^p} \left\{ \frac{1}{N} \|y - X\beta\|_2^2 + \lambda \|\beta\|_1 \right\} \tag{3}$$

One effect of the regularization is to set coefficients that fall below a certain threshold value to zero. The reasoning is that since these coefficients were so small as to be negligible in any case, they might as well be removed from the calculation. In this way, LASSO regression can be used as a method of variable selection. The value of λ must be chosen carefully, however. An overly large value of λ causes too many coefficients to be set to zero, and an overly small value causes too few to be set to zero. Neither condition is desirable.

The most common (and most reliable) way to set λ is to search through several possible values and perform k-fold cross validation (where k is a separate variable from k, the number of Bloom filter bits) on the dataset with each of them, to see which produces the lowest value of the loss function. This approach is quite time-consuming, having a temporal complexity on the order of $\mathrm{O}(M^2 \mathrm{k} n g)$, where n is the number of reports, M is the number of domains to be searched for, k is the number of folds in the cross validation, and g is the number of guesses to be made for the value of λ. We attempted to estimate λ by developing a simpler heuristic, without success. The optimal value of λ for a particular distribution did appear to scale linearly with the number of reports analyzed, but we could not prove that this relationship held true for other distributions. Since the distribution of reports will be unknown in the wild, this approach was abandoned. Instead, k-fold cross validation with a low k and few guesses g was implemented. This strategy still presents a challenge for large values of M, but appears to be the best possible option.

3 Measuring Classification Accuracy

In order to evaluate how accurate Metis is at determining which domains are censored and which are not, it was necessary to establish some sort of ground truth to compare to Metis's classifications. For this purpose, we used ooniprobe, a censorship measurement tool developed by the Tor Project. [18]

3.1 Ooniprobe

OONI stands for the Open Observatory of Network Interference. Its purpose is to develop and deploy its software, which is called ooniprobe, to allow Internet users to contribute to a global database of knowledge about censorship.

Ooniprobe has the capability to test the accessibility of websites, instant messaging apps such as WhatsApp, and circumvention tools such as Tor. It can also test for the presence of middleboxes in the network that might be performing censorship or surveillance, but these tests are not a reliable indication of nefarious intent, since there are a great many uses for a middlebox that do not involve censorship.

It is not currently possible to integrate ooniprobe directly into a real-time circumvention system such as Metis, since the tests are much more time intensive than they would need to be to provide an improvement in speed over existing circumvention tools.

The ooniprobe test that Metis compares its classification accuracy to is the web connectivity test. For every website, the test makes two connections: one from the computer of the user and another from a control server in a location known to be free of tampering. Ooniprobe then compares the responses to the DNS lookup, TCP connection, and HTTP GET request for the website from both locations. If the results are significantly different between the user and control connections, ooniprobe records that the site is censored, and what symptoms of censorship it showed. Symptoms of tampering that ooniprobe looks for include:

- Differences in DNS responses, or use of an untrustworthy DNS resolver, that might indicate DNS cache poisoning
- Failure to establish a TCP connection
- Failure to receive an HTTP response over the user's network, when the response over the control network was received
- Differences in HTTP status codes
- ALL of the following: A difference of more than a certain percentage in the lengths of the HTTP response bodies, differences in the HTTP header names, AND differences in the HTML title tags.

Metis is capable of detecting TCP failures, some DNS poisoning (provided it causes an error rather than acting like a redirect), resets, and timeouts. Ooniprobe can detect DNS poisoning that acts like a redirect and does not cause an error, as well as HTTP errors, because of its use of a control network. Metis is unable to access such a network without losing the latency improvement that is its primary goal. A possible future solution is to partner with OONI in order to integrate the ooniprobe database directly into Metis's own measurements, to catch sites censored via DNS poisoning or HTTP injection. Sites censored in this manner make up 1.1% of the websites Metis measured.

3.2 Comparison between ooniprobe and Metis

We tested connectivity to the top 1000 most popular websites, as defined by Alexa. [19]

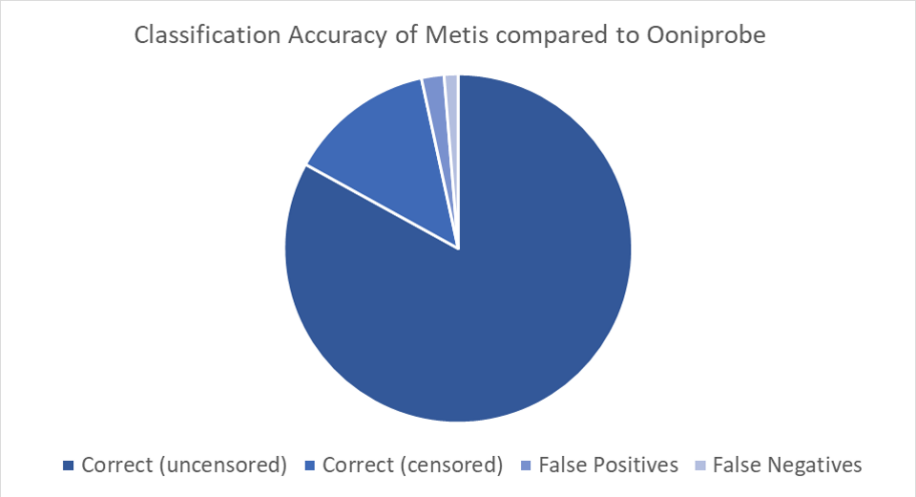

Figure 5: Metis' classification accuracy as compared to ooniprobe's measurements.

Metis achieves 97% accuracy as compared to ooniprobe (Figure 5). 851 websites were classified as uncensored by ooniprobe, and 149 were classified as censored. We split the 3% of sites that remain into two categories, false positives (websites that Metis classified as censored that ooniprobe could access without trouble) and false negatives (websites Metis didn't detect tampering on that ooniprobe thought were censored).

False Negatives				False Positives	
Unknown Reason	Server Error (in US & CN)	DNS Tampering	HTTP Tampering	Misinterpreted Misc. Error	Misinterpreted Timeouts
14%	7%	36%	43%	5%	95%

Figure 6: Classification errors for Metis as compared to ooniprobe. False positives are sites Metis classified as censored when ooniprobe did not, and false negatives are websites Metis classified as unblocked that had tampering detected by ooniprobe.

False negatives were divided into four categories, and false positives into two (Figure 6). Of the false negatives, the "unknown reason" category in Figure 6 were the most puzzling. These websites exhibit unique and often confusing behavior. For example, this category included the home page for WhatsApp, which is known to be censored in China. However, when an attempt to connect to whatsapp.com was made through Metis, the first request appeared to load a partial response. None of the sub-requests loaded, and subsequent reattempts

to connect to the same domain were met with the expected TCP resets. The response that was received initially consisted of the text "What country are you in?" followed by a list of countries, which is not normally part of the whatsapp.com response. Given such strange behavior, further investigation is clearly required to determine the cause of the misclassification. However, since the behavior has proved to be non-reproducible, an explanation is not yet available.

The remainder of the false negatives are straightforward. Some websites (which fall into the second category in Figure 6) exhibited different errors over the control network, which was in the US, than they did in China. Metis treats these errors as being likely to be the fault of the server, not the censor, but ooniprobe classifies them as censored. The last two categories, DNS poisoning and HTTP injection, were populated by websites that showed symptoms of censorship Metis is not equipped to detect.

The false positives were split into two categories. The first, "misinterpreted errors," consisted of errors that ooniprobe chose not to classify as evidence of tampering (for example, a "too many redirects" error). The remainder were all due to overly conservative timeout values. Some sites loaded so slowly that Metis assumed they were censored when they were not. Further data should be gathered in the future to tune the timeout values to appropriate durations, to eliminate this source of error.

4 Measuring Performance

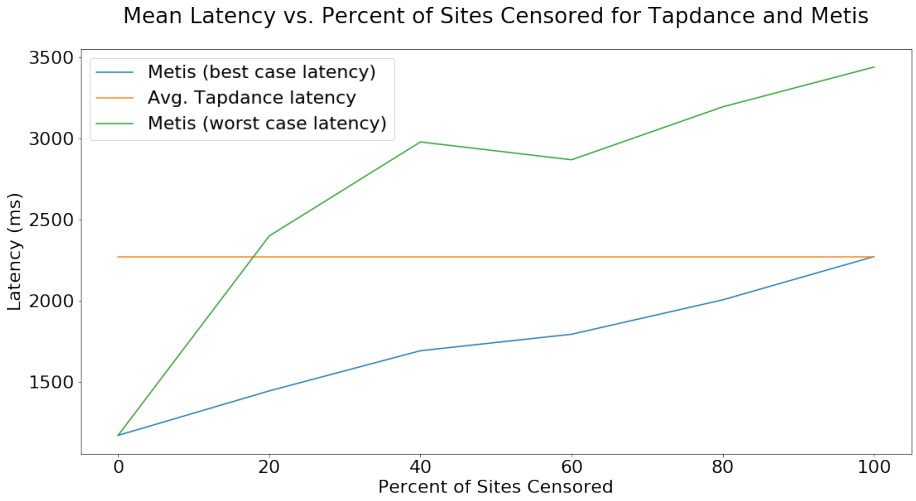

Figure 7: Latencies as a function of percentage of sites censored for Metis and Tapdance. Metis' best case latencies are when the clients have a 100% accurate list of blocked sites already. The worst case scenario is when Metis has no pre-existing list of blocked sites at all, and must test every site the user tries.

Since Tapdance is currently the only transport that Metis uses to route requests that need circumvention technology, Metis's latency comparisons were to Tapdance's average latency. Metis did succeed in providing an improvement in speed over Tapdance in all cases that were measured. Figure 7 shows Metis's best case and worst case average download speeds. The best case scenario is defined to be the case where Metis has a blocked list that includes every censored site that the user attempts to access, meaning that it does not need to test any new sites to classify them correctly. The worst case scenario is when Metis has no list of blocked sites at all, and must test ALL websites the user wishes to access to determine how they should be routed. The average use case is likely to be much closer to the best case scenario than the worst case scenario, because even if access to the server was lost, each Metis client would still have its own, personal list of blocked sites. The worst case scenario will only exist in practice for the duration of time required for the server to begin to build a comprehensive list. Even in the case of the client losing connectivity to the server for some period of time, the clients will retain their personal lists of blocked sites.

In Figure 7, it is possible to see that Metis is faster than or equal in speed to Tapdance for any given ratio of censored to uncensored data in the best case scenario. In other words, if 100% of websites are censored, routing requests through Metis (which then routes them through Tapdance) is no less efficient than routing them through Tapdance alone. In the worst case scenario, the extra time required to test websites for signs of tampering exceeds the benefits of routing some requests directly after approximately 18% of requested websites are censored. In the tests performed on the Alexa top 1,000 websites, only 14.9% were censored in China. It is possible that that number is not representative of the general population of websites, and the true percentage of websites censored in China is higher, but if the error is less than 3%, Metis will still provide an improvement over Tapdance in all situations. Even if the error should prove greater than that, since Metis's speed will improve the more measurements it takes, it is still safe to say that Metis accomplishes its goal of latency reduction in the vast majority of circumstances.

Interestingly, we did not observe a reliable pattern when we graphed the latency distributions. Consider Figure 8. It would have been reasonable to expect the majority of measured latencies to fall near two points: the first being at the average latency for an uncensored packet, and the second being at the average latency for a packet using Tapdance. If that had been the case, the graph of latencies for 100% and 0% of sites censored should have had one peak, and the remainder of the graph should have had two peaks per line. However, this pattern was not observed - there does not appear to be a relationship between the distributions of latencies and the percentage of sites censored. It is probable that since only ten attempts were averaged for every website connected to, and only 100 websites comprised the dataset used to arrive at the average latencies, there were simply insufficient data to plot the smooth curves we expected to see.

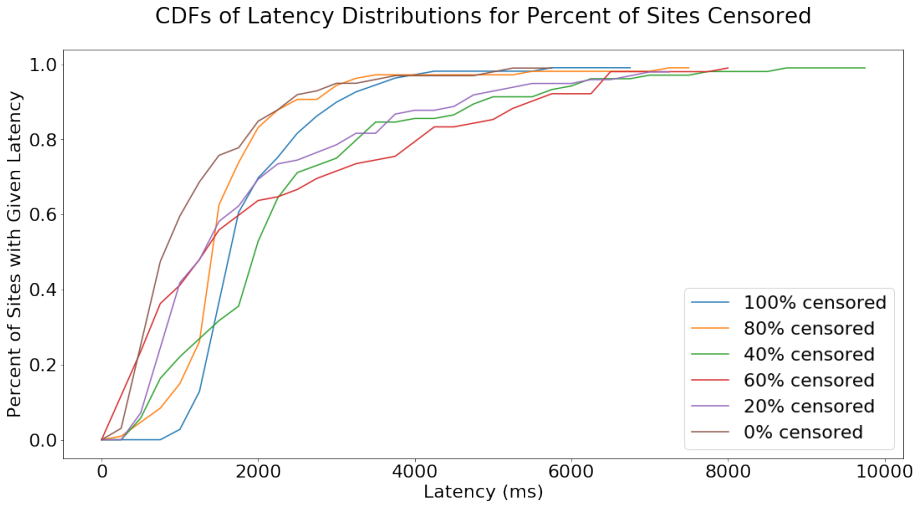

Figure 8: CDFs of latencies for percentage of sites censored

5 Future Work

5.1 Pluggable Transports

The original plan for Metis was not to use only one censorship circumvention tool. Such tools are vulnerable to being blocked, compromised, or overloaded, so it would be useful to be able to swap tools as needed to provide the best user experience. Fortunately, the Tor Project is in the process of designing a specification to make this possible - the pluggable transport specification. The first version of the this specification is already in existence. [20] Pluggable transports are technologies that disguise traffic in a variety of ways, including by scrambling it (obfsproxy), disguising it as another type of traffic (SkypeMorph), or sending it to a decoy destination by using domain fronting (meek) [21]. Tapdance uses refraction networking, which is similar to the domain fronting algorithm employed by meek, so we are confident that Tapdance could be a pluggable transport in the future. At the moment, the Tor pluggable transports are only made to be used by Tor, and it isn't straightforward to use them as part of any other software. However, this may change in the near future, since an update to the PT specification may be in the works. Once pluggable transports are a little more pluggable, it would be advantageous for Metis to be able to swap between several of them.

5.2 Improving Classification

There is a much higher cost associated with a misclassification of a domain as uncensored when it should be censored, than there is for a misclassification of an uncensored site as blocked. A mistake of the former variety leads to the

user trusting the site when they may be being monitored or deceived, while a mistake of the latter type simply leads to a slightly longer site load time than necessary. However, Metis will never be capable of classifying certain symptoms of censorship, such as HTTP and DNS tampering, without routing all requests over a control network and comparing the results. It would be beneficial to partner with the Tor Project in order to integrate the ooniprobe database with Metis's database, to catch the censored sites that Metis cannot detect by itself. Additionally, some of Metis's false negatives were due to its interpretation of certain errors as benign, whereas ooniprobe interpreted them as symptoms of tampering. A future project for Metis will be to compile a better list of errors that are evidence of tampering, by inspecting the responses to websites that Metis and ooniprobe classified differently in more detail.

6 Conclusion

There were two hypotheses laid out for Metis: first, that it was possible to provide a speed improvement over existing anti-censorship tools by routing requests based on their need to avoid tampering, and second, that it would be possible to collect data on censorship in such a way that Metis did not lose its speed advantage. Metis was able to accomplish both of these goals. For tests conducted on the Alexa top 1,000 websites [19], Metis was able to classify sites as censored or uncensored with 97% accuracy. Additionally, Metis provided a significant speed advantage over Tapdance when tested against the top 100 websites, where both Metis and Tapdance accessed each website ten times and then recorded their average latencies. Metis also provides differential privacy to protect its users in the form of Google's RAPPOR algorithm. In the future, it is to be hoped that Metis can provide a significant improvement to the safety and efficiency of its users' Internet browsing experiences.

References

[1] E. E. Schmidt and J. Cohen, "The future of internet freedom," *The New York Times*, 03 2014.

[2] "Freedom house," 2018.

[3] "Freedom on the net report," 2016.

[4] "Freedom on the net report," 2017.

[5] E. of the Encyclopedia Brittanica, "Deng xiaoping, chinese leader," *Encyclopedia Brittanica*.

[6] "Torfox: A stanford project," 2011.

[7] G. KING, J. PAN, and M. E. ROBERTS, "How censorship in china allows government criticism but silences collective expression," *American Political Science Review*, vol. 107, no. 2, p. 326–343, 2013.

[8] M. S. Clayton R. and W. R.N.M., "Ignoring the great firewall of china," *Danezis G., Golle P. (eds) Privacy Enhancing Technologies*, vol. 4258, 2006.

[9] "Tor metrics: Users," 2018.

[10] "Psiphon," 2018.

[11] "Lantern: Faster than a vpn," 2018.

[12] E. Wustrow, C. M. Swanson, and J. A. Halderman, "Tapdance: End-to-middle anticensorship without flow blocking," *Proceedings of the 23rd USENIX Security Symposium*, 2014.

[13] U. Erlingsson, V. Pihur, and A. Korolova, "Rappor: Randomized aggregatable privacy-preserving ordinal response," *Proceedings of the 21st ACM Conference on Computer and Communications Security*, 2014.

[14] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 58, no. 1, pp. 267–288, 1996.

[15] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[16] "1.1.3 lasso," 2018.

[17] C. Dwork, "Differential privacy: A survey of results," in *Theory and Applications of Models of Computation* (M. Agrawal, D. Du, Z. Duan, and A. Li, eds.), (Berlin, Heidelberg), pp. 1–19, Springer Berlin Heidelberg, 2008.

[18] "Ooniprobe," 2018.

[19] "Alexa top sites," 2018.

[20] "Tor index:torspec," 2018.

[21] "Tor pluggable transports," 2018.