Five ways to test bridge reachability

Roger Dingledine
arma@torproject.org

Tor Tech Report 2011-12-001
December 1, 2011

1 Introduction

Once we get more (and more diverse) bridge addresses [2], the next research step is that we'll need to get better at telling which bridges are blocked in which jurisdictions. For example, most of the bridges we give out via https and gmail are blocked in China. But which ones exactly? How quickly do they get blocked? Do some last longer than others? Do they ever get unblocked? Is there some pattern to the blocking, either by time, by IP address or network, by user load on the bridge, or by distribution strategy? We can't evaluate new bridge distribution strategies[1] if we can't track whether the bridges in each strategy are being blocked.

Generally speaking, bridge reachability tests break down into two approaches: passive and active. Passive tests don't involve any new connections on the part of Tor clients or bridges, whereas active tests follow the more traditional "scanning" idea. None of the reachability tests we've thought of are perfect. Instead, here we discuss how to combine imperfect tests and use feedback from the tests to balance their strengths and weaknesses.

2 Passive approaches

We should explore two types of passive testing approaches: reporting from bridges and reporting from clients.

2.1 Reporting from bridges

Right now Tor relays and bridges publish aggregate user counts [7]—rough number of users per country per day. In theory we can look at the user counts over time to detect statistical drops in usage for a given country. That approach has produced useful results in practice for overall connections to the public Tor relays from each country: see George Danezis's initial work [1] on a Tor censorship detector.

[1] https://blog.torproject.org/blog/bridge-distribution-strategies

But there are two stumbling blocks when trying to apply the censorship detector model to individual bridges. First, we don't have ground truth about which bridges were *actually* blocked or not at a given time, so we have no way to validate our models. Second, while overall usage of bridges in a given country might be high, the load *on a given bridge* tends to be quite low, which in turns makes it difficult to achieve statistical significance [6] when looking at usage drops.

Ground truth needs to be learned through active tests: we train the models with usage patterns for bridges that get blocked and bridges that don't get blocked, and the model predictions should improve. The question of statistical significance can be overcome by treating the prediction as a hint: even if our models don't give us enough confidence to answer "blocked for sure" or "not blocked for sure" about a given bridge, they should be able to give us a number reflecting likelihood that the bridge is now blocked. That number should feed back into the active tests, for example so we pay more attention to bridges that are more likely to be newly blocked.

2.2 Reporting from clients

In addition to the usage reporting by bridges, we should also consider reachability reporting by clients. Imagine a Tor client that has ten bridges configured. It tries to connect to each of them, and finds that two work and eight don't. This client is doing our scanning for us, if only we could safely learn about its results. The first issue that comes up is that it could mistakenly report that a bridge is blocked if that bridge is instead simply down. So we would want to compare the reports to concurrent active scans from a "more free" jurisdiction, to pick out the bridges that are up in one place yet down in another.

From there, the questions get trickier: 1) does the set of bridges that a given user reports about create a fingerprint that lets us recognize that user later? Even if the user reports about each bridge through a separate Tor circuit, we'd like to know the time of the scan, and nearby times can be used to build a statistical profile. 2) What if users submit intentionally misleading reports? It seems there's a tension between wanting to build a profile for the user (to increase our confidence in the validity of her reports) versus wanting to make sure the user doesn't develop any recognizable profile. Perhaps the Nymble [4] design family can contribute an "unlinkable reputation" trick to resolve the conflict, but as we find ourselves saying so often at Tor, more research remains.

3 Active approaches

The goal of active scanning is to get ground truth on whether each bridge is really blocked. There's a tradeoff here: frequent scans give us better resolution and increased confidence, but too many scan attempts draw attention to the scanners and thus to the addresses being scanned.

We should use the results of the passive and indirect scans to give hints about what addresses to do active scans on. In the steady-state, we should aim to limit our active scans to bridges that we think just went from unblocked to blocked or vice versa, and to a sample of others for spot checks to keep our models trained.

There are three pieces to active scanning: direct scans, reverse scans, and indirect scans.

3.1 Direct scans

Direct scans are what we traditionally think of when we think of scanning: get access to a computer in the target country, give it a list of bridges, and have it connect directly to each bridge on the list.

Before I continue though, I should take an aside to discuss types of blocking. In September 2009 when China first blocked some bridges, I spent a while probing the blocked bridges from a computer in Beijing. From what I could tell, China blocked the bridges in two ways. If the bridge had no other interesting services running (like a webserver), they just blackholed the IP address, meaning no packets to or from the IP address made it through the firewall. But if there *was* an interesting service, they blocked the bridge by IP and port. (I could imagine this more fine-grained blocking was done by dropping SYN packets, or by sending TCP RST packets; but I didn't get that far in my investigation.)

So there are two lessons to be learned here. First, the degree to which our active scans match real Tor client behavior could influence the accuracy of the scans. Second, some real-world adversaries are putting considerable effort—probably manual effort—into examining the bridges they find and choosing how best to filter them. After all, if they just blindly filtered IP addresses we list as bridges, we could add Baidu's address as a bridge and make them look foolish. (We tried that; it didn't work.)

These lessons leave us with two design choices to consider.

First, how much of the Tor protocol should we use when doing the scans? The spectrum ranges from a simple TCP scan (or even just a SYN scan), to a vanilla SSL handshake, to driving a real Tor client that does a genuine Tor handshake. The less realistic the handshake, the more risk that we conclude the bridge is reachable when in fact it isn't; but the more realistic the handshake, the more we stand out to an adversary watching for Tor-like traffic.

The mechanism by which the adversary is discovering bridges [3] also impacts which reachability tests are a smart idea. For example, it appears that China may have recently started doing deep packet inspection (DPI) for Tor-like connections over the Great Firewall and then doing active-followup SSL handshakes to confirm which addresses are Tor bridges. If that report turns out to be true, testing bridge reachability via active scans that include handshaking would be counterproductive: the act of doing the test would influence the answer. We can solve their attack by changing our handshake[2] so they don't recognize it anymore, or by introducing scanning-resistance[3] measures like bridge passwords. In the shorter-term, we should confirm that simple connection scanning (without a handshake) doesn't trigger any blocks, and then restrict ourselves to that type of scanning in China until we've deployed a better answer.

Second, should we scan "decoy" addresses as well, to fool an observer into thinking that we're not scanning for Tor bridges in particular, and/or to drive up the work the observer needs to do to distinguish the "real" bridges? Whether this trick is useful depends on the level of sophistication and dedication of the adversary. For example, China has already demonstrated that they check IP addresses before blocking them, and in general I worry that the more

[2] https://blog.torproject.org/blog/iran-blocks-tor-tor-releases-same-day-fix
[3] https://svn.torproject.org/svn/projects/design-paper/blocking.html#tth_sEc9.3

connections you make to *anything*, the more likely you are to attract attention for further scrutiny. How would we generate the list of decoy addresses? If we choose it randomly from the space of IP addresses, a) most of them will not respond, and b) we'll invite abuse complaints from security people looking for worms. Driving up the work factor sounds like a great feature, but it could have the side effect that it encourages the adversary to invest in an automated "is this a Tor bridge" checker, which would be an unfortunate step for them to take if they otherwise wouldn't.

Active direct scans come with a fundamental dilemma: the more we think a bridge has been blocked, the more we want to scan it; but the more likely it is to be blocked, the more the adversary might already be watching for connections to it, for example to do a "zig-zag" bridge enumeration attack [3]. So we need to avoid scanning bridges that we think are not blocked. But we also need to explore more subtle scanning techniques such as the ones below.

3.2 Reverse scans

A bridge that gets duplex[4] blackholed by a government firewall can learn that it has been filtered by trying to make a connection *into* the filtered country.

For example, each bridge might automatically connect to baidu.com periodically, and publish the results of its reachability test in its extrainfo descriptor. We could either feed this information into the models and follow up with other active probes if the bridge thinks it's been blocked; or we could use it the other way by instructing bridges that we think have been blocked to launch a reverse scan.

We can actually take advantage of the flexibility of the Tor protocol to do scanning from each bridge to Baidu without changing the bridge code at all: we simply try to extend an ordinary circuit from the bridge to the target destination, and learn at what stage the 'extend' request failed. (We should extend the Tor control protocol[5] to expose the type of failure to the Tor controller, but that's a simple matter of programming.)

Note that these reverse scans can tell us that a bridge has been blocked, but it can't tell us that a bridge *hasn't* been blocked, since it could just be blocked in a more fine-grained way.

Finally, how noticeable would these reverse scans be? That is, could the government firewall enumerate bridges by keeping an eye out for Tor-like connection attempts to Baidu? While teaching the bridges to do the scanning themselves would require more work, it would give us more control over how much the scans stick out.

3.3 Indirect scans

Indirect scans use other services as reflectors. For example, you can connect to an FTP server inside the target country, tell it that your address is the address of the bridge you want to scan, and then try to fetch a file. How it fails should tell you whether that FTP server could reach the bridge or not.

There are many other potential reflector protocols out there, each with their own tradeoffs. For example, can we instruct a DNS request in-country to recurse to the target bridge address,

[4] http://en.wikipedia.org/wiki/Duplex_%28telecommunications%29

[5] https://trac.torproject.org/projects/tor/ticket/2576

and distinguish between "I couldn't reach that DNS server" and "that wasn't a DNS server"? (DNS is probably not the right protocol to use inside China, given the amount of DNS mucking they are already known to do.)

Another avenue is a variant on idle scanning,[6] which takes advantage of predictable TCP IPID patterns: send a packet directly to the bridge to learn its current IPID, then instruct some computer in-country to send a packet, and then send another packet directly to find the new IPID and learn whether or not the in-country packet arrived.

What other services can be bent to our will? Can we advertise the bridge address on a bittorrent tracker that's popular in China and see whether anybody connects? Much creative research remains here.

4 Putting it all together

One of the puzzle pieces holding us back from rolling out the "tens of thousands of bridge addresses offered by volunteers with spare net blocks" plan is that we need better ways to get feedback on when addresses get blocked. The ideas in this blog post hopefully provide a good framework for thinking about the problem.

For the short term, we should deploy a basic TCP connection scanner from inside several censoring countries (China, Iran, and Syria come to mind). Since the "clients report" passive strategy still has some open research questions, we should get all our hints from the "bridges report" passive strategy. As we're ramping up, and especially since our current bridges are either not blocked at all (outside China), or mostly blocked (inside China), we should feel free to do more thorough active scans to get a better intuition about what scanning can teach us.

In the long term, I want to use these various building blocks in a feedback loop to identify and reward successful bridge distribution strategies, as outlined in Levchenko and McCoy's FC 2011 paper [5].

Specifically, we need these four building blocks:

1. A way to discover how much use a bridge is seeing from a given country. Done: see the WECSR10 paper [7] and usage graphs[7].

2. A way to get fresh bridge addresses over time. The more addresses we can churn through, the more aggressive we can be in experimenting with novel distribution approaches. See the "more bridge addresses" report [2] for directions here.

3. A way to discover when a bridge is blocked in a given country. That's what this report is about.

4. Distribution strategies that rely on different mechanisms to make enumeration difficult. Beyond our "https" and "gmail" distribution strategies, we know a variety of people in censored countries, and we can think of each of these people as a distribution channel.

[6] http://nmap.org/book/idlescan.html
[7] https://metrics.torproject.org/users.html#bridge-users

We can define the *efficiency* of a bridge address in terms of how many people use it and how long before it gets blocked. So a bridge that gets blocked very quickly scores a low efficiency, a bridge that doesn't get blocked but doesn't see much use scores a medium efficiency, and a popular bridge that doesn't get blocked scores high. We can characterize the efficiency of a distribution channel as a function of the efficiency of the bridges it distributes. The key insight is that we then adapt how many new bridges we give to each distribution channel based on its efficiency. So channels that are working well automatically get more addresses to give out, and channels that aren't working well automatically end up with fewer addresses.

Of course, there's more to it than that. For example, we need to consider how to handle the presence of bridge enumeration attacks that work independently of which distribution channel a given bridge address was given to. We also need to consider attacks that artificially inflate the efficiency of bridges (and thus make us overreward distribution channels), or that learn about a bridge but choose not to block it. But that, as they say, is a story for another time.

References

[1] George Danezis. An anomaly-based censorship-detection system for Tor. Technical Report 2011-09-001, The Tor Project, September 2011. https://research.torproject.org/techreports/detector-2011-09-09.pdf.

[2] Roger Dingledine. Strategies for getting more bridge addresses. Technical Report 2011-05-001, The Tor Project, May 2011. https://research.torproject.org/techreports/strategies-getting-more-bridge-addresses-2011-05-13.pdf.

[3] Roger Dingledine. Ten ways to discover Tor bridges. Technical Report 2011-10-002, The Tor Project, October 2011. https://research.torproject.org/techreports/ten-ways-discover-tor-bridges-2011-10-31.pdf.

[4] Ryan Henry and Ian Goldberg. Formalizing anonymous blacklisting systems. In *Proceedings of the 2011 IEEE Symposium on Security and Privacy*, May 2011. http://freehaven.net/anonbib/#oakland11-formalizing.

[5] Kirill Levchenko and Damon McCoy. Proximax: Fighting censorship with an adaptive system for distribution of open proxies. In *Proceedings of Financial Cryptography and Data Security (FC'11)*, February 2011. http://freehaven.net/anonbib/#proximax11.

[6] Karsten Loesing. Case study: Learning whether a Tor bridge is blocked by looking at its aggregate usage statistics, part one. Technical Report 2011-09-002, The Tor Project, September 2011. https://research.torproject.org/techreports/blocking-2011-09-15.pdf.

[7] Karsten Loesing, Steven J. Murdoch, and Roger Dingledine. A case study on measuring statistical data in the Tor anonymity network. In *Proc. Workshop on Ethics in Computer Security Research*, Tenerife, Canary Islands, Spain, January 2010. https://metrics.torproject.org/papers/wecsr10.pdf.