

October 2022 Internet Freedom RACE Follow-On

Roger Dingledine

October 13 2022

The Tor Project



(0) Tor's track record in Internet Freedom

- (1) Understand where we are censored and react to it
- (2) Strengthen Snowflake
- (3) Make our tooling more agile

Track record in Internet Freedom (1/3)

Leading circumvention innovation

- Tor developed the concept of Pluggable Transports in 2010. Now, PTs are used in major circumvention tools and are their own field of research.

Leading privacy innovation

- Features born in Tor Browser (like “first-party isolation”) have been adopted as standard features in browsers like Firefox, Safari, and Brave.

Building a research community

- Tor people helped found Privacy Enhancing Technologies Symposium (PETS) and Free and Open Communications on the Internet (FOCI) to foster research in privacy and censorship circumvention.

Track record in Internet Freedom (2/3)

Legacy of real-world success

- From helping people stay connected during the Arab Spring uprisings to supporting Russians to route around government censorship.

Adoption by major players

- Leading social media and news outlets use Tor to help their users circumvent censorship and to communicate safely with sources.

Invested community

- More than 80,000 volunteers have started Snowflake proxies in the last 30 days, almost tripling the number of proxies in the network.

Track record in Internet Freedom (3/3)

Recognition

- Levchin Prize for Real-World Cryptography (2021)
- USENIX Test of Time Award (2014)
- EFF Pioneer Award (2012)
- Free Software Foundation Social Benefit Award (2011)

Supporting an ecosystem / critical infrastructure

- A large ecosystem of tools rely on Tor to provide circumvention and privacy options to their users.
- We host subgrantees in our funded projects, to build an ecosystem of collaborating circumvention tools and strong organizations.

Always free software, always open source. Since 2003.

Objectives

(0) Tor's track record in Internet Freedom

(1) Understand where we are censored and react to it

(2) Strengthen Snowflake

(3) Make our tooling more agile

The bridge “subscription” model (1/3)

We distribute bridges in many ways today (https, gmail, telegram, moat).

Each way has some different barrier to access, which aims to slow bridge enumeration.

But bridges go down, or get blocked, or they're on dynamic IP addresses and they just move, so users need to ask for new ones.

The key insight is: the barrier-to-access can be different for follow-on requests.

The bridge “subscription” model (2/3)

Step one: flexible new protocol between Tor Browser and our back-end bridge service:

⇒ (1) client presents something ⇒

⇐ (2) service replies with bridge info ⇐

(3) Tor Browser auto adjusts our bridge list

Conceptually, this extends Tor Browser’s “moat” design that fetches new bridges + presents a Captcha in-browser.

The bridge “subscription” model (3/3)

The subscription model is a key building block for future plans:

- auto replace bridges that go down
- fast flux bridges that get blocked
- reputation-based bridge distribution schemes like Salmon

For first two, client sends hash-of-bridge-fingerprint to prove knowledge of bridge. For Salmon/Lox, it's a cryptographic identity.

How to choose a replacement bridge?

Easy: bridge moves to a new IP address $\Rightarrow$ give users its new info.

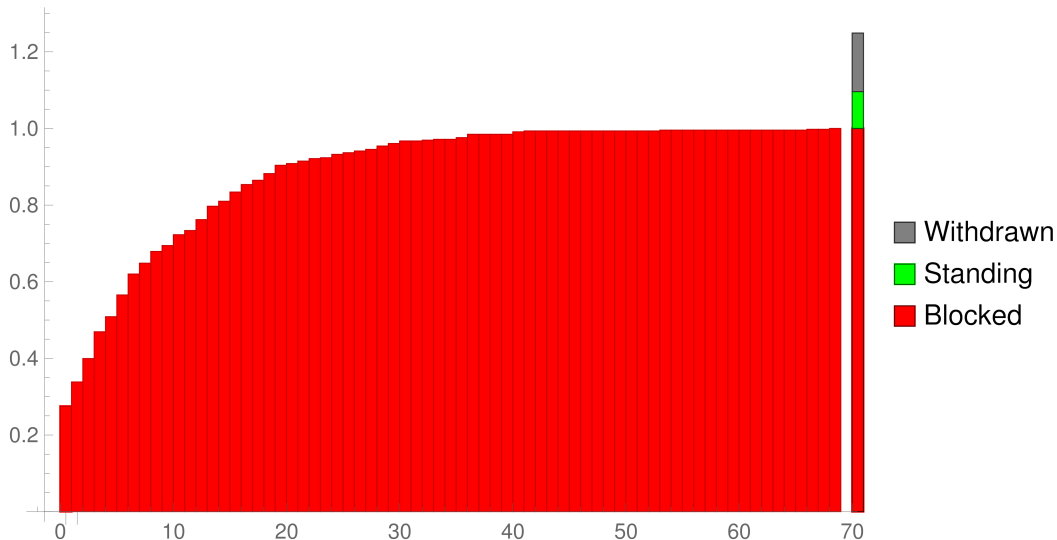
Easy: bridge goes offline $\Rightarrow$ give users a deterministic replacement.

Harder: bridge gets blocked.

Auto-replacing means the censor auto-gets the new one too!

Whether to auto-replace blocked bridges depends on context: how dynamic are our bridges vs how quick is the censor?

Public bridges blocked in China in days to weeks



Strategies for choosing a replacement bridge

Reconcile the contradiction by having two different classes of bridges—volunteer ones and centrally managed ones. Can cycle the managed ones quickly, to play a faster arms race with them.

Remaining agile here is key: we will start with a simple plan for *how* we will choose a replacement bridge, and then we will learn from our deployment and adapt.

Using bridge reachability vantage points

Over the past months we've been doing active bridge reachability measurements from three countries: China, Russia, Turkey.

Privacy-preserving daily snapshots are published on <https://gitlab.torproject.org/tpo/anti-censorship/connectivity-measurement/bridgestatus>

With the “vantage point”, “subscription model”, “dynamic cloud bridges”, and “signaling channel” building blocks, the next step is to complete the feedback loop: we need to auto notice that bridges have been blocked, and auto replace the bridge for the user.

Refinements at the client side

When exactly should Tor Browser launch a bridge refresh attempt?

Every time it fails to bootstrap? Periodically?

What about when bridges fail after bootstrap? How to distinguish from simple network failure.

Refinements at the scanning side

We want to scan failed bridges sooner—so we can get working bridges to users asap.

We want to scan working bridges less often—so we expose less info at our vantage points.

Long term vision: use heuristics for prioritizing which bridges to test from our sensitive vantage points, e.g. observing a drop in usage or a spike in users requesting a replacement for it.

Scanning for ‘down’ is a different race than scanning for ‘blocked’, but getting really good at the former can help the latter.

rdsys + bridgestatus + bridgestrap integration

Strengthen the feedback loop between rdsys and the vantage points.

Need to expand the vantage points to more countries.

Objectives

- (0) Tor's track record in Internet Freedom
- (1) Understand where we are censored and react to it
- (2) Strengthen Snowflake**
- (3) Make our tooling more agile

Better understand Snowflake use and bottlenecks

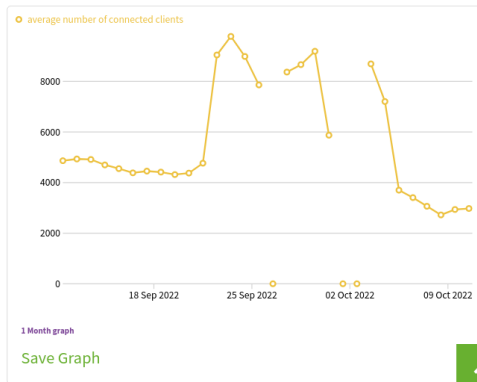
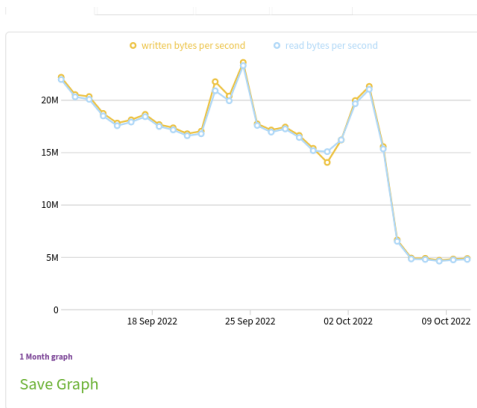
Initial goal: Make Snowflake user counts more accurate.

Right now we mainly count Snowflake users via self-reported “consensus fetch” counts at the bridge.

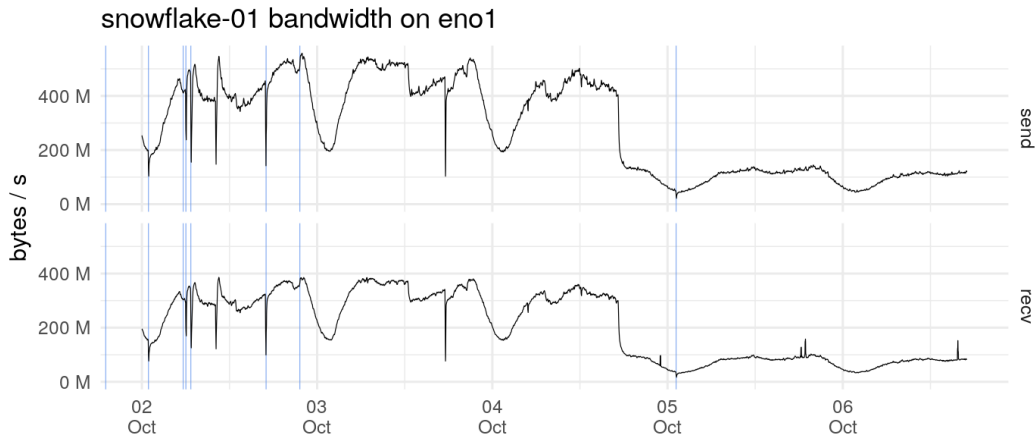
But counting by consensus fetch undercounts by as much as 10x.

Worse, our “many bridges, one fingerprint” scalability band-aid exposed bugs in the metrics portal: another 12x undercount!

Snowflake activity reported by metrics portal



Snowflake activity reported by back-end server





باران !! توییت پین !!

@radfembaran

...

ایرانی‌های خارج نشین، می‌خواین کمک کنین؟
Snowflake رو نصب کنید، مثل یه پل
میمونه. من الان هیچکدوم از VPN هام کار
نمیکنن، Orbot رو نصب کردم و کانکت شدم
و به لطف بچه‌های خارج که Snowflake
نصب کردن توییترو، تلگرام و واتسپ رو باز
کردم. #مهسا_امینی #Oplran
لینکش:

snowflake.torproject.org

Translate Tweet



ch .lll VPN

2:27 PM

@ 8

Chats



Google Play Ranking: The Top Free Overall in Iran

Track the rankings of your Android apps for free with AppBrain. The rankings are refreshed daily from Google Play.

CSV

Top Free

Iran

Overall

Rank		App	Category	Rating	Installs
1	=	 Tor Browser by The Tor Project	Communication	★★★★★ 4.4	10 M+
2	▲1	 Orbot: Tor for Android by The Tor Project	Communication	★★★★★ 4.2	10 M+
3	▼1	 HulaVPN - Fast Secure VPN by Hula Link	Tools	★★★★★ 4.2	1 M+
4	=	 VPN Fast - Secure VPN Proxy by Phone Master Lab	Tools	★★★★★ 4.7	10 M+
5	=	 Turbo VPN - Secure VPN Proxy by Innovative Connecting	Tools	★★★★★ 4.7	100 M+
6	=	 Ultrasurf - Fast Unlimited VPN by Ultrareach	Tools	★★★★★ 4.8	10 M+

Instrument each layer to better understand usage

We have other sources of information:

- the broker knows how many requests, how many proxies, how many matching attempts
- the snowflake back-end knows how many sessions

The broker can count connection attempts, different from successful users.

Especially important to be able to watch these stats as we make more architectural changes e.g. multiple snowflake back-ends.

Improve Snowflake's denial-of-service resilience

Make and implement a roadmap for ensuring Snowflake keeps working in the future—even with increased scrutiny.

Understand the scope of the problem + design mitigations.

Two focuses:

- Redundancy of components (e.g. deploy multiple brokers, multiple bridges, multiple proxy pools)
- Agility at detecting and handling protocol-level attacks (e.g. somebody pretends to be a million snowflake users and uses up all our Snowflakes).

Objectives

- (0) Tor's track record in Internet Freedom
- (1) Understand where we are censored and react to it
- (2) Strengthen Snowflake
- (3) Make our tooling more agile**

We're replacing BridgeDB with rdsys

BridgeDB has grown organically since we first wrote it in 2007.

At this point, maintenance by new developers is a constant source of surprises. The last change broke translations when it was an unrelated change.

We especially need agility for adjusting bridge assignments, e.g. if we find that China is doing something new around IP ranges and we want to change which bridges we give out via moat.

A refresh redesign, named rdsys, is making it better.

Some Snowflake pieces are still proofs-of-concept

The broker aims to be simple (just match users to proxies); some users not being matchable to some proxies was not expected.

We want to expand to multiple proxy pools, multiple brokers, multiple back-ends, load balance better across proxies.

Eventually we will want to keep track of which proxies work well in which countries, and the matching constraints will get even more complex.

Making the distributors agile again

Two focuses here:

(A) redesign the bridge assignment / snowflake assignment code so it is easy to add or change constraints.

(B) staging environments, sandboxes, and continuous integration, for testing changes in each of these services.

These staging / sandbox environments will let us experiment and try out new ideas without putting the production services at risk.

Bringing it all together

- (1) Understand where we are censored and react to it
 - design and implement subscription model
 - use it to auto-replace broken bridges for Tor Browser users
- (2) Strengthen Snowflake
 - instrument each layer to better understand usage
 - redundancy of components, look at protocol-level attacks
- (3) Make our tooling more agile
 - redesign the bridge assignment / snowflake assignment code
 - staging (sandbox) environments and continuous integration