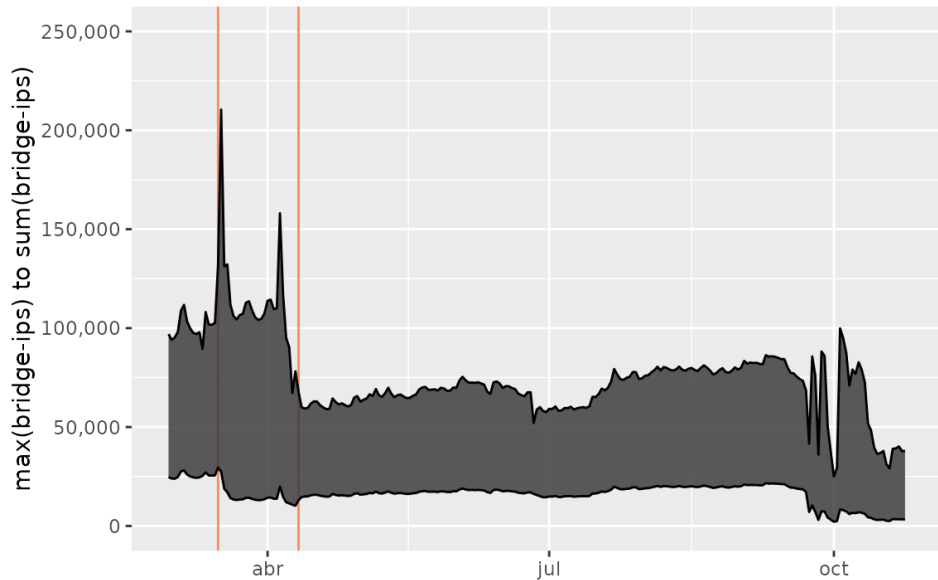
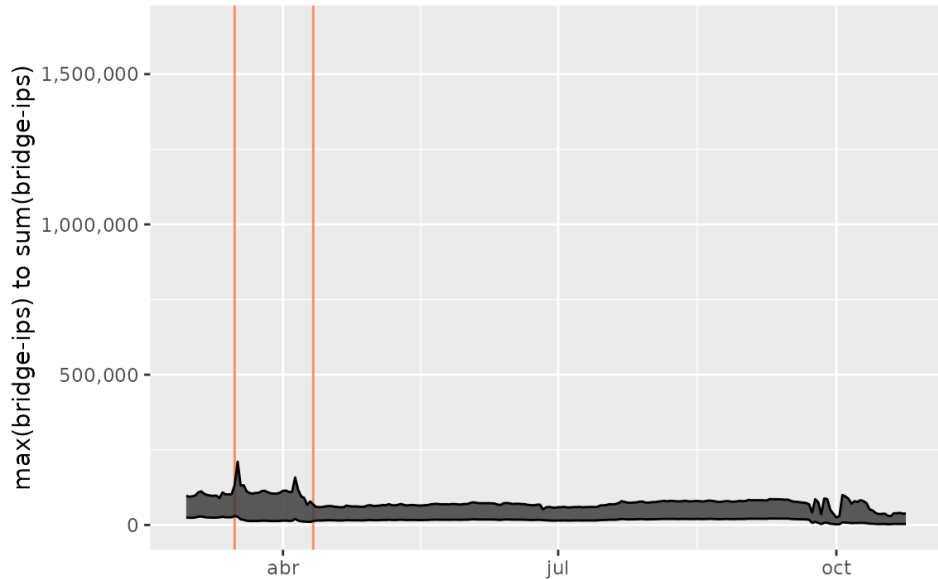


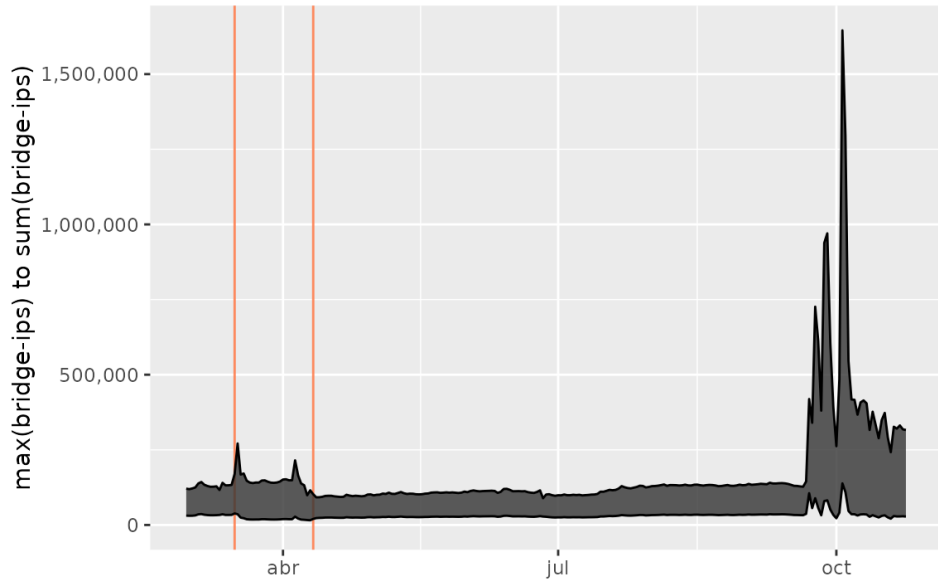
Snowflake unique daily users estimate (Russia only)

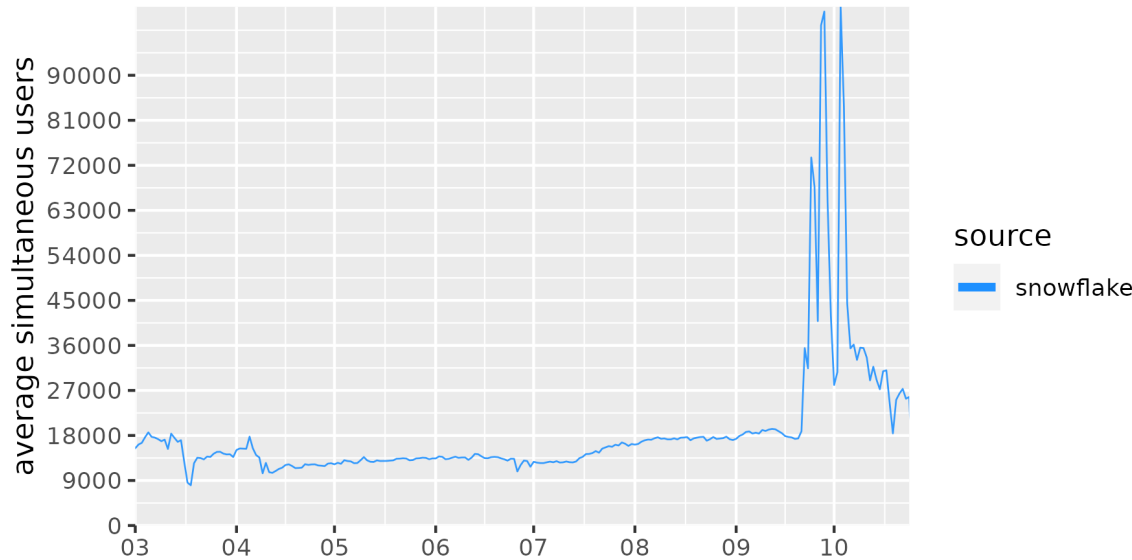


Snowflake unique daily users estimate (Russia only)



Snowflake unique daily users estimate







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



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

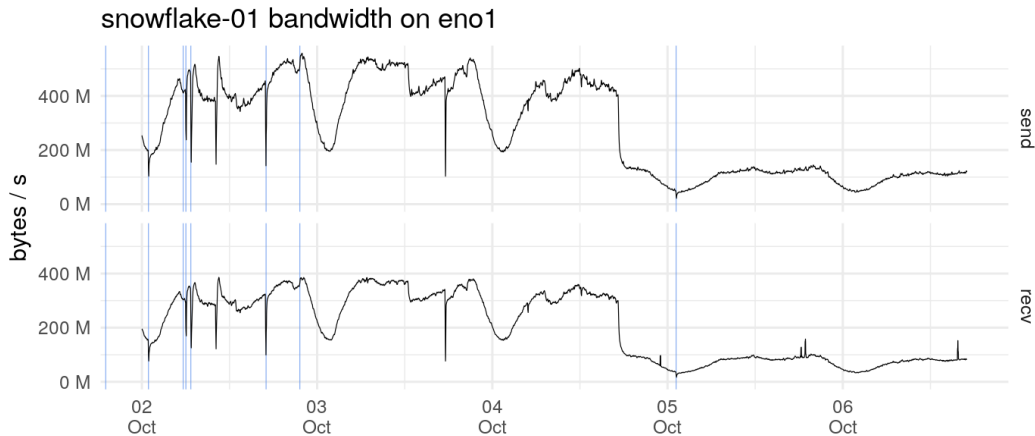
Top Free

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+

Snowflake activity reported by back-end server



S



Sudden reduction in snowflake-01 bridge bandwidth, 2022-10-04 17:15



D

David Fifield @dcf · 3 weeks ago

Author

Owner



I think I found the cause. The snowflake-server process was running out of file descriptors. I am not sure why that should have caused such a drastic reduction in throughput, but the log messages around the time of the drop are clear:

```
2022/10/04 17:13:59 http: Accept error: accept tcp [scrubbed]: accept4: too many open files; r
2022/10/04 17:13:59 http: Accept error: accept tcp [scrubbed]: accept4: too many open files; r
2022/10/04 17:13:59 http: Accept error: accept tcp [scrubbed]: accept4: too many open files; r
2022/10/04 17:13:59 http: Accept error: accept tcp [scrubbed]: accept4: too many open files; r
2022/10/04 17:13:59 handleConn: failed to connect to ORPort: dial tcp [scrubbed]->[scrubbed]:
2022/10/04 17:13:59 http: Accept error: accept tcp [scrubbed]: accept4: too many open files; r
2022/10/04 17:13:59 http: Accept error: accept tcp [scrubbed]: accept4: too many open files; r
2022/10/04 17:13:59 http: Accept error: accept tcp [scrubbed]: accept4: too many open files; r
2022/10/04 17:13:59 http: Accept error: accept tcp [scrubbed]: accept4: too many open files; r
2022/10/04 17:13:59 handleConn: failed to connect to ORPort: error reading TOR_PT_AUTH_COOKIE_
2022/10/04 17:13:59 http: Accept error: accept tcp [scrubbed]: accept4: too many open files; r
2022/10/04 17:13:59 http: Accept error: accept tcp [scrubbed]: accept4: too many open files; r
2022/10/04 17:13:59 http: Accept error: accept tcp [scrubbed]: accept4: too many open files; r
2022/10/04 17:13:59 http: Accept error: accept tcp [scrubbed]: accept4: too many open files; r
2022/10/04 17:13:59 handleConn: failed to connect to ORPort: error reading TOR_PT_AUTH_COOKIE_
2022/10/04 17:13:59 http: Accept error: accept tcp [scrubbed]: accept4: too many open files; r
2022/10/04 17:13:59 http: Accept error: accept tcp [scrubbed]: accept4: too many open files; r
2022/10/04 17:13:59 http: Accept error: accept tcp [scrubbed]: accept4: too many open files; r
2022/10/04 17:13:59 handleConn: failed to connect to ORPort: error reading TOR_PT_AUTH_COOKIE_
2022/10/04 17:13:59 handleConn: failed to connect to ORPort: error reading TOR_PT_AUTH_COOKIE_
2022/10/04 17:14:01 http: TLS handshake error from [scrubbed]: EOF
2022/10/04 17:14:03 http: TLS handshake error from [scrubbed]: read tcp [scrubbed]->[scrubbed]
```



1



No mil...



None



None



Tor

8

3

99+

S

Open

Sudden reduction in snowflake-01 bridge bandwidth, 2022-10-04 17:15

country	2022-10-04 17:01:14	2022-10-05 17:01:14	increase/decrease
ir	52408	15680	-70%
??	12232	8208	-33%
us	10560	4480	-58%
ru	5232	5056	-3%
cn	656	600	-9%
mu	648	384	-41%
tn	472	200	-58%
de	408	624	+53%
ma	256	112	-56%
gb	208	272	+31%
eg	200	168	-16%

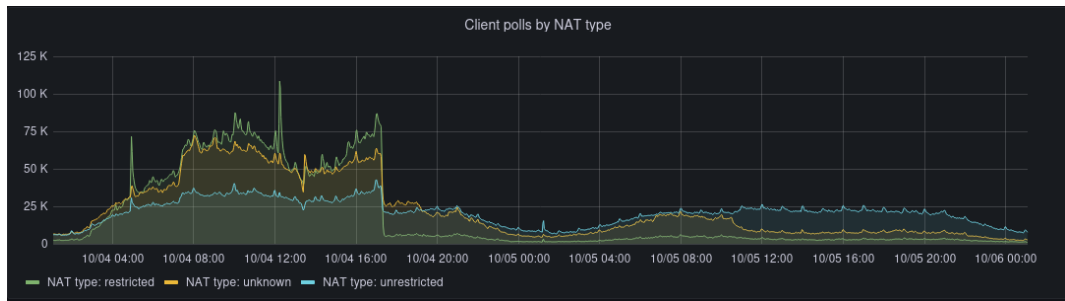
<<

1

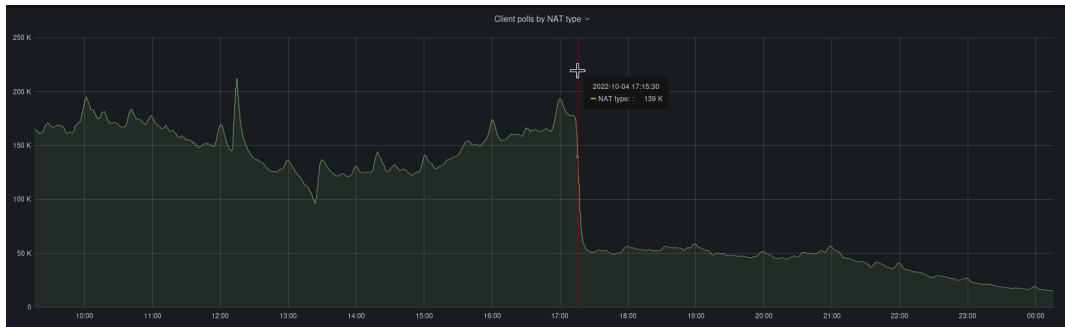
No mil...

None

None



number of clients contacting the Snowflake broker



S

Open

Sudden reduction in snowflake-01 bridge bandwidth, 2022-10-04 17:15

D

D

David Fifield @dcf · 1 week ago

Author

Owner

😊

💬

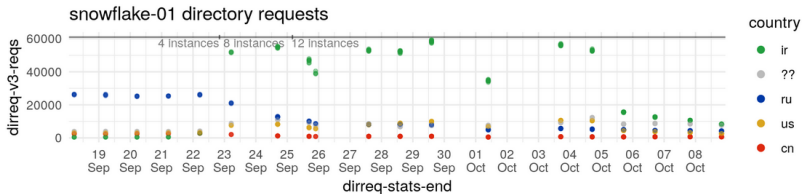
⋮

However the metrics *do* show that the cause is not a country-wide block of broker rendezvous.

I have a new working hypothesis: the sudden decline *is* caused by a partial block of the broker in Iran. As for why countries other than Iran are seemingly affected, my best guess is that they are geolocation errors: IP addresses in Iran wrongly being attributed to other countries (chiefly `us` and `??`).

Below are the top 20 countries according to `dirreq-v3-reqs` on 2022-10-04, and how much they changed 24 hours later.

Here's that information with more context.



📎 [snowflake-01-dirreqs-20221014.zip](#)

«

✓

👤

📄 1

🕒

No mil...

📅

None

🕒

None

👁

🔒

»



Unexplained drop in Snowflake client polls and bandwidth, testers wanted #131

wkrp opened this issue 21 days ago · 68 comments



SaSyda commented 21 days ago

...

ok, the thing is i've been using tor in the first days of outage, but for certain reasons, been using other solution for the days since. As i saw ur post last night, decided to help out and started things up, first thing i realised, i could connect from my pc but not my mac laptop, but on a second thought i'm using tor browser on windows and orbot on mac. so i downloaded tor browser on mac os and wow! it's connecting with superspeeds, so, my conclusion, it's not actually a problem of ur systems, but the fact that people are mainly using orbot, and whatever thing they've done to stop us, is related to that app and the streams its using and going through. actually i'm more confident about my assumption, cuz i've suggested orbot to many people as it runs on mobile devices and clearly most users in iran use a phone to get to websites like instagram or certain messengers. And computer celebs over twitter and other places, been suggesting that app as well.

And i assume, as I can connect to the free internet, my log is no use to u, but still, tell me if u need me to send it as i could successfully go through ur tutorial.

and in my next experiments, i'll be using cellular connection, as it has been way heavier censored and the so called "national internet" which is an actual intranet with measures to limit connection to foreign servers and computers, is mainly implemented on OTG internet connections.

Reachability testing from inside Iran

	Works?
Tor Browser Linux	yes
Tor Browser Android	yes
Orbot	yes
Orbot	no

Tor

≡

🔍

+

▼

📺

8

🔗

3

▼

✅

99+

?

▼

👤

▼

S

Open

Sudden reduction in snowflake-01 bridge bandwidth, 2022-10-04 17:15

⏪

📖

📄

📄

🔗

🚀

🛡️

🕒

🏠

📺

📊

📄

✂️

⏪

D

David Fifield @dcf · 1 week ago

Author

Owner

😊

⋮

Thank you, that's very helpful. The fingerprint of your orbot.pcap is indeed identical to [my one](#). In `tlsfingerprint.io` terms, it is `adfe55afa6f23950`.

This fingerprint `adfe55afa6f23950` [differs only minorly](#) from `750e3f0f585283bd`, which was observed in <https://github.com/net4people/bbs/issues/139#issuecomment-1280057679> to be produced by a Go program running on Raspberry Pi.

The critical thing in native Go crypto/tls fingerprints [since go1.17](#) is that the order of ciphersuites depends on whether the platform has support for accelerated AES-GCM. See how there are two versions of everything: `cipherSuitesPreferenceOrder` and `cipherSuitesPreferenceOrderNoAES`; `defaultCipherSuitesTLS13` and `defaultCipherSuitesTLS13NoAES`. This explains the two different ciphersuite orderings you observed. The choice of which to use [happens at runtime](#):

```
hasGCMAsmAMD64 = cpu.X86.HasAES && cpu.X86.HasPCLMULQDQ
hasGCMAsmARM64 = cpu.ARM64.HasAES && cpu.ARM64.HasPMULL
hasGCMAsmS390X = cpu.S390X.HasAES && cpu.S390X.HasAESCBC && cpu.S390X.HasAESCTR &&
    (cpu.S390X.HasGHASH || cpu.S390X.HasAESGCM)
hasAESGCMHardwareSupport = runtime.GOARCH == "amd64" && hasGCMAsmAMD64 ||
    runtime.GOARCH == "arm64" && hasGCMAsmARM64 ||
    runtime.GOARCH == "s390x" && hasGCMAsmS390X
```

⏪

📄

👤

📄

1

🕒

No mil...

📅

None

🕒

None

👁️

🔒

Reachability testing from inside Iran

	Works?	
Tor Browser Linux	yes	⇐ go 1.17, AES-GCM
Tor Browser Android	yes	⇐ go 1.18, AES-GCM
Tor Browser Android	yes	⇐ go 1.18, no AES-GCM
Orbot	yes	⇐ go 1.17, AES-GCM
Orbot	no	⇐ go 1.17, no AES-GCM



build failing godoc reference

uTLS is a fork of "crypto/tls", which provides ClientHello fingerprinting resistance, low-level access to handshake, fake session tickets and some other features. Handshake is still performed by "crypto/tls", this library merely changes ClientHello part of it and provides low-level access.

Golang 1.11+ is required.

If you have any questions, bug reports or contributions, you are welcome to publish those on GitHub. If you want to do so in private, you can contact one of developers personally via sergey.frolov@colorado.edu

Documentation below may not keep up with all the changes and new features at all times, so you are encouraged to use [godoc](#).

Features



Shutdowns, intensified blocking in Iran since 2022-09-21 #125

wkrp opened this issue on Sep 21 · 45 comments



n8fr8 commented 7 days ago



Orbot for Android 16.6.3-BETA-2-tor.0.4.7.10 with utls enabled, now available here:
<https://github.com/guardianproject/orbot/releases/tag/16.6.3-BETA-2-tor.0.4.7.10>

(arm64 direct APK: <https://github.com/guardianproject/orbot/releases/download/16.6.3-BETA-2-tor.0.4.7.10/Orbot-16.6.3-BETA-2-tor.0.4.7.10-fullperm-arm64-v8a-release.apk>)

This uses "utls-imitate=hellochrome_auto" - we will add the other options and ability to customize/select in the next update.

This release also has the ability to get Snowflake logs directly from the log window (enable Prefs->DEBUG log, tap on status messages to open log window, tap on snowflake icon to show snowflake log, then share!)



2



2



mehdifirefox commented 7 days ago





Unexplained drop in Snowflake client polls and bandwidth, testers wanted #131

wkrp opened this issue 21 days ago · 68 comments



SaSyda commented 8 days ago



@wkrp hello man and sorry for my long absence, i dont think u would want to know where i've been these day. Got back here to see your new theory and man u are right, just tested tor browser on android with ur custom snowflake, and it works, not perfectly or fast, but does connect and brings certain restricted websites. actually i'm not that of a professional in that app. but i was able to test it and really, well done. I'll be back with more details soon. keep rocking.



1



 **quartermarsh** commented 8 days ago • edited



@hexrot It may recover after a few hours of idle. Wait and see. A restart of the proxy can help (assuming you're running standalone). The snowflake team is working on it.

<https://gitlab.torproject.org/tpo/anti-censorship/pluggable-transport/snowflake>



Unexplained drop in Snowflake client polls and bandwidth, testers wanted #131

wkrp opened this issue 21 days ago · 68 comments



iRhonin commented 7 days ago



There is now a release available of Orbot that enables uTLS for Snowflake (from [#125 \(comment\)](#)).

You can download APKs here: <https://github.com/guardianproject/orbot/releases/tag/16.6.3-BETA-2-tor.0.4.7.10>

This release makes it possible to see the snowflake-client log. If there's a failure to connect, it will help us figure out what is going wrong. Enable **Settings** → **Debug Log**, then go back to the main screen and **Start**. Tap on a status message to show the tor log, then tap the **snowflake** snowflake icon to view the snowflake-client log.

I can confirm this works in Iran.



2



2



free-the-internet commented 7 days ago



uTLS for broker negotiation

The connection from the snowflake client to the broker server currently uses go's `net/http.DefaultTransport`. That connection is optionally domain fronted, but the TLS handshake is easily fingerprintable as a go handshake, which might trigger additional scrutiny in regimes where that kind of TLS inspection is possible and actionable.

[This paper](#), though a bit out of date now (2019) references meek and even snowflake:

Snowflake is under active development, and its authors were aware of potential TLS fingerprintability issues. Indeed, we find that Snowflake (built from git master branch on April 17, 2018) generates a fingerprint that is close to, but not exactly the same as the default Golang TLS fingerprint. In particular, it diverges by including the NPN and ALPN extensions, and offers a different set of signature algorithms. As a result, this fingerprint is seen in fewer than 0.0008% of connections, making it susceptible to blocking.

The author of that paper, Sergey Frolov, maintains <https://tlsfingerprint.io/> which is a list of the most popularly seen TLS fingerprints, and created <https://github.com/refraction-networking/utls> which is a library designed for creating TLS connections with various commonly witnessed TLS fingerprints.

There's a fork of that library, <https://gitlab.com/yawning/utls> which seems to be used in obfs4's `meeklite` implementation, and it looks like `@dcf` implemented a version of that in <https://gitweb.torproject.org/pluggable-transports/meek.git/tree/meek-client/utls.go> which actually implements a `RoundTripper`. It seems as though actually using that library could be a relatively painless way to adopt utls for the broker negotiation.

```
snowflake 192.0.2.3:80
2B280B23E1107BB62ABFC40DDCC8824814F80A72
fingerprint=2B280B23E1107BB62ABFC40DDCC8824814F80A72
url=https://snowflake-
broker.torproject.net.global.prod.fastly.net/
front=cdn.sstatic.net
ice=stun:stun.l.google.com:19302,stun:stun.altar.com.pl:3478,
stun:stun.antisip.com:3478,stun:stun.bluesip.net:3478,stun:st
un.dus.net:3478,stun:stun.epygi.com:3478,stun:stun.sonetel.co
m:3478,stun:stun.sonetel.net:3478,stun:stun.stunprotocol.org:
3478,stun:stun.uls.co.za:3478,stun:stun.voipgate.com:3478,stu
n:stun.voys.nl:3478 utls-imitate=hellorandomizedalpn
```

Incorrect (non canonical) representative output for `ScalarBaseMult()` #27

New issue



LoupVallant opened this issue on Feb 24, 2020 · 3 comments



LoupVallant commented on Feb 24, 2020

...

Hi,

I learned of this bug was found by [@tankf33der](#). Contrary to the original paper, the Elligator 2 representative of a public key is not always canonical. That is, this library does not make sure the output stay between 0 and $(p-1)/2$ (that is, $2^{254}-10$).

The mapping is such that for each point on the curve we can map, two representative (r_1 and r_2) map to that point on the curve, and those points are such that $r_1 + r_2 = (2^{255} - 19)$. Which means one of them is in $[0, 2^{254}-11]$, and the other is in $[2^{254}-10, 2^{255}-18]$. We are supposed to chose the smaller of the two. If we don't, there is no guarantee the distribution of canonical and non-canonical output will be even. And if it's not, we'll introduce exactly the kind of bias Elligator was meant to eliminate, and basically ruin everything.

The problem can be reproduced with this code:

Assignees

No one assigned

Labels

None yet

Projects

None yet

Milestone

No milestone

Development

obfs distinguishers

First distinguisher, bit 254 should be randomized because we should be picking the lower of the two representations, but we pick either.

Second distinguisher, <details embargoed until Nov 15>.

Two-ish total bits of distinguisher so not the end of the world.

Partly fixed in obfs4proxy 0.0.12, 2021-12-31. Fixed better in obfs4proxy 0.0.14, 2022-09-04.

Dynamic bridge reachability experiment

44 dynamic bridges given out both via moat (in-browser captcha) and via telegram+new. 10 given out only via telegram+new.

Tentative results:

- moat mostly blocked in China
- telegram+new mostly working in China
- both classes working in Russia and Turkey

Follow-up question: what about user load on each class of bridge?