What if the Tor network had 50,000 bridges?

Karsten Loesing
karsten@torproject.org

Tor Tech Report 2012-03-001
March 9, 2012

1 Introduction

The current bridge infrastructure relies on a central bridge authority to collect, distribute, and publish bridge relay descriptors. There are currently 1,000 bridges running in the Tor network.[1] We believe the current infrastructure can handle up to 10,000 bridges. Potential performance bottlenecks include:

- the bridge authority Tonga, where all (public) bridges register and which performs periodic reachability tests to confirm that bridges are running,
- BridgeDB, which stores currently running bridges and hands them out to bridge users, and
- metrics-db, which sanitizes bridge descriptors for later analysis like statistics on daily connecting bridge users.

2 Load-testing BridgeDB and metrics-db

We started this analysis by writing a small tool to generate sample data for BridgeDB and metrics-db to load-test them. This tool takes the contents from one of Tonga's bridge tarball as input, copies them a given number of times, and overwrites the first two bytes of relay fingerprints in every copy with 0000, 0001, etc. The tool also fixes references between network statuses, server descriptors, and extra-info descriptors. This is sufficient to trick BridgeDB and metrics-db into thinking that bridges in the copies are distinct bridges. We used the tool to generate tarballs with 2, 4, 8, 16, 32, and 64 times as many bridge descriptors in them.

In the next step we fed the tarballs into BridgeDB and metrics-db. BridgeDB reads the network statuses and server descriptors from the latest tarball and writes them to a local database. metrics-db sanitizes two half-hourly created tarballs every hour, establishes an

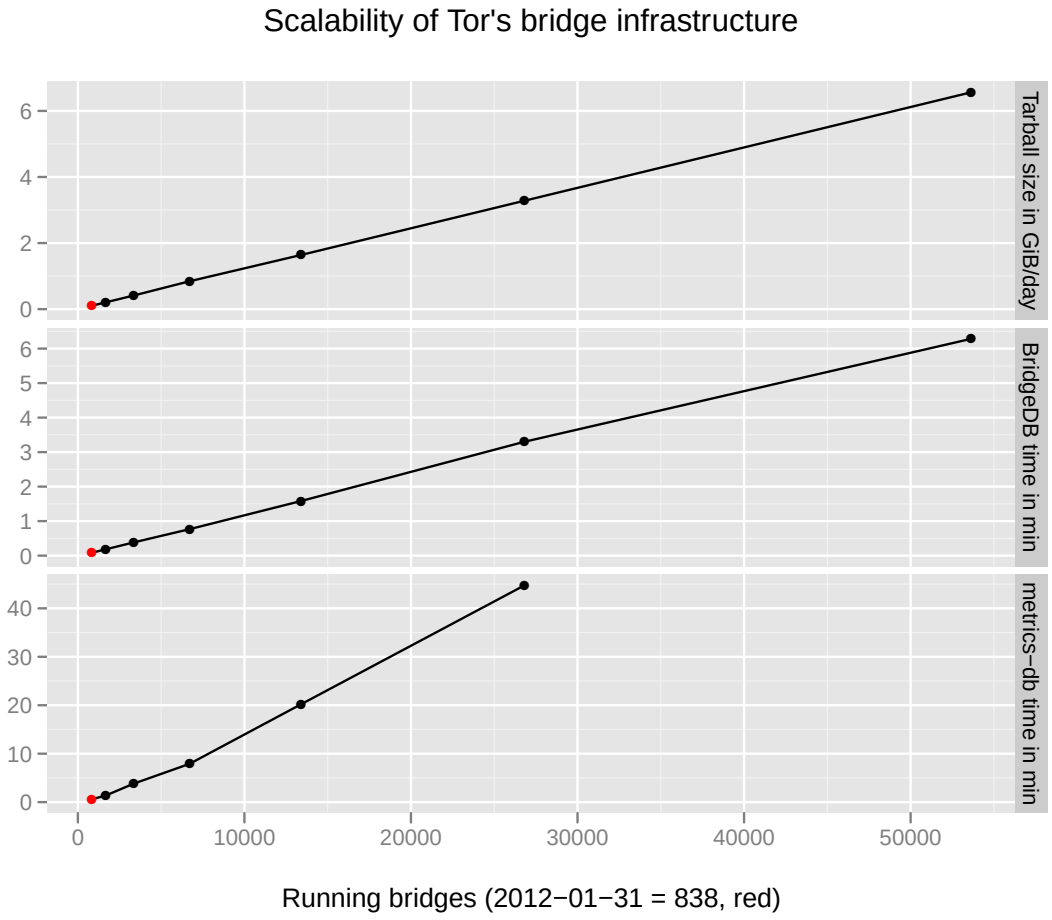

Figure 1: Results from load-testing BridgeDB and metrics-db

internal mapping between descriptors, and writes sanitized descriptors with fixed references to disk. Figure 1 shows the results.

The upper graph shows how the tarballs grow in size with more bridge descriptors in them. This growth is, unsurprisingly, linear. One thing to keep in mind here is that bandwidth and storage requirements to the hosts transferring and storing bridge tarballs are growing with the tarballs. We'll want to pay extra attention to disk space running out on those hosts. These tarballs have substantial overlap. If we have tens of thousands of descriptors, we would want to get smarter at sending diffs over to BridgeDB and metrics-db.[2]

The middle graph shows how long BridgeDB takes to load descriptors from a tarball. This graph is linear, too, which indicates that BridgeDB can handle an increase in the number of bridges pretty well.

The lower graph shows how metrics-db can or cannot handle more bridges. The growth is slightly worse than linear. In any case, the absolute time required to handle 25K bridges is worrisome (we didn't try 50K). metrics-db runs in an hourly cronjob, and if that cronjob doesn't

[1] https://metrics.torproject.org/network.html#networksize

[2] See comment at https://trac.torproject.org/projects/tor/ticket/4499#comment:7

finish within 1 hour, we cannot start the next run and will be missing some data. We might have to sanitize bridge descriptors in a different thread or process than the one that fetches all the other metrics data. We can also look into other Java libraries to handle .gz-compressed files that are faster than the one we're using.

3 Looking at concurrency in BridgeDB

While performing the load-test on BridgeDB we were wondering whether it can serve client requests while loading bridges. Turns out BridgeDB's interaction with users freezes while it's reading a new set of data. This isn't that much of a problem with a few hundred bridges and unlucky clients having to wait 10 seconds for their bridges. But it becomes a problem when BridgeDB is busy for a minute or two, twice an hour. We started discussing importing bridges into BridgeDB in a separate thread and database transaction.[3]

4 Scalability of the bridge authority Tonga

We left out the most important part of this analysis: can Tonga, or more generally, a single bridge authority handle this increase in bridges? Tonga still does a reachability test on each bridge every 21 minutes or so. Eventually the number of TLS handshakes it's doing will overwhelm its CPU.[4]

We're not sure how to test such a setting, or at least without running 50K bridges in a private network. We could imagine this requires some more sophisticated sample data generation including getting the crypto right and then talking to Tonga's DirPort. We didn't find an easy way to test this.

A possible fix would be to increase the reachability test interval from 21 minutes to some higher value. A long-term fix would be to come up with a design that has more than one single bridge authority.

5 Conclusion

In conclusion, we found that a massive increase in bridges in the Tor network by a factor of 10 to 50 can be harmful to Tor's infrastructure. We identified possible bottlenecks: Tonga's reachability test interval, bridge tarball sizes for transfer between Tonga and BridgeDB/metrics-db, loading bridges into BridgeDB, and sanitizing bridges in metrics-db.

During this analysis we discovered a design bug in BridgeDB which makes it freeze while reading new bridge descriptors. This bug should be fixed regardless of scaling to 10K–50K bridges, because it already affects users. The suggested changes to Tonga, transfering tarballs between hosts, and changes to metrics-db can be postponed until there's an actual problem, not just a theoretical one.

[3] https://trac.torproject.org/projects/tor/ticket/5232
[4] https://trac.torproject.org/projects/tor/ticket/4499#comment:7