

Design and Evaluation of Tor Hidden Service Performance Improvements

Roger Dingledine, Karsten Loesing, Steven J. Murdoch,
and Christian Wilms*

August 15, 2008

Abstract

This document describes bugfixes and design changes to overcome performance bottlenecks in the Tor Hidden Service protocol that have been analyzed in an earlier report. While the bugfixes remove unintended performance bottlenecks due to programming errors, some of the design changes have side-effects on anonymity or overall network load which have been weighed up against individual performance gains.

1 Bugfixes

The performance analysis¹ has brought up a couple of bugs in the implementation of hidden services. The following description includes the bugs found in the previous analysis together with the current state of their fix on August 15, 2008.

1.1 Premature Descriptor Upload

In very rare situations new hidden service descriptors were published earlier than 30 seconds after the last change to the service, although the current thinking is that a hidden service descriptor that's been stable for 30 seconds is worth publishing. This minor bug was in the code since Tor version 0.0.9pre6 released on November 15, 2004. This bug is fixed in Tor version 0.2.1.1-alpha released on June 13, 2008.

1.2 Abandoning Valid Introduction Points

While setting up a hidden service, some valid introduction circuits were overlooked and abandoned. This might be the reason for the long delay in making

*Please direct questions and comments either to tor-assistants@torproject.org or or-dev@freehaven.net.

¹<http://freehaven.net/~karsten/hidserv/perfanalysis-2008-06-15.pdf>

a hidden service available. This major bug was introduced with Tor version 0.2.0.14-alpha released on December 23, 2007. It is now fixed and included in Tor 0.2.1.1-alpha and 0.2.0.28-rc which were both released on June 13, 2008.

1.3 Ignoring Cannibalized Introduction Points

When establishing a hidden service, introduction points that originate from cannibalized circuits were completely ignored and not included in rendezvous service descriptors. This might be another major reason for the delay in making a hidden service available. This bug was fixed for Tor version 0.2.1.2-alpha that was released on June 20, 2008 and for 0.2.0.29-rc released on July 8, 2008. It was introduced in Tor with version 0.0.9.x released in 2005.

1.4 Disregarding Predefined Set of Rendezvous Points

When setting up the measurement environment, a bug was discovered considering the `RendNodes` configuration option. Using this configuration option the user can suggest specific Tor relays to be selected as rendezvous node by providing a list of nicknames or identifiers. Since the rendezvous point is usually established using cannibalization, there must be an existing circuit available with the desired rendezvous point at the end. Otherwise the configuration is simply ignored, i.e., a new circuit to the rendezvous point is not build. This bug was introduced with the configuration option `RendNodes` itself in version 0.0.6pre1 that was released on April 8, 2004. The bug was reported as task 754 in the bug tracker². The fix is to mark the affected configuration options obsolete. It is included in 0.2.1.3-alpha released on August 3, 2008.

1.5 Replacement for 1-Second Circuit Creation Loop

The current implementation of a client accessing a hidden service makes use of a 1-second loop that checks whether circuits have been successfully established. It has turned out that this bug affected two places in the code: waiting for a client-side rendezvous circuit to be established and for a `RENDEZVOUS2` cell to be received. This leads to a maximum performance gain of 2 seconds instead when the client could already proceed in connection establishment. This can be avoided by processing the establishment of a circuit immediately. This bug has been reported as task 743³ and fixed for 0.2.1.3-alpha released on August 3, 2008. It is marked as backport for 0.2.0.x as soon as it is tested more.

1.6 Inaccuracies in Descriptor Upload Logic

The logic to decide whether a descriptor should be uploaded needed a reworking. Unrelated events like giving up on an introduction point candidate reset the internal 30-seconds timer. The result was an unexpected exceedance of the

²<http://bugs.noreply.org/flyspray/?do=details&id=754>

³<http://bugs.noreply.org/flyspray/?do=details&id=743>

timer. The timer should only be reset when changes to the set of established introduction points would also affect the last published descriptor. This bug is described in task 763⁴. The fix is included in 0.2.1.3-alpha, released on August 3, 2008. The bug was introduced in version 0.0.9.3 released on January 21, 2005.

1.7 Descriptor Upload Failures

The current logic to upload rendezvous service descriptors does not handle failures in a reasonable way. In case of a failure, Tor waits for a solid hour before making the next attempt. The bug here is that Tor did not have the required router descriptors to upload rendezvous descriptors and did not attempt to download the missing router descriptor. An evaluation of log statements has shown that 14.7% of all descriptor uploads to the three central hidden service authorities failed. A subsequent analysis has confirmed these results for the distributed storage for hidden service descriptors with a failure rate of 13.7%. When attempting to contact a hidden service directory, Tor should notice at the latest that a router descriptor is missing and initiate the download. This bug is described in detail in task 767⁵.

2 Proposed Design Changes

The analysis has further brought up numerous possible design changes to improve performance of Tor Hidden Services. Some of these design changes have side-effects on anonymity or overall network load which had to be weighed up against individual performance gains. A discussion of seven possible design changes⁶ has lead to a selection of four changes that are presented here and will be implemented in the upcoming two months.

2.1 Shorter Circuit Extension Timeout

When establishing a connection to a hidden service a client cannibalizes an existing circuit and extends it by one hop to one of the service's introduction points. In most cases this can be accomplished within a few seconds (cf. Figure 1). Therefore, the current timeout of 60 seconds for extending a circuit is far too high. Assuming that the timeout would be reduced to a lower value, e.g., 30 seconds, a second (or third) attempt to cannibalize and extend would be started earlier. Figure 1 contains a histogram of times that are required for this task. Here, the current timeout of 60 seconds becomes visible at the second peak of values shortly after 60 seconds. Figure 2 visualizes the fraction of successfully extended introduction circuits for given timeouts. With the current timeout of 60 seconds, 93.42% of all circuits can be established, whereas this

⁴<http://bugs.noreply.org/flyspray/?do=details&id=763>

⁵<http://bugs.noreply.org/flyspray/?do=details&id=767>

⁶<http://freehaven.net/~karsten/hidserv/discussion-2008-07-15.pdf>

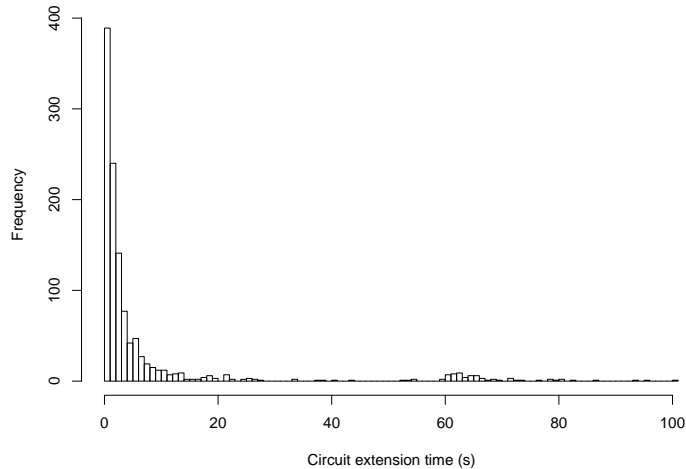


Figure 1: Client-side introduction circuit establishment times

fraction would have been only 0.87% smaller at 92.55% with a timeout of 30 seconds.

Figure 3 shows simulated mean introduction circuit cannibalization and extension times for timeouts between 10 and 60 seconds. For a timeout of, e.g., 30 seconds the performance gain would be approximately 2 seconds in the mean as opposed to the current timeout of 60 seconds. At the same time a smaller timeout leads to discarding an increasing number of circuits that might have been completed within the current timeout of 60 seconds. Figure 4 visualizes the simulated increased number of circuit cannibalization and extension attempts for lower timeouts.

Measurements with simulated low-bandwidth connectivity have shown that there is no significant effect of client connectivity on circuit extension times. The reason for this might be that extension messages are small and thereby independent of the client bandwidth. Further, the connection between client and entry node only constitutes a single hop of a circuit, so that its influence on the whole circuit is limited.

The exact value of the new timeout does not necessarily have to be 30 seconds, but might also depend on the results of circuit build timeout measurements as described in proposal 151⁷.

⁷<https://tor-svn.freehaven.net/svn/tor/trunk/doc/spec/proposals/151-path-selection-improvements.txt>

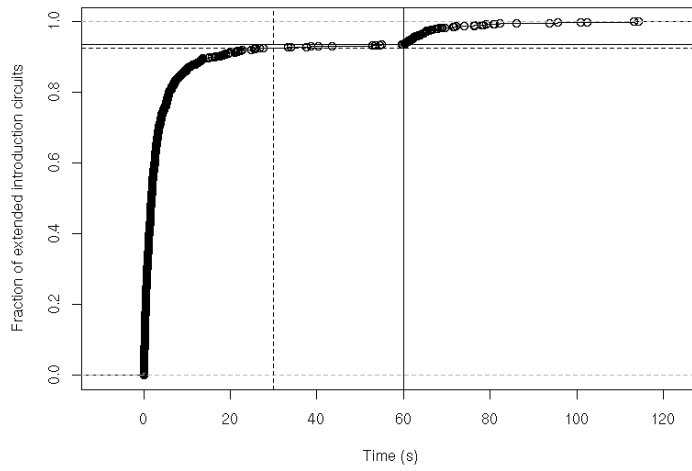


Figure 2: Empirical cumulative distribution function of introduction circuit extensions.

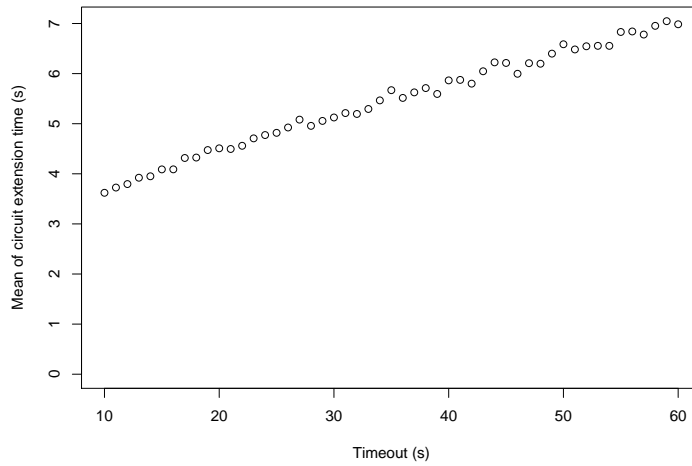


Figure 3: Mean client-side introduction circuit establishment times as a function of timeout

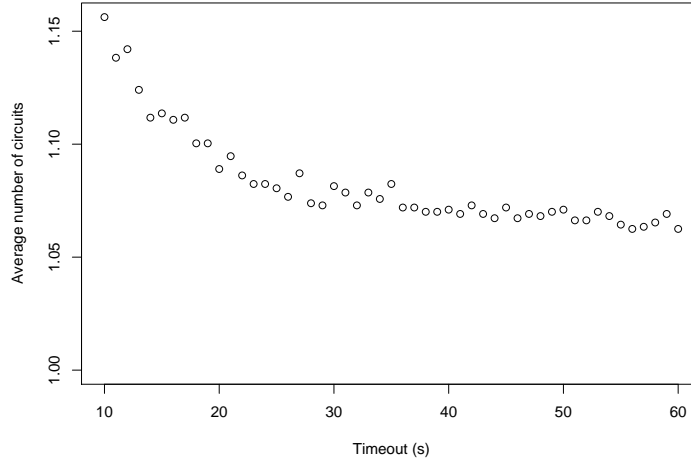


Figure 4: Number of client-side introduction circuit establishment attempts as a function of timeout

2.2 Parallel Connections to Introduction Points

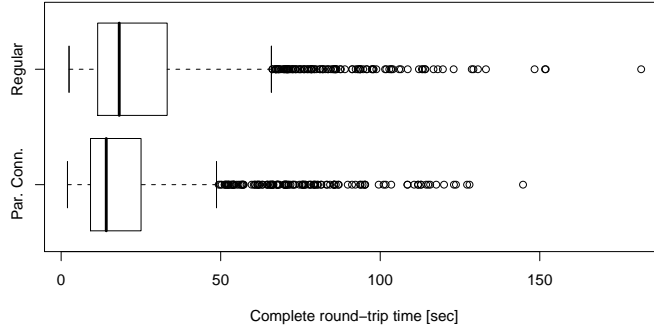
An additional approach to accelerate extension of introduction circuits is to extend a second circuit in parallel to a different introduction point. Such parallel extension attempts should be started after a short delay of, e.g., 15 seconds in order to prevent unnecessary circuit extensions and thereby save network resources. Whichever circuit extension succeeds first is used for introduction, while the other attempt is aborted.

An evaluation has been performed for the more resource-intensive approach of starting two parallel circuits *immediately* instead of waiting for a short delay. The result was a reduction of connection establishment times from 27.4 seconds in the original protocol to 22.5 seconds. Figure 5 shows the possible improvement of connection establishment times.

While the effect of the proposed approach of delayed parallelization on mean connection establishment times is expected to be smaller, variability of connection attempt times can be reduced significantly.

2.3 Increase Count of Internal Circuits

Hidden services need to create or cannibalize and extend a circuit to a rendezvous point for every client request. Really popular hidden services require more than two internal circuits in the pool to answer multiple client requests at the same time. This scenario was not yet analyzed, but will probably exhibit worse performance than measured in the previous analysis. The number of



Protocol	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
Regular	2.479	11.450	18.210	27.370	33.220	181.800
Par. Conn.	1.987	9.212	14.140	22.450	24.990	144.800

Figure 5: Connection establishment times for parallel connections to introduction points

Table 1: Introduction circuit creation times

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
0.139	1.203	2.504	4.659	5.234	56.090

preemptively built internal circuits should be a function of connection requests in the past to adapt to changing needs. Furthermore, an increased number of internal circuits on client side would allow clients to establish connections to more than one hidden service at a time.

Under the assumption that a popular hidden service cannot make use of cannibalization for connecting to rendezvous points, the circuit creation time needs to be added to the current results. An evaluation of internal circuit creating times is shown in Table 2.3. In the mean the connection establishment time to a popular hidden service would increase by 4.7 seconds.

2.4 Build More Introduction Circuits

When establishing introduction points, a hidden service should launch 5 instead of 3 introduction circuits at the same time and use only the first 3 that could be established. The remaining two circuits could still be used for other purposes afterwards.

The effect has been simulated using previously measured data, too. Therefore, circuit establishment times were derived from log files and written to an array. Afterwards, a simulation with 10,000 runs was performed picking 5 (4, 6) random values and using the 3 lowest values in contrast to picking only 3 values at random. The result is that the mean time of the 3-out-of-3 approach

Table 2: Establishment times for the first 3 out of n introduction circuits

Intro. Pnts.	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
3	1.023	5.017	6.820	8.060	9.718	40.490
4	0.882	3.811	4.997	5.421	6.472	30.950
5	0.786	3.211	4.188	4.430	5.367	19.230
6	0.578	2.826	3.702	3.865	4.705	12.080

is 8.1 seconds, while the mean time of the 3-out-of-5 approach is 4.4 seconds. The results are given in Table 2.4.

The effect on network load is minimal, because the hidden service can reuse the slower internal circuits for other purposes, e.g., rendezvous circuits. The only change is that a hidden service starts establishing more circuits at once instead of subsequently doing so.

3 Deferred Design Changes

Part of the discussion resulted in the decision to defer some design changes for certain reasons. They are described below together with the reasons for deferring them.

3.1 Rendezvous Protocol Simplifications

Overlier and Syverson proposed two simplified rendezvous protocols.⁸ Their first protocol aims at using a single circuit on client-side to contain the rendezvous point and connect to an introduction point. The second protocol goes the extra mile to unite the roles of introduction point and rendezvous point and save another circuit. It can be assumed that these simplifications improve connection establishment times. A possible design of the second protocol that combines the roles of introduction and rendezvous point has been described in proposal 142⁹.

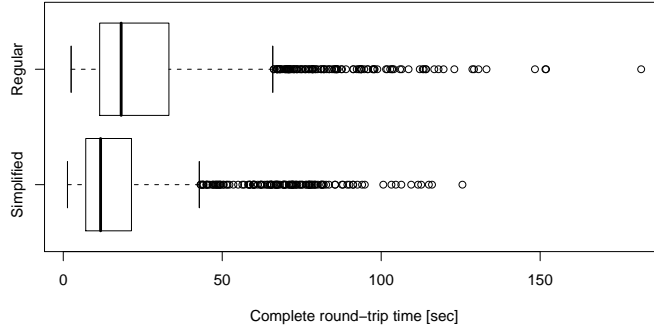
A pre-evaluation¹⁰ of this design change has shown that mean connection establishment times can be reduced from 27.4 seconds in the original protocol to 19.1 seconds in the changed design. Figure 6 shows a comparison of connection establishment times.

One of the original reasons for the separation of introduction and rendezvous points was that a relay shall not be made responsible for relaying data on behalf of a certain hidden service. The changed design needs to make sure that a combined introduction and rendezvous points cannot learn easily on behalf of

⁸Lasse Overlier and Paul Syverson, Improving Efficiency and Simplicity of Tor Circuit Establishment and Hidden Services, <http://www.freehaven.net/anonbib/#overlier-pet2007>

⁹<https://tor-svn.freehaven.net/svn/tor/trunk/doc/spec/proposals/142-combine-intro-and-rend-points.txt>

¹⁰Christian Wilms, Improving the Tor Hidden Service Protocol Aiming at Better Performance, diploma thesis, June 2008



Protocol	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
Regular	2.479	11.450	18.210	27.370	33.220	181.800
Simplified	1.324	7.070	11.740	19.130	21.450	125.600

Figure 6: Connection establishment times for combining introduction and rendezvous points

which hidden service it is working. However, there are still open questions to liability that require more discussion.

A further problem is that an attacker who can take down a combined introduction and rendezvous point does not only eliminate an access point to the hidden service, but also breaks current client connections to the hidden service using that contact point.

Indirect traffic analysis is another issue of this approach. An attack that was originally intended to identify Tor clients¹¹ could be applied to attack hidden services. Firstly the attack by Murdoch and Danezis¹² could be used to identify all the Tor nodes on the path. Then from each node the attacker could try to identify the hidden service using an Internet map and measuring link congestion from multiple vantage points. The attacker can only load the data channel. The combination of introduction and rendezvous point changes the characteristics. Previously there were 3 hops, and a chosen rendezvous point; with the change, there are two hops and a known combined introduction and rendezvous point. When applying the attack above, there are now only two candidates for the entry node, rather than three. Similarly, for an attacker that wants to get the connection logs from all possible entry nodes, there are now only two, rather than three nodes.

The implementation of this design change is not straightforward. The circuit between hidden service and combined introduction and rendezvous point would need to transfer data for multiple clients. This is not supported by the current Tor implementation and would be a major design change. An alternative would be to pre-emptively establish circuits between hidden service and combined

¹¹<http://web.crypto.cs.sunysb.edu/spday/presentations/Angelos.Keromytis.pdf>

¹²Murdoch, Danezis, Low-Cost Traffic Analysis of Tor, IEEE Symposium on Security and Privacy, May 2005.

introduction and rendezvous point, which, however, would only be half way of a solution.

This design change requires relays in the middle of the Tor network to upgrade. From the experience with deploying the distributed hidden service directory this might take quite some time.

In summary there are too many issues to include this design change within a schedule of the next months.

3.2 Descriptor Upload Timing

The choice to wait for 30 seconds for a service to have a stable set of introduction points is rather arbitrary. An analysis of typical delays in establishing introduction points might help to apply a more suitable algorithm here. When finding a useful algorithm for uploading rendezvous descriptors, there are two conflictive objectives: a) upload descriptors as early as possible and b) avoid uploading descriptors that need to be replaced shortly after.

An evaluation of introduction point establishments and abandonings within the first 30 minutes of providing a hidden service can help determining the optimal delay. One can determine the number of uploaded descriptors from empirical data for every possible delay. Figure 7 shows the delays in descriptor publication for stabilization times between 1 and 90 seconds. Figure 8 displays the mean number of published descriptors.

These results show that there is a clear trade-off between the two objectives for delays lower than 50 seconds. A delay of 60 seconds significantly reduces the mean number of published descriptors, but at the price of almost doubling the mean upload time of the first descriptor as compared to the current choice of 30 seconds.

It might be useful to re-evaluate descriptor upload timing once the other design changes are in place. For the time being the 30-seconds delay appears to be a good choice.

4 Future Evaluations

Some evaluations have turned out to require more time than expected and had to be scheduled for a later time in the project. The current plan is to perform these evaluations within the timeframe until January 15, 2009 and work with assumptions until final results are available.

4.1 Low-Bandwidth Measurements

For some improvement suggestions, e.g. reducing timeouts, the effect on clients with low bandwidth is yet unclear. Future measurements should therefore include clients and possibly hidden servers on (real rather than simulated) low-bandwidth Internet connections.

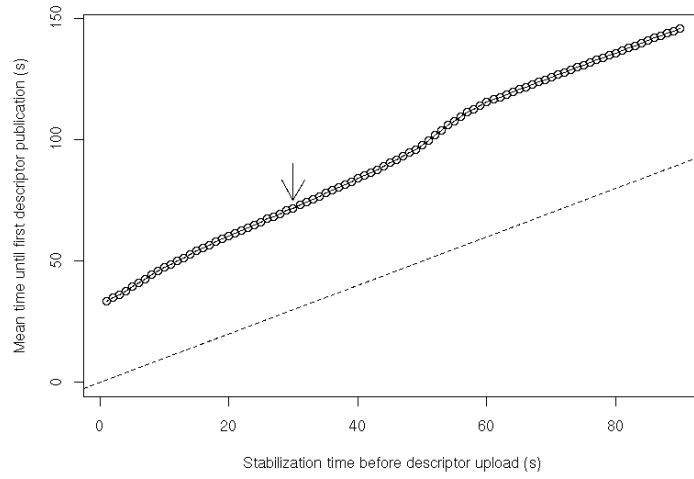


Figure 7: Time until first descriptor upload as a function of stabilization time (arrow denotes current stabilization time of 30 seconds)

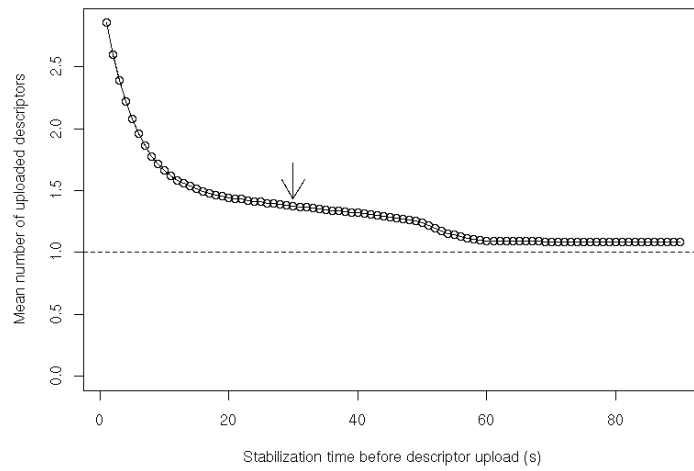


Figure 8: Number of uploaded descriptors as a function of stabilization time (arrow denotes current stabilization time of 30 seconds)

Performing low-bandwidth measurements is out of scope of the current list of performance improvements. However, learning about how low-bandwidth clients experience hidden service usage might be valuable for future improvements.

4.2 Grand Scaling Plan

There are other attempts to make Tor more useful for low-bandwidth clients. Part of this project might be that clients do not download the complete set of router descriptors, but request descriptors during the process of circuit establishment. Alas this would slow down circuit creation even more, so that hidden services should even try harder to avoid on-demand circuit creation.

Before being able to estimate the effects of requesting router descriptors on demand, these changes need to be implemented and included in the Tor code. An evaluation can then show possible improvements for hidden services to adapt to these changes. This evaluation should also be performed until January 15, 2009.