Performance of Tor

Paul Buder, Daniel Heyne, and Martin Peter Stenzel

Technische Universität Darmstadt,
Theoretische Informatik - Kryptographie und Computeralgebra,
Cryptography, Privacy and Security
Karolinenplatz 5, 64289 Darmstadt, Germany
paulbuder@gmx.de,danielheyne@gmx.net,martinstenzel@gmx.de
https://www.cdc.informatik.tu-darmstadt.de/cdc/

Abstract. Tor is a popular decentralized anonymity-enhancing network that helps users to improve their privacy. At first we give a short introduction into Tor. We then describe the mechanisms that are implemented and how they work. Furthermore we describe performance limits and then show possibilities to improve the performance of Tor. Afterwards we compare Tor to other anonymity-enhancing networks. Finally we give a conclusion.

Keywords: Tor, Performance, Anonymity Network, Onion Routing

1 Introduction

Tor is an anonymity network that preserves users' online privacy and security by utilizing a distributed overlay network. This is achieved by making use of virtual tunnels through wich traffic is routed and thus concealed. Therefore Tor makes it possible for two peers to communicate with each other over the internet without reavealing transmitted data or any information on who is communicating with whom. Hence traffic analysis and network surveillance can be prevented by the mechanisms of Tor. The Tor Software is developed and distributed under the Berkley Software Distribution license and thus open-source. The name is an acronym for ***T****he* ***O****nion* ***R****outer* referring to different layers of encryption which are nested like the layers of an onion.

1.1 Structure

The tor network differentiates between different kinds of components including directory servers and tor nodes respektively routers or onion routers. Tor nodes can operate as entry points, realys and exit points. Directory servers are used to advertise tor nodes which are assumed to be trusted and available. For that reason they provide signed directories in which they store the description and current state of the routers. Directory servers are queried by new tor users prior to joining the network in order to acquire necessary information concerning these trusted nodes.

1.2 Circuit Construction

The construction of a circuit is a fundamental process performed by an onion router for the purpose of creating anonymous links. Once a node has successfully obtained a list of tor nodes from a directory server it starts incementally building a circuit of tor relays. The circuit is extended in a hop-to- hop manner so that each relay only knows about its predecessor and successor and therefor has no knowledge of the complete path. In the process of creating a pathway the next hop is always selected randomly with the result that no observer at any single point is able to determine the origin and destination of the data packets. Once the circuit has been constructed it can be used to transmit TCP traffic to the destination and vice versa. The communication process is illustrated below in figure 1.

As a result of this approach an eavesdropper needs to control all three relay nodes in order to obtain information on who is communicating with whom. Since this is rather unlikely considering that many tor relays are operated by regular people it can be assumed that a communication via such a relay circuit is secure.

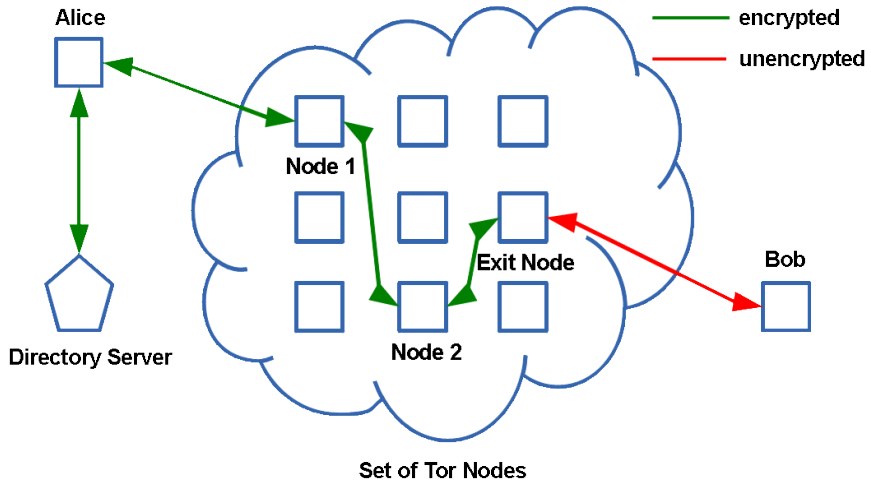

Fig. 1. Alice is communicating with Bob over the tor network after building a circuit of tor relays.

1.3 Hidden Services

Hidden Services are servers that accept inbound connections only via the tor network. It is therefor possible to provide anonymity to those servers. To conceal the servers identity its IP address remains unknown, instead the service can be accessed only through its .onion address. Tor relays are able to use these addresses to route traffic to and from servers that provide hidden services while providing anonymity to both sides.

In detail (comp. 2), at first the hidden service provider picky up some introduction points and build circuits to them. Then he advertises his hidden service, introduction points and public key at a database server. Links to hidden services are collected in a directional service called *The Hidden Wiki*. So now a user can ask the database server for a specific *.onion* address and gets more information about this service. Also he set up a rendezvous point. After that the user sends a encrypted message (public key from the service) to the hidden service over one introduction point and asks for a introduction point. The hidden service connects to the users rendezvous point and there negotiate the communication channel. It is even possible to bypass network address translaters and firewalls.

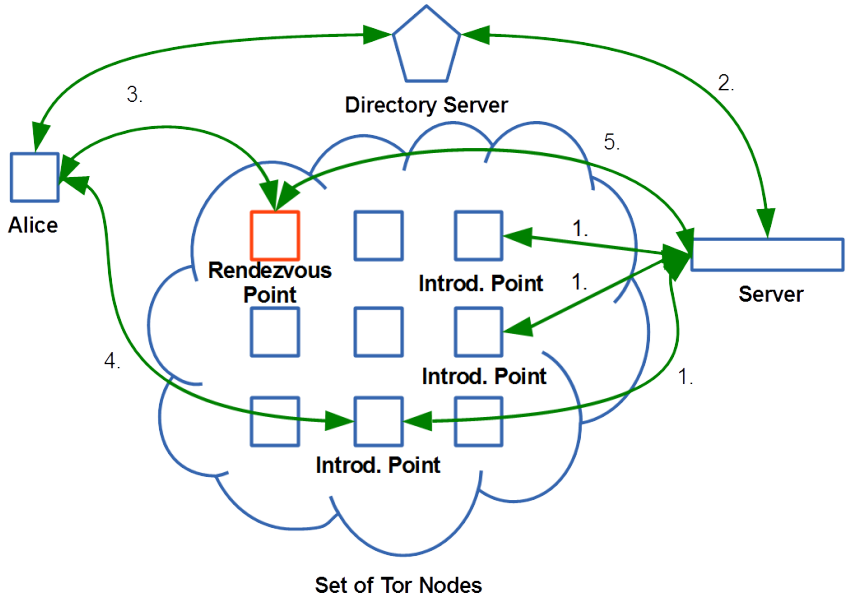

Fig. 2. Set up and connect to a hidden service

A well-known example of a hidden service is *silk road*. Silk road was an e-commerce platform which turned out to be a black market. It has been used to trade illegal goods without the possibility of surveillance by prosecuting authorities. Payments were made using bitcoins. This in combination with the use of the tor network should provide anonymity to customers and sellers.

1.4 Advantages and Disadvantages

Tor has a lot of advantages for its users. It is for example possible to surf the web anonymously, prevent websites from tracking oneself or accessing locally blocked content. Those features also help protecting whistleblowers and avoiding traffic analysis. While these advantages are not far to seek there do exist some disadvantages.

One of the biggest disadvantages concerning the usability is the speed issue with tor. Due to the fact that all traffic is encrypted and transmitted through three tor relays which may be located around the world, tor can be quite slow. This is because every relay can be seen as a bottle neck in the circuit and therefor limits the data rate. This problem intensifies when one relay server is part in several circuits and thus has to forward multiple data streams.

Furthermore due to tors centralized nature the network is vulnerable to distributes denial of service attacks. The directory servers are needed to obtain information about tor relays in the network. An adversary could try to disable one or more directory servers by generating a traffic overload. This would lead to the problem that these servers cannot be queried by tor users anymore and therefor incapacitates them from creating new circuits. Additionally depending on the severity of the attack it may become impossible for new users to join the tor network.

Another negative aspect is that the tor network can be abused to shroud illegal activities like shown by the example in section 1.3. This involves the danger that cyber criminals can operate anonymously without the possibility for prosecuting authorities control their activities.

Finally the advantages and disadvantages are shown in table 1.4.

Advantages	**Disadvantages**
anonymously webbrowsing	slow speed
prevent tracking	usability
access to locally blocked content	centralized structure (directory server)
protect whistleblowers	DDoD attacks possible
avoiding traffic analysis	shroud illegal activities

Table 1. Overview of the advantages and disadvantages

2 Performance

With the ongoing growth of the tor network in the past few years its performance has suffered. This has among other things lead to high latencys resulting in inconvenient delays when requesting websites. These performance issues are caused by a number of reasons.

One problem is that the congestion control implemented in tor does not work well which negatively affects the co-existence of quiet streams and bulk transfers. Additinally there are many users who contribute far less in comparison to what they consume in terms of network resources. This results in a high utilization of the tor network which cannot be handled by the existing tor relays. Furthermore the path selection algorithms implemented in tor do not distribute network load correctly and therefor do not utilize the full capacity of some nodes while overloading others [4].

2.1 State of the Art

A good path selection algorithm for a perfect anonymity would select the relays from a set of all active nodes. So a attacker cannot influence the path selection, except by having more relays. This path selection algorithm implies a poor performance, because the probability to choose a weak or a good node is the same. There is a trade-off between selecting nodes for optimal performance and providing maximum anonymity protection. In reality the client gets the list from the directory server and a self advertised not trustworthy bandwidth information about every node. However the client collects the information to establish a connection. By default they maintain a list of 3 nodes that are fast, stable and have a long uptime. The nodes are chosen with a probability proportional to their bandwidths.

2.2 Tor's Congestion Control

To avoid congestions Tor mainly uses two mechanisms. One is the TCP flow control and the other is a windowing scheme per circuit. In the paper by Dingledine and Murdoch [4] it is suggested to chose another size for the maximum unacknowledged cells.

TCP's flow control aims to prevent the sender from transmitting data too fast to the receiver in order to ensure that the receiver is able to proccess all data received. Such a mechanism is crucial in networks where routers differ from each other in their communication speed. To achieve this goal TCP uses the sliding window flow control protocol which will now be briefly explained. First the receiver has to indicate the window size respectivly the amount of data he is willing to buffer. The sender can now only transmit as much data as he can fit into the remaining window. He then has to wait for the receiver to send an acknowledgement. This indicates that the receiver has successfully received all packets with sequence numbers up to the acknowledgement sequence number

and thus has freed buffer space and also shifted the window. To visualize this process it is illustrated below in figure 3.

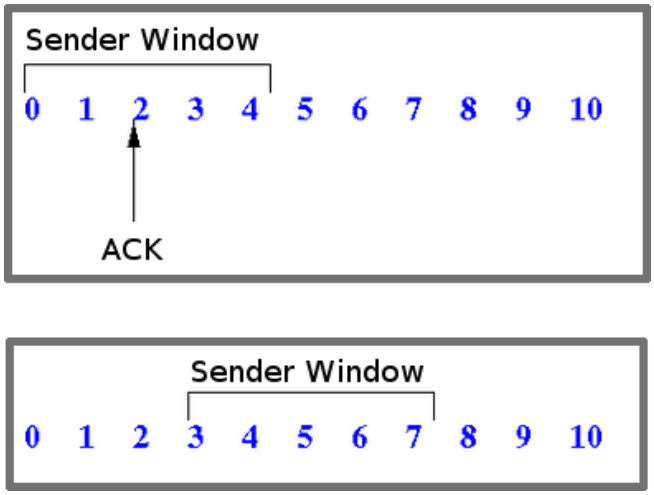

Fig. 3. Each time the sender receives an acknowledgement from the receiver he updates his window. As illustrated the receiver acknowledges all packets with sequence numbers less than or equal to the acknowledgement sequence number.

Tor deploys such an sliding window mechanism as described above. In this context Dingledine and Murdoch [4] recommended to reduce the window size. Therefore relaying nodes have to buffer less data per connection and are thus able to maintain more connections. Additionally the buffered data can be processed faster by the relaying node. Currently the window size utilized by tor nodes is 512kb. Kiraly [11] elaborated that the best size for fixed network latency an relay bandwidth would be 100kb in case all nodes would use this value.

2.3 Improving the Path Selection in Tor

Since the data in Tor network has to overcome several hops to reach its destination, some experiments on the path selection of Tor were made [2]. To increase the throughput of traffic in the network, three aspects are taken into account. The first thing is the geographical distance between the routers int the network. For testing the impact of geographic diversity on the data rate a different path search algorithm was implemented. The distance between the routers can be calculated by using the geographical coordinates. The evaluation of this experiment shows that selecting a path with routers, located within a certain range to each other, increases the data rate. As it is shown in figure 4, the throughput decreases when the distance between two hops is around 4000 miles.

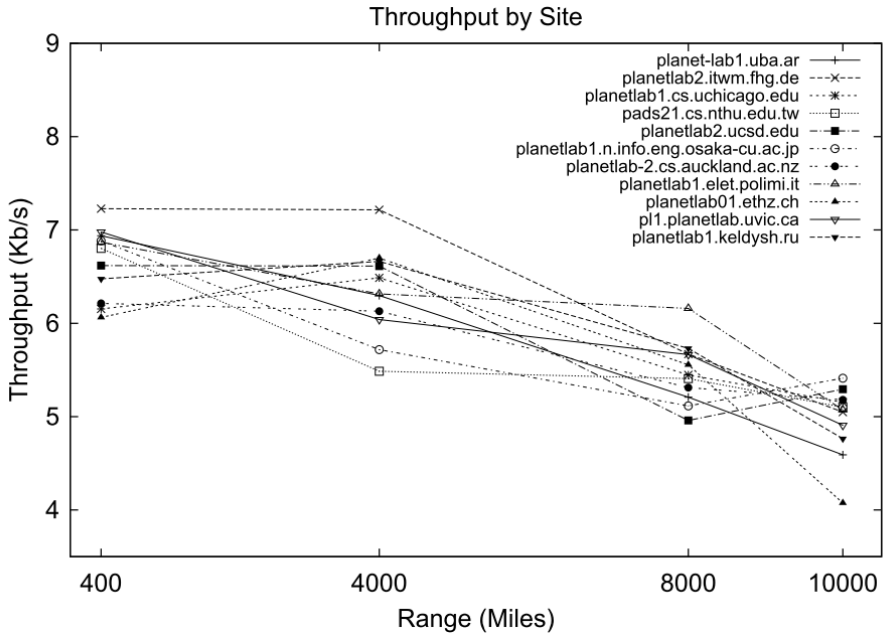

Fig. 4. Average throughput while restricting geographic distance between routers [7]

Second experiment is about the varying of router performance flags. The flags for every router in Tor are stored by the directory servers. They give information about the uptime and the observed width of the router. Several combinations of router flags were tested and the result was likely the same for every combination except for router without any given flags. For them the throughput was lower. The result is also shown in figure 5 The last approach is about varying the number of hops. As described Tor uses three hops for a connection by default. For thee experiment connections with two, three and four hops were tested. The

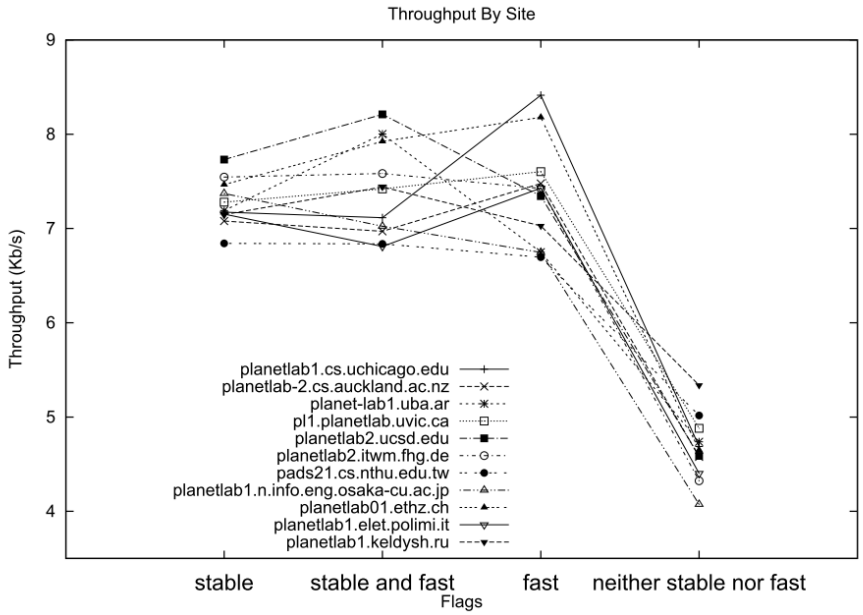

Fig. 5. Average throughput while varying performance flags [7]

result was that a increasing number of hops reduces the data rate and decreases the reliability as it is shown in figure 6.

2.4 Multipath circuit construction and stream splitting

Tor mainly aims to provide its users with sufficient bandwidth for low-latency real-time interactive applications. However Tor somewhat lacks the handling of congestion and high loads which ultimately leads to a poor user experience. Such high loads are caused by peer-to-peer downloaders. This in combination with Tors path selection algorithm which is not optimally load balancing the generated traffic can lead to congestion. However the following aspects that can be improved using multipath routing:

- *Improved load balancing.* During a transmission a tor relay may become unable to handle the amount of traffic that has been forwarded to it. In this case splitting traffic across different paths can help the over-utilized router to cope with the load. However circuits need one shared exit node.
- *Increase throughput.* Because data can be splitted over multiple relaying nodes the throughput of one circuit can be increased up to the throughput of the exit node. This is especially useful when the entry node and the second node are low-bandwidth routers.

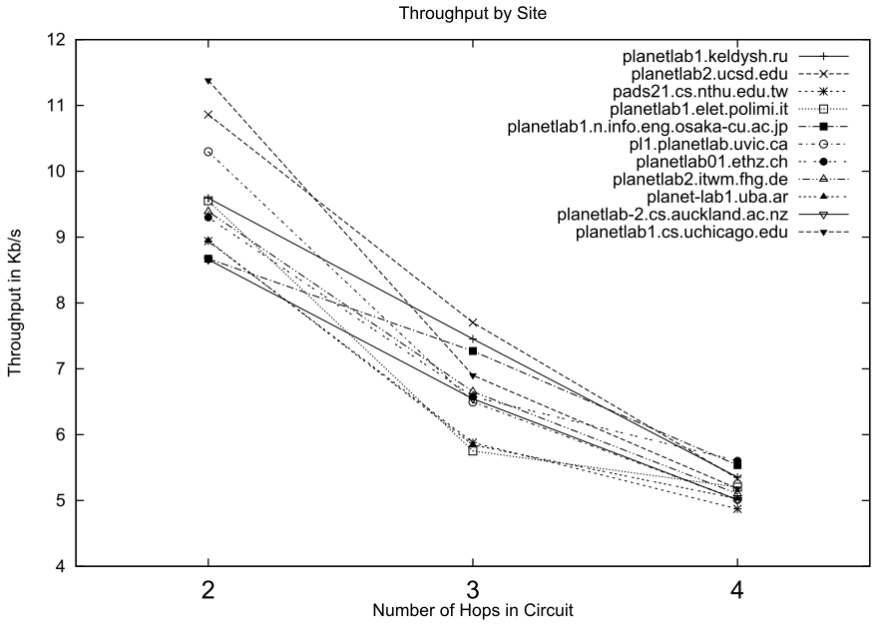

Fig. 6. Average throughput while varying number of hops [7]

– *Improve performance with low-bandwith relays.* To achieve a high throughput Tor prefers routers which supply a high bandwidth. By combining multiple circuits containing low-bandwidth nodes the throughput which can be achieved increases.

Concerning this problem [5] proposes an interesting approach. The Strategy here is to overcome bottlenecks in the Tor network using multipaths. Therefor a traffic splitting algorithm was implemented, tested and evaluated. The traffic splitting is divided in three parts. The first one is the construction of the multipath. As shown in figure 7 a primary circuit is constructed. This happens according to Tor router selection algorithm. If needed the algorithm computes a second circuit. The important thing about is that both paths share the same exit point. Entry points have to be different to compute another path.

In the second part of the algorithm will be split depending on the observed throughput of the network. So here a dynamic load balancing algorithm was implemented to react on the networks behavior as well as to react on the congestion state of a circuit.

The second part is followed by sequencing, buffering and recording part. This part is needed because data is send via several circuits and the receiver must be

able to record all of this data correctly on its arrival. Therefore Tor cell format was changed. The first 4 bytes now contain a circuit ID so the payload of the cells is reduced by 1%. Since the traffic is split into several streams the routers and hops are responsible for recording and buffering the traffic.

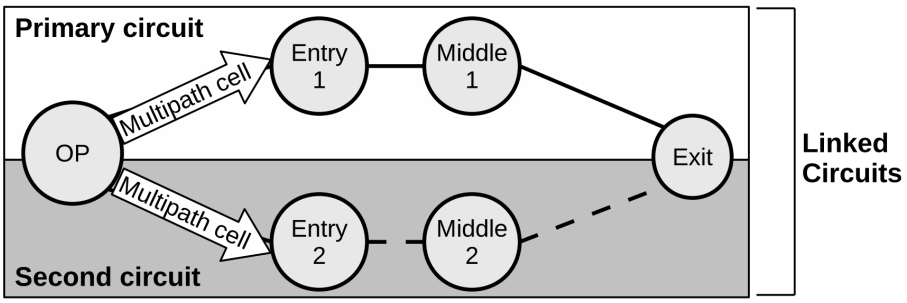

Fig. 7. Multipath routing example [5]

2.5 Real-time Traffic Classification

This approach is about defining classes of service for Tor's traffic. With the definition of different classes, each application should be mapped to its Quality of Service (QoS) requirement. For example it allows relays to throttle one application class over another. In [6] a framework *DiffTor* is introduced, which enables differentiated QoS using traffic classification in the anonymous communication in Tor. Therefore different attributes for the classification are described The circuits for different applications have different lifetimes in Tor. For example browsing circuits exceeds earlier than streaming or BitTorrent downloads. Also the transferred amount of data is different for interactive circuits, e.g. browsing transfers less data than streaming. Last for example a BitTorrent download causes bidirectional traffic and builds many idle circuits.

In [6] there is a defintion of three classes introduced to classify the traffic:

- *Bulk transfer*: The class contains applications, which transfer large amounts of data. They have no time constraints and will try to use as much available bandwidth as possible.
- *Interactive*: These applications require an interaction between a client and a server. Furthermore there are also time sensitive. The main application focused on is web browsing.
- *Streaming*: This type of application is non-interactive and bulk transferring. Therefore time and quality constraints are required.

Based on the attributes several algorithms where implemented to classify the traffic. There is the Naive Bayes classifier, Baysian nets and Decision Trees. All of them where tested in different experiments including a live Tor experiment. The result is, that the Bayes nets and the Naive Bayes classifier are accurate algorithms. Also the performance increases with the traffic classification. The download time can be improved up to 86% as the figure 8 shows.

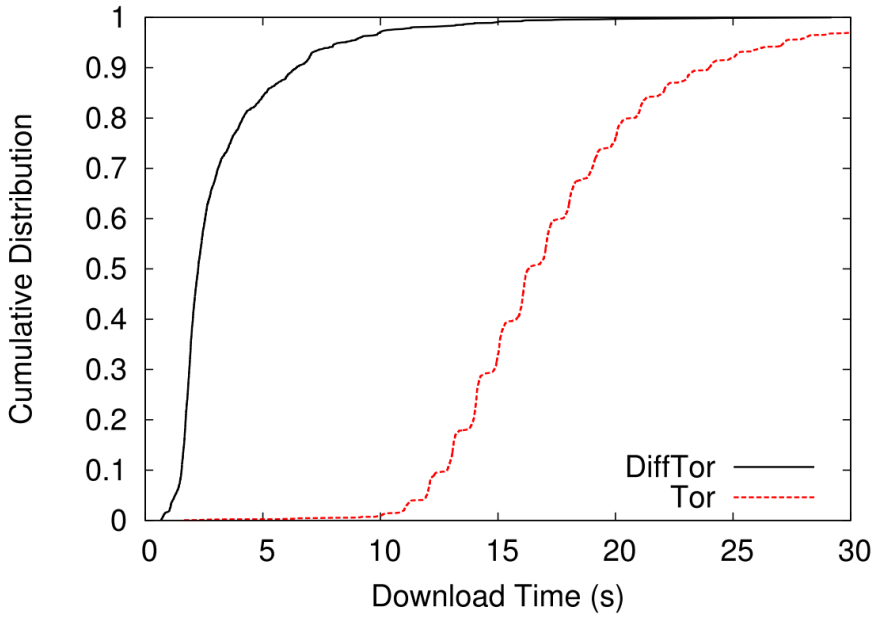

Fig. 8. Real-time Traffic performance [6]

2.6 Tor compared with other anonymous networks

Today there are some other known, anonymous communications networks existing like Freenet, GNUnet, I2P, JonDo and RetroShare. All of them have the same goal to communicate anonymous. These systems will be briefly described and compared to Tor.

Freenet is an anonymous, distributed, peer to peer publishing network. It gives secure ways to store data. There are further applications existing on top of *Freenet*, so that it is possible to provide static websites or message services. Compared to Tor, *Freenet* is designed and optimized for hidden services.

GNUnet is an anonymous, distributed, peer to peer network for file sharing. Every peer in the network monitors and evaluates the other participant's behavior and interactions. The performance of each user depends on the evaluation by other users in the network and is feasible to detect bad participant's.

I2P (Invisible Internet Project) is an anonymous and pseudonymous network. It has no directory server. Therefore information about the participants are stored in a fully distributed database (a modified distributed hash table). In contrast to Tor every node in the I2P network can be a exit point, also I2P was invented as a network that acts like a *Darknet*. The design is optimized for hidden services, which are much faster than in Tor.

JonDo is a proxy system to get anonymous access to the web. A user gets a list of possible mix cascades from a trusted server. He connect to the internet over two (free version) or tree (premium version) mix cascades to disguise his identity. The system is so more centralized and based on fixed mix cascades. In contrast to Tor there is a premium version of JonDo existing to increase the speed, the numbers of available ports and the file size for up- and downloads. The optimum bandwidth of the free version of JonDo is much inferior than Tor, at around 50 kb/s.

RetroShare is a distributed friend-to-friend communication protocol for chat, mail, forum, voice-over-ip and file sharing. Depending on the settings of each client, it is possible to share data either with friends or with friends and their friends or with friends and their friends and further related friendships. The performance and available services of every participant depends on the quantity and quality of their own friends.

Summary For all anonymous networks the user need a specific software to join the system. But anonymous networks have a different focus in which they are specialized e.g.: hidden services, file sharing, anonymous web browsing. From these specialized priorities there arise various structures: centralized, decentralized or hybrid. The anonymous networks are good in their special field and most of them have a good trade-off between performance and anonymity. Tor is a open source solution to browse anonymous on the internet with a acceptable bandwidth. JonDo is most similar to Tor but for a good performance a user must pay for the premium service.

2.7 Tor compared to the Internet

Tor operates as an overlay network on top of the internet providing additional features to its users. Thus tor also changes the users perception for example when browsing the web. However the utilization of such an overlay network varies quite a lot from the conventional use of the internet.

One of the biggest differences is that tor can provide anonymity using particular routing techniques. This is a feature which cannot be provided by the internet due to its structure. Users are identifiable by their internet addresses and therefor their traffic can be tracked and analyzed.

Furthermore the internet primarily uses the two transport layer protocols UDP and TCP to transfer data from one host to another. However when using tor it is not possible to use UDP because only TCP is supported. This makes it for example impossible to send particularly crafted packets via tor. Thus attacks like denial of service cannot be masqueraded by tor.

Finally the most crucial distinctions are compared in table 2.7.

Properties	**TOR**	**Internet**
bandwidth	low	high
traffic analysis	nearly impossible	possible
protection against website profiling	10-minute possibility	possible
availability	depends on the nodes	very good
usability	limited, because of bandwidth and initial steps	good
non-repudiation	not likely	is given by provider logs
anonymity	possible	limited
routing path	changed every 10-minute	depends on provider routing algorithm
confidentiality	random distribution	individual provider

Table 2. Compared the properties between TOR and the Internet

3 Conclusion

In conclusion you can say that Tor is one of the most common network to have anonymous access to the web and it has big amount of researching topics not only when it comes to performance. Routers and the path selection have probably the biggest impact on the performance because not every router is capable as well as another one.
In this paper we learned about several strategies for getting higher data rates in the Tor network by changing things in the congestion control and several path selection strategies as well as a multipath construction strategy to overcome bottlenecks.

In the future there surely will be some work on the performance of Tor to get higher number users and greater anonymity properties.

References

1. Snader, R., Nikita, B., A Tune-up for Tor: Improving Security and Performance in the Tor Network, 2008, http://www.freehaven.net/anonbib/cache/snader08.pdf
2. Chen, F., Pasquale, J., Toward Improving Path Selection in Tor, 2010, cseweb.ucsd.edu/users/pasquale/Papers/globecom10c.pdf
3. Johnson, N., Raja, G., Improving Security And Performance In The Network Through Opportunistic Bandwidth Measurement Mechanism, 2012, www.ijcae.org/admin/journals/Journals25.pdf
4. Dingledine, R., Murdoch S., Performance Improvements on Tor or, Why Tor is slow and what we're going to do about it, 2009, https://www.torproject.org/press/presskit/2009-03-11-performance.pdf
5. AlSabah, M., Bauer, K., Elahi, T., Goldberg, I., The Path Less Travelled: Overcoming Tor?s Bottlenecks with Multipaths, 2011, http://cacr.uwaterloo.ca/techreports/2011/cacr2011-29.pdf
6. AlSabah, M., Bauer, K., Goldberg, I., Enhancing Tor's Performance using Real-time Traffic Classification, 2012, http://www.cypherpunks.ca/~iang/pubs/difftor-ccs.pdf
7. Chen, F., Pasquale, J., Toward Improving Path Selection in Tor, http://cseweb.ucsd.edu/users/pasquale/Papers/globecom10c.pdf
8. Geddes, J., Jansen,R., Hopper,N., How Low Can You Go: Balancing Performance with Anonymity in Tor, 2013, http://www-users.cs.umn.edu/~hopper/howlow-pets2013.pdf
9. Panchenko, A., Lanze, F., Engel,T., Improving Performance and Anonymity in the Tor Network, http://lorre.uni.lu/~andriy/papers/ipccc12-tor-performance.pdf
10. Akhoondi, M., Yu, C., Madhyastha, H., LASTor: A Low-Latency AS-Aware Tor Client, http://freehaven.net/anonbib/cache/oakland2012-lastor.pdf
11. Kiraly, C. Effect of Tor window size on performance. Email to or-dev@freehaven.net, February 2009. http://archives.seul.org/or/dev/Feb-2009/msg00000.html