Analysis of Multi-path Onion Routing-Based Anonymization Networks

Wladimir de La Cadena, Daniel Kaiser, Asya Mitseva, Andriy Panchenko, Thomas Engel

Analysis of Multi-path Onion Routing-based Anonymization Networks

Wladimir De la Cadena[1], Daniel Kaiser[1], Asya Mitseva[1], Andriy Panchenko[2], and Thomas Engel[1]

[1] University of Luxembourg, Luxembourg
{wladimir.delacadena, daniel.kaiser, asya.mitseva, thomas.engel}@uni.lu
[2] Brandenburg University of Technology, Cottbus Germany
{firstname.lastname}@b-tu.de

Abstract. Anonymization networks (e.g., Tor) help in protecting the privacy of Internet users. However, the benefit of privacy protection comes at the cost of severe performance loss. This performance loss degrades the user experience to such an extent that many users do not use anonymization networks and forgo the privacy protection offered. Thus, performance improvements need to be offered in order to build a system much more attractive for both new and existing users, which, in turn, would increase the security of all users as a result of enlarging the anonymity set. A well-known technique for improving performance is establishing multiple communication paths between two entities. In this work, we study the benefits and implications of employing multiple disjoint paths in onion routing-based anonymization systems. We first introduce a taxonomy for designing and classifying onion routing-based approaches, including those with multi-path capabilities. This taxonomy helps in exploring the design space and finding attractive new feature combinations, which may be integrated into running systems such as Tor to improve users' experience (e.g., in web browsing). We then evaluate existing implementations (together with relevant design variations) of multi-path onion routing-based approaches in terms of performance and anonymity. In the course of our practical evaluation, we identify the design characteristics that result in performance improvements and their impact on anonymity.

1 Introduction

In modern society, people disclose a large quantity of digital traces via the Internet. Hence, privacy is attracting more and more attention and has become a serious concern. *Anonymization* is a basic technical means for achieving privacy. Despite the variety of approaches proposed for anonymous communication, only a few have reached widespread deployment. Currently, Tor [9] is the most popular low-latency anonymization network designed for TCP-based applications, serving more than two million daily users[3]. The main objective of Tor is to hide

[3] https://metrics.torproject.org/userstats-relay-country.html, October 2018.

the identities (i.e., IP addresses) of users who communicate through the Internet. To start a connection via Tor, the user runs local software, an *onion proxy* (OP), and creates a virtual tunnel, referred to as a *circuit*, to the destination over three nodes, known as *onion relays* (ORs) [8]. The ORs are run by volunteers who determine the amount of bandwidth they are willing to share. Depending on their position on the circuit, the ORs are denoted as *entry*, *middle*, and *exit*. Via a Diffie-Hellman key exchange, the user negotiates a distinct symmetric key with each OR on the circuit. The symmetric keys are used to encrypt the actual user data in multiple layers of encryption [8]. While forwarding user traffic, each OR on the circuit removes (or adds, depending on the direction) a layer of encryption. This ensures that none of the ORs on the circuit knows both the source and the destination of a connection at the same time. Along a circuit, user traffic travels encapsulated in fixed-size units referred to as *cells*.

Due to the diverse resource capabilities of ORs and their dynamic nature — anybody can join the network by running an OR or leave the network at any time — Tor suffers from both high congestion and latency. This often leads to significant delays for users which, in turn, may discourage them from using the network. Since the strength of anonymity provided by Tor strongly depends on the number of users, the protection of Tor clients utilizing the network is weakened by any user leaving the network. Therefore, performance improvements are necessary to make the system more attractive for both new and existing users. This will further improve the security of all users due to the increased anonymity set.

In response to this, a significant amount of research has focused on optimizing Tor's performance by improving its circuit processing [4, 36, 38], transport mechanisms [29, 39, 41], and relay selection algorithms [2, 33, 42], analyzing relay recruiting techniques [12, 20, 21], and adopting throttling methods [22] to reduce the load on the network. However, none of this work has investigated the performance benefits of multiple, disjoint paths used at overlay level when transmitting user data for a single Tor client. Although a few works [3, 44] have suggested concrete approaches to deploying multi-path techniques in Tor, their evaluations are limited by unrealistic and outdated conditions.

In this paper, we present an up-to-date review of existing multi-path approaches particularly designed for Tor and similar onion routing-based low-latency anonymization systems. By conducting experimental evaluations at different scales, we analyze the state-of-the-art multi-path anonymization techniques in terms of the performance gain and anonymity implications of each approach. Our contribution is two-fold:

1. We provide a systematic survey of currently-existing multi-path approaches for Tor and other similar onion routing-based anonymization systems as well as techniques that allow adding multi-path capability. To this end, we introduce a taxonomy for onion routing-based low-latency designs with a focus on multi-path approaches and classify the existing related works accordingly.
2. We conduct a comprehensive evaluation to compare these approaches in terms of both performance and anonymity. Based on the results from our

evaluation and our theoretical analysis, we discuss which design choices should be considered to achieve a desired set of properties in new systems.

2 Related Work

To improve the performance of the Tor network, a significant amount of research has focused on exploring a variety of relay selection algorithms, e.g., by trying to avoid congested ORs [42], considering the geographical location [2] or bandwidth [34, 35] of chosen ORs. Another group of works [10, 27, 29] criticizes the transport design applied by Tor, i.e., circuits from several users are multiplexed through a single TCP connection between two ORs. This may slow down the performance of interactive circuits. In response to this, several works evaluate advanced circuit scheduling mechanisms [36,38], propose improved congestion control algorithms [4,29] or even replacing the underlying transport protocol [26,41] to optimize the utilization of available bandwidth in Tor. In contrast to our study, these works did not evaluate the effect of multi-path techniques in Tor. Nevertheless, these proposals complement our work and their coexistence can further improve the performance and harden the security of Tor.

Karaoglu et al. [23] propose a multi-path routing scenario which emulates the operation of multi-path TCP [13]. Here, the Tor client is responsible for splitting and sending the traffic through multiple disjoint circuits to a web server which, in turn, is required to merge the received data. Thus, the authors do not require any modification in the core Tor network. However, Karaoglu et al. consider only a unidirectional scenario, in which the client uploads a file to the web server. Furthermore, the authors do not make any comparison with existing state-of-the-art multi-path approaches proposed for Tor or other onion routing-based anonymization systems. Last, but not least, Ries et al. [30] compare different low-latency anonymization networks with respect to their usability and the level of anonymity that they provide. Unlike our work, there is no evaluation of the applicability of multi-path techniques within these anonymization networks.

3 Multi-path in Anonymization Systems

Using multiple paths in anonymization systems has been also considered in previous theoretical analyses, simulations, and non onion routing approaches. The objectives pursued by those works were: passive attack resilience [11,32], multi-path as a means of anonymity [24], and performance improvements [23,33]. However, only three systems have been fully developed and implemented as multi-path onion routing-based approaches. Two of these, Conflux [3] and mTor [44], are extensions to vanilla Tor that adapt its traffic management design to utilize multiple circuits; the third, MORE [25], comprises a multi-path design over UDP where each cell travels along a different circuit. To our knowledge, there is no fully-developed multi-path approach that is both UDP-based and uses, as Tor does, fixed circuits per data transfer. For a more comprehensive analysis of standard transport protocol (UDP, TCP)-based multi-path approaches, we consider

closing this gap in the design space to be necessary and so added multi-path support to UDP-OR [41] as a further contribution; we refer to the result as mUDP-OR. We chose enhancing UDP-OR because it is fully-developed and relies on standard transport protocols (see section 4). The remainder of this section describes the multi-path onion routing-based systems analyzed and evaluated in this paper.

3.1 Conflux

In this design (see [3]), the OP builds multiple circuits with the same *exit* OR. Once those circuits are created, the OP sends a cell with a random nonce towards the *exit* OR as an identifier of the multi-path structure. To send each cell, the OP and the *exit* OR, known as *end-points*, select one of the multiple circuits according to its congestion, which is estimated as the time interval between the 100^{th} cell being sent, and the corresponding *sendme*[4] being received. Cells that arrive out-of-order to the *end-points* are merged and sorted using a 4-byte sequence number included in the cell's payload. Conflux presents results from an implementation that supports only two circuits. For our analysis, we have enhanced the Conflux's design in order to support m circuits.

3.2 mTor

Here (see [44]), the multi-path structure and cell merging procedure is similar to Conflux. However, *end-points* choose one of the multiple circuits according to its current stream-level window[5] value. The *end-point* drops cells to the circuits in a first-in-first-out manner, while their stream window is greater than zero.

3.3 MORE

In MORE (see [25]), it is required that the client participates as OR within the network (peer-to-peer network). To send data, the client OR captures TCP data via a TUN device[6] and encapsulates it in cells, which will each be sent across a different circuit. This means that no initial circuit establishment takes place, but that each cell travels along its own randomly-chosen path. To guarantee reliability of cells traveling along different routes, MORE takes advantage of the TUN device's functionality and provides an IP overlay service for tunneling TCP data. In this sense, a multi-path layer TCP session exists between sender and receiver. To discover each cell's route, an intermediary OR onion-decrypts and reads the corresponding successor node from the header. To reduce the computational cost

[4] Cell used for flow control: an *end-point* sends it to acknowledge the arrival of 100 cells within a circuit, or 50 cells within a stream.

[5] As part of the flow control, the stream-level window caps the maximum amount of cells to 500 per stream at any moment.

[6] A TUN device is a virtual kernel network interface that works as a bridge between the user and kernel spaces.

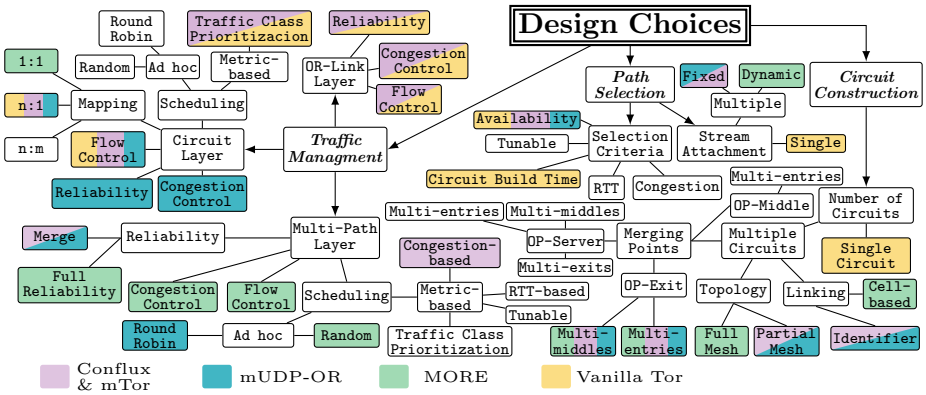

Fig. 1. Taxonomy of design choices for onion routing-based approaches

of re-setting up a cryptographic context for each cell, MORE uses elliptic curve cryptography (ECC). While using one circuit for each cell increases the resilience against traffic analysis attacks, it also considerably reduces performance.

3.4 mUDP-OR

Here (see [41]), the multi-path structure and circuit identification is performed in a manner similar to Conflux. However, ORs in a circuit communicate with each other using the UDP transport protocol. This circuit is used for tunneling TCP application data. Instead of encapsulating complete TCP segments, an *end-point* builds cells, appending to the header the necessary TCP fields (e.g. sequence numbers) to reconstruct a TCP packet at the other *end-point*. This TCP virtual connection is realized by setting up a SOCKS proxy in the *exit* OR, and establishing a virtual tunnel from a virtual TUN device in the OP. We implemented two strategies to dispatch cells into the circuits. In the first, the *end-point* chooses the circuit in a round robin (RR) manner with a configurable number of cells per circuit. In the second, the *end-point* randomly chooses through which circuit the next cell will be sent. We leverage the existing circuit-layer TCP session to merge cells arriving from different circuits. In this sense, the existing virtual end-to-end TCP connection is agnostic to the circuit(s) used.

4 Classifying Design Choices

In this section we introduce a hierarchical taxonomy for classifying and discussing onion routing design choices. The top level classes of our taxonomy comprise *traffic management*, *path selection*, and *circuit construction*; Figure 1 illustrates our taxonomy. We focus on the multi-path aspects and the effect of adding multi-path capabilities. Based on the structure of our taxonomy, we classify and discuss the multi-path OR approaches introduced in the previous section.

4.1 Traffic Management

The *traffic management* class comprises design choices which are concerned with transmitting data over already-established circuits in an anonymization overlay network; specifically regarding providing a TCP-like end-to-end service and scheduling decisions. This class is a key element of designing OR approaches and significantly affects performance. It also has an effect on anonymity, as feedback mechanisms might leak information, allowing for fingerprinting attacks [29].

We classify traffic management into *OR-link layer*, *circuit layer*, and *multi-path layer*. These layers are intertwined, as their combination must provide the same service as a direct TCP connection, namely reliability, congestion control, and flow control. Inter-layer dependency causes some issues, the most prominent of which is cross-circuit interference [42]. In general, cross-circuit interference is a consequence of OR-link layer connection artifacts affecting virtually independent circuits, because several circuit-layer connections share the same OR-link.

OR-Link Layer: The *OR-link layer* comprises the transport connection between ORs. We classify the *OR-link layer* design according to which of reliability, congestion control, and flow control it incorporates. Tor uses TCP on the OR-link layer, realizing reliability, congestion control, and flow control on this layer. Since Tor multiplexes all circuit segments over a single OR-link layer connection (TCP connection) between ORs and TCP mechanisms are agnostic to these circuits, it is subject to cross-circuit interference; specifically, because of shared I/O buffers and congestion control. Shared I/O buffers are a problem because segments are taken out of the shared TCP buffer on a first-come-first-served basis, no matter which circuit they are associated with. This leads to high latency for all circuits in the presence of high-throughput circuits that congest the shared TCP I/O buffer. This, in turn, may render interactive sessions using a low-throughput circuit over the same TCP connection unusable, as there is no means for prioritizing an interactive session.

Congestion control causes TCP connections to be throttled in the case of a congestion event[7]; thus, if a congestion event occurs related to a single circuit, all circuits over the same TCP connection are throttled. Even without congestion control, reliability[8] would cause cross-circuit interference because the recovery from packet loss in one circuit would also affect all other circuits sharing the same TCP connection.

Two classes of solutions addressing Tor's cross-circuit interference have been proposed; firstly, dedicating a TCP connection to each circuit segment [5]; and secondly, using a simple transport protocol, e.g., UDP [41]. Conflux and mTor are Tor extensions that add the multi-path layer while inheriting this weakness of Tor. mUDP-OR and MORE both use UDP as a transport protocol, avoiding cross-circuit interference. However, this countermeasure leads to aggressive traffic[9], which might congest the network. This issue has been addressed in [39].

[7] A congestion event might, e.g., be a packet loss.
[8] The realization of reliability is typically intertwined with congestion control.
[9] traffic sent at high rates even in case of network congestion

A multi-path based mitigation technique for cross-circuit interference on the OR-link layer, which to our knowledge has not yet been discussed, would be the use of multi-path TCP [13] as a transport protocol. Since multi-path TCP handles scheduling among the various TCP sub-streams on the transport layer, it is not suited to circuit-aware scheduling. Still, having several TCP sub-streams would lower the risk of cross-circuit interference while potentially multiplexing several circuits over a single connection hiding them in an anonymity set. However, especially in congested networks, having several TCP connections also increases the aggressiveness of traffic [40].

Circuit Layer: The *circuit layer* comprises a single overlay connection between an OP and an *exit* OR. As with the OR-link layer, we classify the *circuit layer* design by which of reliability, congestion control, and flow control it incorporates. The Tor circuit layer protocol [8] does not implement reliability, since it is already provided by TCP at the OR-link layer. It provides flow control with a fixed-size-window-based mechanism and no congestion control. Reliability methods do not benefit from inter OR-link or inter-layer communication and thus should be realized on one layer exclusively. Flow control and congestion control can benefit from inter OR-link and inter-layer interaction [39], and thus may be (partially) realized on several layers. Both having a fixed-size window for flow control and not providing congestion control have been identified as the major performance limiting factors of Tor [6]. Prioritization of interactive connections on the circuit-level has been proposed by Tang et al. [36] as a mitigation technique for cross-circuit interference, making interactive connections more responsive.

Conflux and mTor also inherit the properties of Tor for the circuit layer. mUDP-OR tunnels TCP, meaning the onion proxy and the exit node have a virtual TCP connection; thus, mUDP-OR provides all of flow control, congestion control, and reliability on the circuit layer. MORE is an overlay IP service where TCP data can be tunneled, making it part of the same class as mUDP-OR. The advantage of both mUDP-OR and MORE is being able to avoid cross-circuit interference. However, the OP-to-exit feedback loop for congestion control and reliability realization is very long and therefore not responsive. If a packet is dropped on the first circuit segment, this packet loss is detected at the end of the last circuit segment and the notification of this event needs to travel all the way back. The same problem occurs for adapting the TCP congestion window. Further, because mUDP-OR tunnels kernel-level TCP, the feedback across the whole circuit allows OS fingerprinting attacks [29].

A further property we use to classify the *circuit layer* by is *circuit to OR-link mapping*. The *circuit to OR-link mapping* decides how circuit segments are mapped to connections between the corresponding pair of ORs. Realizations comprise (1) $n : 1$, where all circuit segments between a pair of ORs are multiplexed over one transport connection, (2) $1 : 1$, where each circuit segment is mapped to a dedicated transport connection, and (3) $n : m$, where several circuit segments between a pair of ORs are multiplexed over a set of transport connections. While (1) may suffer from cross-circuit interference (e.g., when reliability

is provided) but offers the best anonymity properties, (2) prevents cross-circuit inference but may allow passive attackers to infer which circuit a given packet is associated with, which in turn might allow association with the sender. A compromise is provided by (3) which reduces cross-circuit interference while still hiding packets in an anonymity set. Tor implements strategy (1), which is inherited by Conflux and mTor. mUDP-OR also implements this strategy. MORE implements strategy (2) and further uses a new circuit for each (set of) cell(s). The multi-path TCP based solution described above is an example of (3).

We also classify the *circuit layer* by its *circuit scheduling* method. If several circuits share a transport connection, cells associated with various circuits are multiplexed over this connection. Circuit-layer scheduling is concerned with how to choose which cell from various circuit-level output queues should be the next to be put into the transport-level output queue. We classify *circuit scheduling* methods into (1) *ad hoc*, and (2) *metric-based*. *Ad hoc* methods do not depend on a metric; subclasses are, e.g., (1a) *random*, where cells are randomly taken from input queues and put into the output queue, and (1b) *round robin*. *Metric-based* methods collect information about available circuits. This information is used to calculate a metric, based on which scheduling decisions are made. A subclass is (2a) *traffic class prioritization*, where specific traffic classes, e.g., traffic from an interactive connection, are prioritized. (1) is simple to implement and neither consumes additional computational power nor needs extra network messages. However, as shown in [6], (2) provides superior overall performance.

Prior to 2012, Tor used *round robin* as its scheduler. Then, an improved scheduler based on the recent circuit's activity was implemented [36]. Most recently, in 2017 a new scheduler called KIST [18] was introduced. It uses feedback from the kernel to prioritize the traffic of each circuit's queue. Conflux and mTor inherit this characteristic from Tor. mUDP-OR does not maintain circuit-level queues and therefore directly passes cells to the transport layer. Because MORE has a $1:1$ mapping between circuit segments and OR-links, it too does not implement any circuit-level scheduling and leaves this task to the transport layer. Not having a circuit-level queue decreases feedback time and total queueing delay, but comes at the cost of not having the advantages of *circuit scheduling*.

Multi-Path Layer: The *multi-path layer* incorporates sets of circuits jointly building communication channels. We classify the *multi-path layer* design by which of congestion control and flow control it considers. While it is a feasible design choice for the multi-path layer to be agnostic to both flow control and congestion control, the realization of reliability for a multi-path approach always includes the multi-path layer. The subclasses of multi-path reliability are *merge* and *full reliability*. The former expects the underlying circuits to provide a reliable ordered stream of cells — either by realizing reliability on the OR-link layer or on the circuit layer — and merges cells coming from different circuits. The latter collects all packets from the associated circuits and fully implements reliability. Having reliability on the multi-path layer allows for sending control information on less-congested circuits to reduce feedback time.

While Tor does not offer multi-path capabilities, both Conflux and mTor can be seen as multi-path extensions to vanilla Tor. As Tor already provides reliability and congestion control on the OR-link layer and flow control on the circuit layer, both solutions apply the *merge* strategy on the multi-path layer. Since mUDP-OR is a multi-path extension of UDP-OR, which already provides a means for anonymizing a reliable connection, mUDP-OR adds *merge* on top of the circuit layer provided by UDP-OR. MORE sends cell(s) over a different unreliable circuit; thus, *full reliability* is performed at the *multi-path layer*.

Another multi-path layer design choice is *multi-path scheduling*. While *circuit scheduling* decides from which circuit-level queue the next cell is put into the transport-level queue, *multi-path scheduling* decides over which circuit a given cell should be sent. The classes of scheduling algorithms, however, are the same as for *circuit scheduling*. New subclasses are (2b) *congestion-based*, where cells are sent through less congested circuits, (2c) *round trip time (RTT) based*, where circuits with lower RTT are prioritized, and (2d) *tunable*, which is a tunable combination of the other subclasses. As multi-path layer scheduling allows for congestion control which, in turn, leads to more even utilization of circuits, it also helps in mitigating cross-circuit interference. Both Conflux and mTor implement *congestion-based* scheduling. While Conflux's scheduling strategy has a very long feedback loop (see section 3), mTor implements a more responsive method based on the stream-level receive window size. Still, in absolute terms, the feedback loop is long. The mTor scheduling algorithm improves the throughput of bulk transfers while not negatively affecting interactive sessions. The default scheduler used in mUDP-OR is *round robin*. MORE is special in this case, as it creates new circuits on the fly for each cell and sends cells over the respective newly-created circuit. Thus, it depends on path selection and circuit construction discussed in the following subsections. The scheduling itself is therefore ad hoc, because a cell is scheduled to the only available circuit at a given point in time.

Multi-path TCP [13] could be used not only on the OR-link layer, but also on the circuit and multi-path layers, tunneling multi-path TCP's sub-streams on the circuit layer and using its scheduling and merging strategy on the multi-path layer. While this solution has the advantage of using an established protocol, it comes with little flexibility for adapting it to be a Tor transport. Such a solution should *not* use TCP at the OR-link layer as this would lead to TCP over TCP throttling effects [37].

Summarizing the realization of TCP functionality, all approaches directly use TCP and do not introduce custom designs. Both Conflux and mTor use TCP on the OR-link layer, mUDP-OR uses TCP on the circuit layer, and MORE uses TCP[10] on the multi-path layer. Like Tor, Conflux and mTor add only a simple flow control mechanism on the circuit layer. More sophisticated approaches tailored to anonymization overlay networks (see, e.g., [39]) have not as yet been used in the context of multi-path onion routing.

[10] MORE uses TCP when anonymizing a reliable service. Because MORE provides an IP service on the overlay, it can also be used without providing TCP functionality at all.

4.2 Circuit Construction

This design class comprises the considerations for building the path(s) that the OP will employ. The only subclass of *circuit construction* is the *number of circuits* required by the OP for exchanging data. The subclass *single circuit* is valid for Tor, since only one circuit is required by the OP for a data transfer. If *multiple circuits* are required, the design choice needs to specify where the *merging/splitting points* are. This in turn defines how many ORs per position (*entry*, *middle*, or *exit*) can compose a circuit. This design choice influences anonymity, performance and implementation complexity. Conflux, mTor, and mUDP-OR enlarge the bandwidth capacity of the last hop by building extra middle-to-exit connections. From the anonymity perspective, using multiple *entry* ORs may improve the resilience against some attacks (see section 6). None of the considered approaches merge on a *middle* OR; this scheme would represent a more complex implementation but at the same time an easier deployment in the network, since there are fewer requirements for starting a *middle* OR in Tor [1].

Another class refers to the *topology* formed by the selected ORs. Conflux, mTor, and mUDP-OR form a *partial mesh*, since each *entry* OR communicates with one *middle* OR. MORE tends to form a *full mesh* as the number of sent cells increases.

Lastly, the *linking* subclass refers to the mechanism to associate/save several circuits as a singular structure upon their creation. In Conflux, mTor and mUDP-OR, multiple circuits are referred by an *end-point* under a common identifier exchanged via a control cell. This type of *linking* comprises the subclass *identifier*. The other subclass, *cell-based*, is used by MORE. Here, paths are not linked in the construction process, but their cells will be grouped during the data transmission based on their header. This *linking* class is strongly related to the scheduling from the multi-path layer, and choosing it properly results in faster multi-path build times, and a more secure multi-path structure.

4.3 Path Selection

Preemptively, more than the required circuits can be built before streams are attached to them. This design choice determines which of the built path(s) will be next used for the data transfer. Once the path(s) are selected, the OP sends cells based on the *traffic management* design choices.

The subclass *selection criteria* determines which parameter(s) must be considered for defining which circuit(s) will be employed. In Tor, after discarding circuits with slow build times, the newest available is chosen. Other parameters such as RTT, congestion, or a tunable combination of these may be also considered. The subclass *stream attachment* comprises special choices for multi-path approaches. In contrast to Tor, where the stream will be directly attached to a single circuit, multiple circuits allow this attachment to be *fixed*, when the set of selected circuits does not change after they are chosen, or to be *dynamic* when the set of selected circuits may change during the data transfer. When the

set of selected circuits changes as dynamically as in MORE, this design choice determines the *multi-path layer* scheduling.

To sum up, the top level classes of the taxonomy address the design choices to be considered before user data is sent (*path selection* and *circuit construction*), and for the data transmission itself (*traffic management*). In our evaluation, we identify the effects of the design choices employed by the analyzed approaches.

5 Performance Evaluation

In this section, we evaluate each approach within two scenarios: on an isolated private local network, and in a larger network using the NetMirage[11] emulator.

5.1 Private Local Network Experiment

We use this experiment to understand the differences between all designs without external influence. In a local network we set up seven ORs, one client for measurements, four web servers, and up to 30 clients to generate load on the employed circuit(s). Three metrics are reported on the client: TTFB (Time to First Byte), and download times (DT) for HTTP web (320 KiB) and bulk (1 MiB) requests[12]. Furthermore, on each OR the CPU usage was periodically logged. Considering that there are no congestion effects from other sources in an isolated network, we evaluated each approach with the round robin multi-path scheduler. This also ensures that multiple circuits will be equitably used. Effects of congestion-based schedulers are evaluated in the second scenario.

Multi-path Circuits and Load Balancing: In the left columns of Table 1 we present the average CPU load on each OR for the maximum number of clients. For multiple circuits, we present results for only one of them, since values in others are similar. It is clearly observable that the load assigned to each OR is decreased by using multiple circuits simultaneously. Furthermore, it is noticeable that translating the reliability and congestion control tasks to the *end-points* (in mUDP-OR and MORE) results in a higher load on them. We also observe that *entry* ORs are more loaded than others (except by MORE) in the circuit due to the cryptographic operations performed.

Client Performance: Performance metrics are presented in the right columns of Table 1. As an expected consequence of the dynamic stream attachment in MORE, its clients experience the slowest download times, making it unfeasible to complete data transfers in many cases (e.g., for bulk downloads). We confirm that a UDP-based approach such as mUDP-OR responds faster than a TCP-based approach. In contrast to Conflux and mTor, our multi-path enhancement to UDP-OR did not produce the desired improvements due to the still-existing very long feedback for retransmissions and acknowledgments.

[11] https://crysp.uwaterloo.ca/software/netmirage/

[12] Values presented with 95% confidence and based on 200 repetitions.

Table 1. CPU usage percentage on the onion routers, and performance metrics for the maximum number of clients.

		CPU Usage on the OR			Client Performance Metrics		
Approach	Paths	Entry	Middle	Exit	TTFB[s]	DT Web [s]	DT Bulk[s]
Conflux	1	57.87 ± 2.16	49.85 ± 1.82	31.74 ± 1.15	0.027 ± 0.005	0.059 ± 0.003	0.113 ± 0.008
	2	33.22 ± 2.25	31.01 ± 2.21	31.51 ± 2.22	0.016 ± 0.002	0.052 ± 0.0005	0.082 ± 0.002
	3	27.24 ± 1.94	22.78 ± 1.56	32.94 ± 2.28	0.014 ± 0.003	0.049 ± 0.0007	0.079 ± 0.0024
mTor	1	57.87 ± 2.16	49.85 ± 1.82	31.74 ± 1.15	0.027 ± 0.005	0.059 ± 0.003	0.113 ± 0.008
	2	25.10 ± 2.21	23.19 ± 1.73	32.51 ± 2.26	0.012 ± 0.002	0.055 ± 0.001	0.079 ± 0.003
	3	13.45 ± 0.97	14.62 ± 0.85	32.90 ± 2.27	0.017 ± 0.003	0.049 ± 0.0007	0.081 ± 0.002
mUDP-OR	1	88.31 ± 1.51	86.89 ± 1.48	71.33 ± 1.24	0.002 ± 0.0003	0.031 ± 0.0018	0.076 ± 0.005
	2	36.75 ± 2.51	32.67 ± 2.79	73.48 ± 3.09	0.002 ± 0.0001	0.034 ± 0.0017	0.081 ± 0.001
	3	22.73 ± 3.11	24.45 ± 3.27	75.30 ± 3.21	0.0024 ± 0.0001	0.035 ± 0.001	0.113 ± 0.014
MORE	1	6.73 ± 0.14	6.93 ± 0.19	75.92 ± 2.34	0.014 ± 0.004	1.41 ± 0.007	N.A
	2	6.65 ± 0.13	6.93 ± 0.2	70.09 ± 2.24	0.014 ± 0.004	1.37 ± 0.19	N.A
	3	6.57 ± 0.11	6.52 ± 0.08	68.19 ± 2.48	0.016 ± 0.006	1.39 ± 0.014	N.A

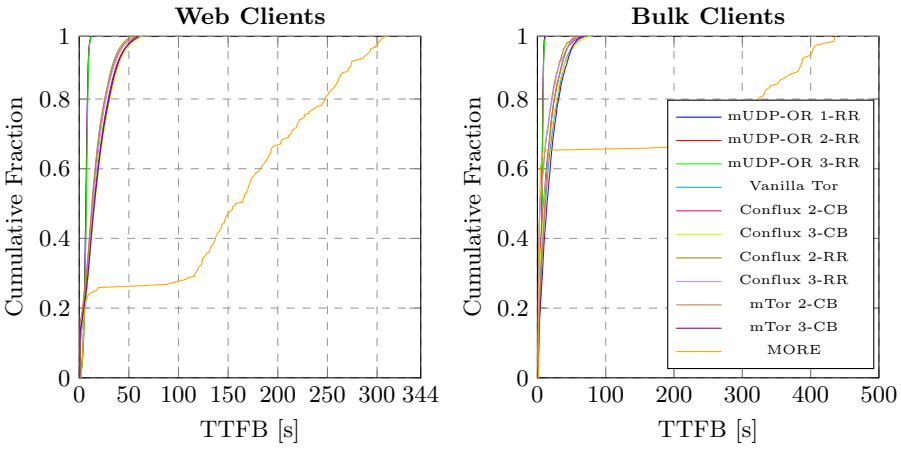

Fig. 2. Time to first byte for web and bulk clients

5.2 Larger-Scale Experiment

Currently, the Shadow [19] tool is widely used for large-scale Tor simulations. However, due to lack of support for some required functions (e.g., TUN devices) in Shadow, we opted for the NetMirage tool for building a common testbed. We based our experiment on the PlanetLab and Tor topologies included in version 1.12.1 of the Shadow simulator. It consisted of 303 nodes distributed all over the world, where we set up 206 web clients, 22 bulk clients, 14 exit nodes, 59 non-exit nodes and 14 web servers. Web clients performed successive downloads of 320 KiB data, waiting randomly from 0 to 20 seconds between each download, and bulk clients downloaded 1 MiB sequentially without pausing.

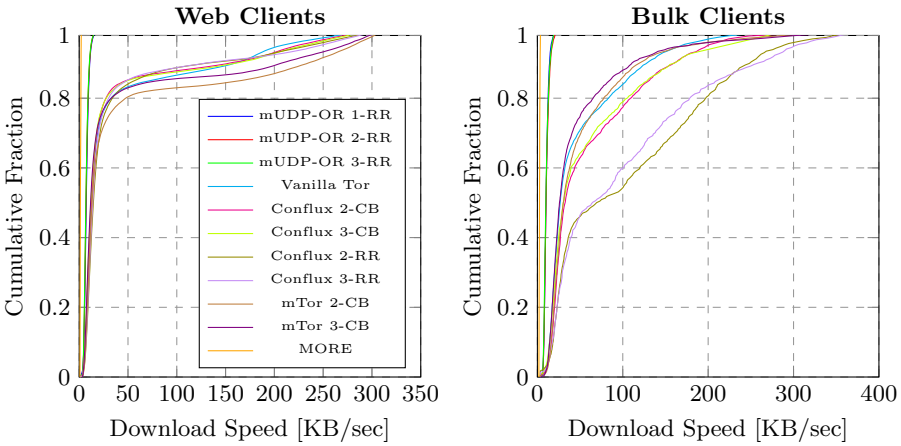

Fig. 3. Download speed for web and bulk clients

Client Performance: For every approach, each design variation[13] was emulated for two hours (see Figure 2 and Figure 3). We observe that, in a congested environment, mUDP-OR only outperforms other approaches in the TTFB metric. Moreover, the congestion-based scheduling techniques of mTor and Conflux do not profit completely from the utilization of multiple circuits; this may explain why the RR scheduler performs better, particularly for bulk downloads. Thus, it is necessary to develop a more efficient circuit congestion estimation procedure. Since Tor does not directly access the congestion information provided by TCP for each *OR link*, the estimations done in the circuit layer are not completely reliable and may not represent the state of the circuit at that moment. We observe that the improvements in downloading data are more advantageous to bulk transfers. Moreover, the TCP-based approaches outperform the UDP-based ones in terms of download speed for nearly all the 228 clients.

Network Scalability: In this experiment we incrementally introduced up to 228 clients (10% bulk and 90% web clients) and measured TTFB and download speed for each iteration (see Figure 4). The fast first response of mUDP-OR is clearly advantageous within a congested network; however, clients of Conflux and mTor download data faster. We observe that the download speed for all approaches stabilizes to its minimum value when around 140 clients are present. After this point, differences between all approaches remain constant. We notice that using RR for multi-path scheduling scales better, due to the equitable usage of network resources. It is also noticeable that congestion-based mechanisms perform better in a lightly-congested environment; this reinforces the intuition

[13] Two design variations were evaluated: the number of paths (labeled as 1,2,3) and the *multi-path* scheduler (labeled as RR for round robin and CB for congestion based).

that the employed congestion estimation techniques are not fully precise. We refrain from comparing MORE in this regard due to its poor performance.

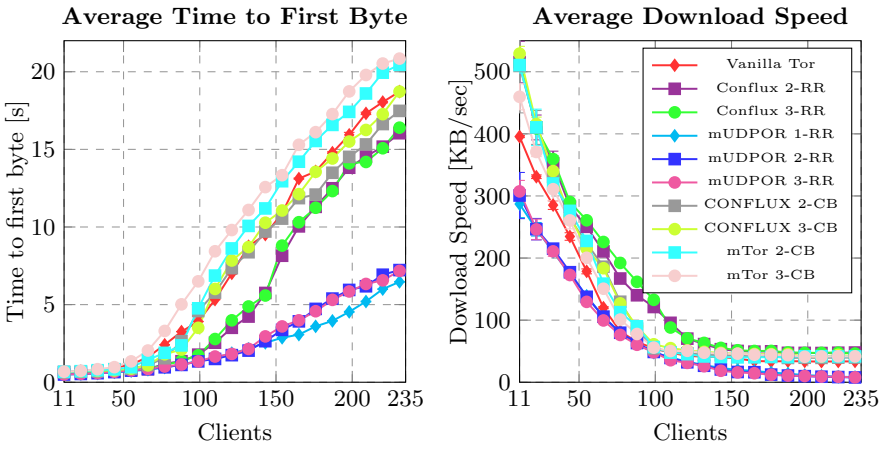

Fig. 4. Performance metrics for different number of clients

5.3 Design Recommendations

From the performed evaluations we identify that any design (single or multi-path) based on UDP, provides a fast first response. This feature comes however, at the cost of a degraded performance. If fast download speed is desired (e.g., in web browsing), the design should use the TCP protocol on the OR-link layer together with an effective congestion-based multi-path scheduler. If the objective is to ease the burden on ORs, the round robin scheduler ensures an equitable traffic distribution. If performance is not of the essence — for instance in non-time-sensitive applications like messaging or microblogging — even higher anonymity can be achieved by systems with the characteristics of MORE.

6 Anonymity Analysis

In this section, we address the anonymity implications produced by using multiple paths in the context of the evaluated approaches.

6.1 Client Multi-path Circuits Compromise

A circuit becomes compromised if an attacker gains control over both its edges. An adversary that controls a fraction of entry and exit nodes (f_g and f_x), can compromise any single circuit with a probability $P(c) \approx f_g f_x$ [3, 7]. For multi-path clients employing m *entries* and one *exit* OR, this expression becomes

$P_m(c) \approx f_x(1-(1-f_g)^m)$. This expression is valid for all evaluated approaches; however, for MORE, an adversary must compromise many more than m circuits to fully affect one client, which means that this approach provides higher levels of anonymity. Even though $P_m(c) \geq P(c)$, this difference is negligible even in the presence of a powerful attacker[14].

6.2 Using Multiple Entry Onion Routers

To make the probability of de-anonymization vanishingly small, Tor clients try to choose the same *entry* OR from the priority-ordered *primary* list[15]. Since a multi-path client uses m entries, they should be taken from a *primary* list of minimum size m. In order to evaluate the anonymity implications, we leverage the framework presented in [17] together with metrics and adversary models presented in [15]. Two adversary models are considered for a client using m entries, the first determines that a client is compromised if at least one entry is controlled, which may be valid for confirmation and correlation attacks [14]. The second, defines a compromised client if and only if all m entries are controlled, which may be valid for *website fingerprinting* attacks [16,28,31,43]. Both models are valid for designs that assign streams to a fixed set of circuits (Conflux, mTor and mUDP-OR). For systems with dynamic stream attachment (MORE), the models are valid during the usage interval of the circuits. Using consensus data from 2015, we simulated 500,000 clients and a high-resource adversary controlling 10% of the overall entry bandwidth. Figure 5 shows the mean compromise rate (CR) of 50 simulations. We notice that, for the second adversary model, the CR decreases exponentially with each additional entry. Conversely, if one from m entries is enough to compromise a client, they become around twice as vulnerable.

Lastly, we analyze the *guard fingerprinting* attack[16], where using multiple entries decreases the mean anonymity set size ($\overline{A}$). Currently, each Tor client shares its *entry* OR with on average another 1,000 users ($\overline{A} = 1000$). If clients used m paths, $\overline{A}$ would drastically decrease to $\frac{2\times10^6}{\binom{2000}{m}}$. Using the Tor source code, we simulated the creation of *primary* lists for 83,000 clients. For $m = 1$, we experimentally obtained $\overline{A} = 112$, while for $m = 2$ roughly 90% of clients had a unique pair of entries, and the user with the largest anonymity set shared its entries with another 14 users. For this attack, the dynamism of MORE is also favorable, because all clients tend to use all nodes as entries. Thus, $\overline{A}$ converges to its upper limit (the total number of clients).

To sum up, the anonymity advantages of using multiple *entry* ORs, together with the presented performance gains, are compelling reasons to enhance systems such as Tor with multi-path capabilities. The main constraint is the fact that

[14] An adversary controlling 20% of the total bandwidth (ca. 60 Gbit/s in Tor) is considered as very powerful; in this case $P(c) \approx 4\%$ and $P_m(c) \approx 9.8\%$ for $m = 3$.

[15] From all ORs with entry flags listed in the consensus, by default, a client filters three ORs from several lists, choosing the first that is reachable as its *entry* OR.

[16] Guard ORs refer to the entry ORs regularly selected by a client. This set of nodes may be used as a fingerprint for de-anonymizing a client.

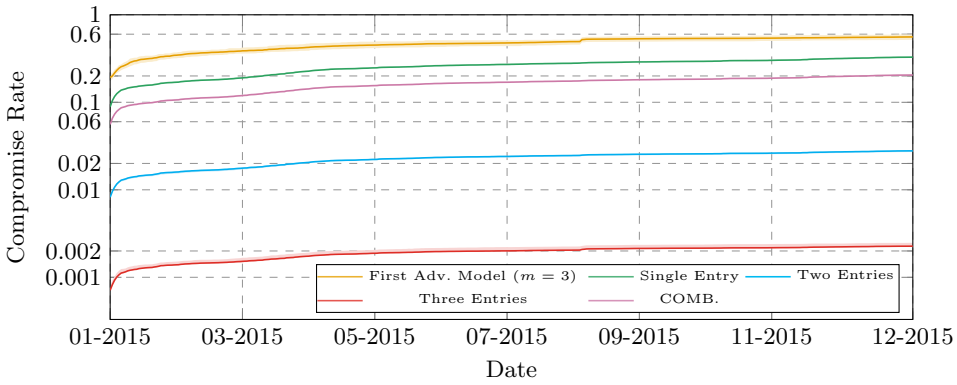

Fig. 5. Fraction of compromised clients (Compromise Rate) during one year: Single, two, and three entries refer to the second adversary model. In the scenario labeled as COMB, 60% of the clients use a single entry, 20% two entries and 20% three entries.

using multiple entries is not considered in the latest Tor specification, however future research directions [17] aim to give more flexibility in this regard.

7 Conclusions and Future Work

Onion routing-based approaches (e.g., Tor) can leverage multi-path capabilities as a means of enhancing the users' experience through performance improvement. To investigate these capabilities, we have presented a comprehensive analysis and evaluation of multi-path onion routing approaches regarding their design choices and realizations. By using the proposed taxonomy, we presented important guidelines to be followed not only for future multi-path onion routing-based designs, but also for other types of anonymization systems.

For future multi-path designs, greater performance improvements are expected if the current congestion estimation mechanisms can be refined to reflect the actual transport layer congestion into the multi-path layer. Furthermore, other aspects such as anonymity and load balancing should be taken into consideration when designing the multi-path circuit structure and scheduling mechanisms. We notice that for some attacks (e.g., *guard fingerprinting*) a considerable modification in the current node selection strategy is needed to guarantee a level of anonymity. Meanwhile, for other attacks such as *website fingerprinting* a quantitative analysis of their impact is required.

In future work we also plan to address cross-circuit interference, which is a significant problem in Tor, with mitigation techniques that often affect anonymity. We plan to analyze trade-offs between using different subsets of the mechanisms that TCP offers on the OR-link layer and specifically look into alternative congestion control methods. We want to improve performance while still avoiding network congestion, and also protect anonymity by not introducing end-to-end feedback and so opening additional attack vectors.

Acknowledgments: We thank the anonymous reviewers for their useful comments, and all authors of the evaluated approaches for providing their source code and assistance. This research was funded by the Luxembourg National Research Fund (FNR) within the CORE Junior Track project PETIT.

References

1. The Tor Relay Guide. `https://trac.torproject.org/projects/tor/wiki/TorRelayGuide` (2018)
2. Akhoondi, M., Yu, C., Madhyastha, H.: LASTor: A Low-Latency AS-Aware Tor Client. In: Symposium on Security and Privacy (S&P). IEEE, San Francisco, CA, USA (May 2012)
3. AlSabah, M., Bauer, K., Elahi, T., Goldberg, I.: The Path Less Travelled: Overcoming Tor's Bottlenecks with Traffic Splitting. Privacy Enhancing Technologies (PETS) **7981**, 143–163 (July 2013)
4. AlSabah, M., Bauer, K., Goldberg, I., Grunwald, D., McCoy, D., Savage, S., Voelker, G.: DefenestraTor: Throwing Out Windows in Tor. Privacy Enhancing Technologies (PETS) **6794**, 134–154 (July 2011)
5. AlSabah, M., Goldberg, I.: PCTCP: Per-circuit TCP-over-IPsec Transport for Anonymous Communication Overlay Networks. In: Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security. ACM, Berlin, Germany (November 2013)
6. AlSabah, M., I. Goldberg, I.: Performance and Security Improvements for Tor: A Survey. ACM Computing Surveys (CSUR) pp. 32:1–32:36 (November 2016)
7. Das, A., Borisov, N.: Securing Anonymous Communication Channels under the Selective DoS Attack. In: Proceedings of Financial Cryptography and Data Security. pp. 362–370. Springer, Okinawa, Japan (April 2013)
8. Dingledine, R., Mathewson, B.: Tor Protocol Specification. `https://gitweb.torproject.org/torspec.git?a=blob_plain;hb=HEAD;f=tor-spec.txt` (2018)
9. Dingledine, R., Mathewson, N., Syverson, P.: Tor: The Second-generation Onion Router. In: 13th conference on USENIX Security Symposium. USENIX Association, San Diego, CA, USA (August 2004)
10. Dingledine, R., Murdoch, S.: Performance Improvements on Tor or, Why Tor is slow and what we're going to do about it. Online: http://www.torproject.org/press/presskit/2009-03-11-performance.pdf (2009)
11. Feigenbaum, J., Johnson, A., Syverson, P.: Preventing Active Timing Attacks in Low-latency Anonymous Communication. In: Privacy Enhancing Technologies (PETS). pp. 166–183. Springer, Berlin, Germany (July 2010)
12. Ghosh, M., Richardson, M., Ford, B., Jansen, R.: A TorPath to TorCoin: Proof-of-Bandwidth Altcoins for Compensating Relays. In: 7th Workshop on Hot Topics in Privacy Enhancing Technologies (HotPETs). Amsterdam, Netherlands (July 2014)
13. Handley, M., Bonaventure, O., Raiciu, C., Ford, A.: TCP Extensions for Multipath Operation with Multiple Addresses. RFC 1654 (1995)
14. Hayes, J.: Traffic Confirmation Attacks Despite Noise. In: Understanding and Enhancing Online Privacy Satellite Workshop of NDSS. San Diego, CA, USA (April 2016)
15. Hayes, J., Danezis, G.: Guard Sets for Onion Routing. In: Proceedings on Privacy Enhancing Technologies (PETS). Springer, Philadephia, PA, USA (June 2015)

16. Hayes, J., Danezis, G.: k-fingerprinting: A Robust Scalable Website Fingerprinting Technique. In: 25th USENIX Conference on Security Symposium. USENIX Association, Austin, TX, USA (August 2016)
17. Imani, M., Barton, A., Wright, M.: Guard Sets in Tor using AS Relationships. In: Proceedings on Privacy Enhancing Technologies (PETS). Springer, Barcelona, Spain (July 2018)
18. Jansen, R., Geddes, J., Wacek, C., Sherr, M., Syverson, P.F.: Never Been KIST: Tor's Congestion Management Blossoms with Kernel-Informed Socket Transport. In: Proceedings of the 23rd USENIX Security Symposium. USENIX Association, San Diego, CA, USA (August 2014)
19. Jansen, R., Hopper, N.: Shadow: Running Tor in a Box for Accurate and Efficient Experimentation. In: Proceedings of the Network and Distributed System Security Symposium (NDSS). Internet Society (August 2012)
20. Jansen, R., Johnson, A., Syverson, P.: LIRA: Lightweight Incentivized Routing for Anonymity. In: 21th Annual Network and Distributed System Security Symposium (NDSS) (February 2013)
21. Jansen, R., Miller, A., Syverson, P., Ford, B.: From Onions to Shallots: Rewarding Tor Relays with TEARS. In: 7th Workshop on Hot Topics in Privacy Enhancing Technologies (HotPETs). Amsterdam, Netherlands (July 2014)
22. Jansen, R., Syverson, P., Hopper, N.: Throttling Tor Bandwidth Parasites. In: 21st USENIX Conference on Security Symposium. USENIX Association, Bellevue, WA, USA (August 2012)
23. Karaoglu, H., Akgun, M., Gunes, M., Yuksel, M.: Multi Path Considerations for Anonymized Routing: Challenges and Opportunities. In: 5th International Conference on New Technologies, Mobility and Security (NTMS). pp. 1–5. IEEE (May 2012)
24. Katti, S., Cohen, J., Katabi, D.: Information Slicing: Anonymity Using Unreliable Overlays. In: 4th Usenix Conference on Networked Systems Design and Implementation (NSDI). USENIX Association, Cambridge, MA, USA (April 2007)
25. Landsiedel, O., Pimenidis, A., Wehrle, K., Niedermayer, H., Carle, G.: Dynamic Multipath Onion Routing in Anonymous Peer-To-Peer Overlay Networks. In: 50th Annual IEEE Global Telecommunications Conference (GLOBECOM). pp. 64–69. IEEE, Washington, DC, USA (November 2007)
26. Loesing, K., Murdoch, S., Jansen, R.: Evaluation of a libutp-based Tor datagram implementation. Tech. rep., Tech. Rep. 2013-10-001, The Tor Project (2013)
27. Murdoch, S.: Comparison of Tor Datagram Designs. Tech. rep. (2011)
28. Panchenko, A., Lanze, F., Pennekamp, J., Engel, T., Zinnen, A., Henze, M., Wehrle, K.: Website Fingerprinting at Internet Scale. In: 23nd Annual Network and Distributed System Security Symposium (NDSS). Internet Society, San Diego, CA, USA (February 2016)
29. Reardon, J., Goldberg, I.: Improving Tor Using a TCP-over-DTLS Tunnel. In: 18th Conference on USENIX Security Symposium. pp. 119–134. USENIX Association, Montreal, Canada (August 2009)
30. Ries, T., Panchenko, A., State, R., Engel, T.: Comparison of Low-Latency Anonymous Communication Systems – Practical Usage and Performance. In: Ninth Australasian Information Security Conference (AISC) (2011)
31. Rimmer, V., Preuveneers, D., Juarez, M., Goethem, T.V., Joosen, W.: Automated Website Fingerprinting through Deep Learning. In: 25nd Annual Network and Distributed System Security Symposium (NDSS). Internet Society, San Diego, CA, USA (February 2018)

32. Serjantov, A., Murdoch, S.: Message Splitting Against the Partial Adversary. In: Privacy Enhancing Technologies (PETS). pp. 26–39. Springer, Cavtav, Croatia (June 2005)
33. Snader, R.: Path Selection for Performance- and Security-Improved Onion Routing. Ph.D. thesis, University of Illinois at Urbana-Champaign (2010)
34. Snader, R., Borisov, N.: A Tune-up for Tor: Improving Security and Performance in the Tor Network. In: 16th Annual Network and Distributed System Security Symposium (NDSS) (February 2008)
35. Snader, R., Borisov, N.: Improving Security and Performance in the Tor Network through Tunable Path Selection. IEEE Transactions on Dependable and Secure Computing (September 2011)
36. Tang, C., Goldberg, I.: An Improved Algorithm for Tor Circuit Scheduling. In: 17th ACM Conference on Computer and Communications Security. pp. 329–339. ACM, Chicago, Illinois, USA (October 2010)
37. Titz, O.: Why TCP Over TCP Is A Bad Idea, `http://sites.inka.de/bigred/devel/tcp-tcp.html`, `http://sites.inka.de/bigred/devel/tcp-tcp.html`, accessed 2018-11-16
38. Tschorsch, F., Scheuermann, B.: Tor is Unfair – And What to Do about It. In: 36th Conference on Local Computer Networks. IEEE, Bonn, Germany (October 2011)
39. Tschorsch, F., Scheuermann, B.: Mind the Gap: Towards a Backpressure-based Transport Protocol for the Tor Network. In: 13th Usenix Conference on Networked Systems Design and Implementation (NSDI). pp. 597–610. USENIX Association, Santa Clara, CA, USA (March 2016)
40. Tschorsch, F., Scheurmann, B.: How (not) to build a transport layer for anonymity overlays. In: Proceedings of the ACM Sigmetrics/Performance Workshop on Privacy and Anonymity for the Digital Economy. ACM, New York, NY, USA (June 2012)
41. Viecco, C.: UDP-OR: A Fair Onion Transport Design. In: 1st Workshop on Hot Topics in Privacy Enhancing Technologies (HotPETS). Leuven, Belgium (July 2008)
42. Wang, T., Bauer, K., Forero, C., Goldberg, I.: Congestion-Aware Path Selection for Tor. In: Financial Cryptography and Data Security (FC). pp. 98–113. Springer, Kralendijk, Bonaire (March 2012)
43. Wang, T., Cai, X., Nithyanand, R., Johnson, R., Goldberg, I.: Effective Attacks and Provable Defenses for Website Fingerprinting. In: Proceedings of the 23rd USENIX Security Symposium. USENIX Association, San Diego, CA, USA (August 2014)
44. Yang, L., Li, F.: mTor: A Multipath Tor Routing Beyond Bandwidth Throttling. In: IEEE Conference on Communications and Network Security (CNS). pp. 479–487. IEEE, Florence, Italy (September 2015)