PhD-FSTM-2021-006
The Faculty of Science, Technology and Medicine

DISSERTATION

Defence held on 11/01/2021 in Esch-sur-Alzette

to obtain the degree of

DOCTEUR DE L'UNIVERSITÉ DU LUXEMBOURG

EN INFORMATIQUE

by

Augusto Wladimir DE LA CADENA
Born on 4 September 1988 in Quito (Ecuador)

MULTIPATH ROUTING ON ANONYMOUS COMMUNICATION SYSTEMS: ENHANCING PRIVACY AND PERFORMANCE

Dissertation defence committee

Dr. Thomas Engel, dissertation supervisor
Professor, Université du Luxembourg

Dr. Andriy Panchenko, Vice Chairman
Professor, Brandenburg University of Technology

Dr. George Danezis
Professor, University College London

Dr. Ulrich Sorger, Chairman
Professor, Université du Luxembourg

Dr. Thorsten Ries
Post Luxembourg

"We are children of the days, children of time, and every day has a story to be told because, according to scientists, we are made of atoms, but a little birdie told me that we are also made of stories."

Eduardo Galeano

Abstract

We live in an era where mass surveillance and online tracking against civilians and organizations have reached alarming levels. This has resulted in more and more users relying on anonymous communications tools for their daily online activities. Nowadays, Tor is the most popular and widely deployed anonymization network, serving millions of daily users in the entire world. Tor promises to hide the identity of users (i.e., IP addresses) and prevents that external agents disclose relationships between the communicating parties.

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. On the other hand, the popularity of Tor has captured the attention of attackers wishing to deanonymize their users.

As a response, this dissertation presents a set of multipath routing techniques, both at transport and circuit level, to improve the privacy and performance offered to Tor users. To this end, we first present a comprehensive taxonomy to identify the implications of integrating multipath on each design aspect of Tor.

Then, we present a novel transport design to address the existing performance unfairness of the Tor traffic. In Tor, traffic from multiple users is multiplexed in a single TCP connection between two relays. While this has positive effects on privacy, it negatively influences performance and is characterized by unfairness as TCP congestion control gives all the multiplexed Tor traffic as little of the available bandwidth as it gives to every single TCP connection that competes for the same resource. To counter this, we propose to use multipath TCP (MPTCP) to allow for better resource utilization, which, in turn, increases throughput of the Tor traffic to a fairer extend. Our evaluation in real-world settings shows that using out-of-the-box MPTCP leads to 15% performance gain. We analyze the privacy implications of MPTCP in Tor settings and discuss potential threats and mitigation strategies.

Regarding privacy, in Tor, a malicious entry node can mount website fingerprinting (WFP) attacks to disclose the identities of Tor users by only observing patterns of data flows. In response to this, we propose splitting traffic over multiple entry nodes to limit the observable patterns that an adversary has access to. We demonstrate that our sophisticated splitting strategy reduces the accuracy from more than 98% to less than 16% for all state-of-the-art WFP attacks without adding any artificial delays or dummy traffic. Additionally, we show that this defense, initially designed against WFP, can also be used to mitigate end-to-end correlation attacks.

The contributions presented in this thesis are orthogonal to each other and their synergy comprises a boosted system in terms of both privacy and performance. This results in a more attractive anonymization network for new and existing users, which, in turn, increases the security of all users as a result of enlarging the anonymity set.

Acknowledgements

This work was conducted at the SECAN-Lab of the University of Luxembourg's Interdisciplinary Centre for Security Reliability and Trust (SnT).

First, I would like to thank Prof. Dr. Thomas Engel for giving me the opportunity to be part of the Secan-Lab group, where I conducted the research that resulted in this dissertation. Second, I would like to thank Prof. Dr. Andriy Panchenko for his constant guidance, patience, support, trust, and the valuable knowledge I have learned from him. Thanks to him, I know what doing top research means, and I am deeply indebted for that. Third, I would like to thank Prof. Dr. George Danezis for his constructive feedback on my work.

I want to thank all my colleagues at the Secan-Lab, particularly Asya and Daniel, and all the co-authors of the achieved publications. My research would not have been possible without their support and the fruitful discussions.

Most of all, I would like to thank my family. To Gina, for being always with me along this journey. To my parents, Misbel and Augusto, and to my brothers, César and William. All my accomplishments would not be possible without their support and understanding. My triumphs are theirs.

This research was funded by the Luxembourg National Research Fund (FNR) within the CORE Junior Track project PETIT. The experiments presented in this dissertation were carried out using the HPC facilities of the University of Luxembourg [1].

Contents

List of Figures

List of Tables

The unity is like a corn cob, if the kernel goes, the row goes,
and if the row goes, so does the cob.

Tránsito Amaguaña

1 Introduction

1.1 Motivation

Internet has undoubtedly become a fundamental catalyst for the modern society. For the majority of the worldwide population, daily life involves several online activities such as access to information, communication with other parties, education, voting, social networking, and commercial activities. Although in the past decades, Internet has become a massive communication means and people are more aware about the implications of using it, many users still ignore that their online activities and identity can be tracked and disclosed.

Internet was originally not designed to provide anonymous communications. Users' identities are generally linkable and observable by the destination or any third party with access to the communication channel. Even though the content of information exchanged on the Internet is typically protected via encryption, the exposure of sender-receiver relationships opens the door to numerous threats against privacy. Besides the commercial purposes of disclosing and exchanging online information (e.g., selling the history of websites visited by users), endangering users' privacy also leads to the restriction of their fundamental right to freely express their ideas and access to information. Considering the history of mass surveillance programs [2], Internet users such as journalist, whistle-blowers, or people living in oppressive regimes cannot exchange information without fearing the risk of prosecution.

In response to this, the research community has proposed over the years several methods to improve online privacy, and to allow users to retain control over the data they share on Internet [3]. However, since the real-world adoption of such systems is bound to the performance

offered to end users, only a handful of them have gained enough deployment to achieve attractive conditions for daily use (e.g., web browsing). Currently, the onion router (Tor) is the most popular anonymous communication system serving more than two million daily users. The main objective of Tor is to hide the identity (i.e., IP address) of Internet users and to prevent third parties from linking the communicating partners. However, despite the continuous acceptance and development of Tor, several weaknesses on its design produce performance degradation that may drive away existing users and discourage new ones to join the network. Furthermore, the popularity of Tor has also captured the attention of attackers wishing to deanonymize its users. These factors highlight the necessity of providing a more compelling system for both new and existing users, which, in turn, would increase the security of all users as a result of enlarging the *anonymity set*. Giving its enormous relevance and continuous deployment, the contributions of this dissertation are mainly oriented to Tor. However, several of the proposals here exposed are applicable to other systems with similar design characteristics.

To provide anonymity, Tor operates as an overlay service in which users communicate with their destinations via a single virtual path (called *circuit*) composed by typically three volunteer nodes. When packets transverse several intermediaries to reach the destination, security properties of the communication rely on the integrity of each forwarder. Thus, Tor can ensure anonymous communication while no more than one intermediate node is malicious. In this dissertation, we propose to upgrade Tor to a multipath scheme in order to boost both the performance and anonymity properties of the system. Specifically, we present routing design alternatives for allowing Tor to transmit data using multiple paths at transport and overlay level. The former is designed to tackle the suboptimal usage of bandwidth in Tor, and the latter aims to limit the information that an adversary can use for an attack. Though, from the security perspective, it may not seem compelling to increase the number of nodes to build more circuits for forwarding data (because we would increase the number of exposure points that an attacker could compromise), we will demonstrate in this thesis that when an adversary, still controlling certain nodes, has a limited view of the data he can exploit, attacks to users' privacy become significantly less harmful. The formal definition of the adversarial model considered in the scope of the attacks and defenses of this thesis is presented in Section 2.4.1.

As for many other communication systems, the migration from this single-path scheme to more sophisticated pathing paradigms represents a natural evolution in order to provide enhancements in terms of reliability, performance, and resources optimization [4–6]. In particular, it has been shown that increasing the number of available paths (i.e., multipath) between two communicating parties can enlarge bandwidth, provide redundancy, optimize the routing, and secure the communication [4, 6–8]. We believe that these factors also represent a strong motivation for the investigation conducted in this thesis — the advantages of incorporating multipath techniques in Tor.

In this dissertation, we present novel *multipath routing techniques* for anonymous communication systems, particularly Tor. To this end, we start conducting a comprehensive exploratory

analysis to identify the implications of including the multipath scheme in the Tor design space. Based on this analysis, we design and evaluate novel multipath-based techniques that enhance both the *privacy* and *performance* levels of Tor users.

Despite being conceived to individually address performance or privacy issues, the contributions presented in this thesis are orthogonal to each other, and thereby, their synergy contributes to develop a significantly boosted anonymous communication system on both aspects.

1.2 Research Question and Objectives

In this thesis, we seek to answer the following research question:

"How could multipath routing techniques improve the privacy and performance of Tor users?"

The objective of this dissertation is to identify where and how multipath routing is applicable in the design space of Tor for boosting their performance and privacy properties. It is well known that improving privacy comes at the cost of degrading performance [9]. Therefore, our objective is to address each one of these aspects independently without noticeably harming the other one. We aim that the techniques here presented are orthogonal to each other, which, in turn, leads to enhance both the privacy and performance of the overall system.

At first glance, the design component subject to be enhanced with multipath is the currently single-circuit scheme. Therefore, prior works have focused primarily on this aspect for improving exclusively performance of Tor clients [10, 11]. Nonetheless, Tor's design is complex [12] and there are more aspects where multipath techniques can bring advantages in terms of both privacy and performance. Hence, our first objective to extensively explore the Tor design space in order to assess the viability of multipath on each component.

Once we have identified the opportunities for multipath in the Tor design, the next goal is to propose novel techniques addressing each specific aspect. Regarding performance, the objective is to make use of the already known advantages of multiple paths (e.g., enlarging available bandwidth and better load balancing). On the other hand, for privacy enhancements, the goal is to counter traffic-analysis threats by not exposing all observable information on a single path, but over a dispersed structure, in which adversarial exploitations are less harmful.

Another objective of this thesis is to ensure, to the best of our ability, that our novel multipath routing techniques are not severely unfavorable for other elements of Tor. The goal is to minimize the probability of novel emerging surface attacks, which, in turn, would make our proposals unattractive for adoption in the real Tor system. Moreover, we also assess whether our contributions — designed to address specific weaknesses of Tor — can extend their benefits to further existing issues (e.g., other anonymity attacks).

1.3 Contributions

The goal of this thesis is to design advanced multipath routing techniques to enhance anonymity and performance of anonymous communication systems (i.e., Tor). To address the challenges raised from this main objective, we next provide a summary of the contributions of this dissertation:

1. Due to their complex designs, anonymous communication systems can be enhanced with multipath routing at different levels. Thus, our first contribution comprises a comprehensive taxonomy of these systems considering the incorporation of multipath capabilities. After this systematic analysis, we conducted evaluations of existing multipath-based realizations and propose new approaches to fill gaps in the design space. Our evaluation focuses on the corresponding implications regarding performance and anonymity, and serves to outline the techniques used in our next contributions.

2. As a first outcome of our proposed taxonomy, we present a novel design alternative to address performance unfairness in Tor. To allow for low-latency anonymous communication, Tor relies on TCP for transporting data across circuits. Between two consecutive nodes, a single TCP connection carries multiplexed data from multiple users. This leads to an unfair allocation of bandwidth to Tor traffic as several circuits would receive as little resources as other concurrent applications serving a single client. In response to this, we have identified that implementing the multipath paradigm in the transport design of Tor can increase the available bandwidth for Tor traffic to a fairer extend. Instead of proposing to entirely redesign the Tor transport design, our solution comprises the utilization of multipath TCP as an *out-of-the-box* transport protocol between any pair of nodes. Our evaluation at different scales shows that our easy-to-deploy solution decreases download times for Tor users in average by 15%. Furthermore, we identify potential anonymity risks emerged from our design, and propose mitigation strategies.

3. We also identified that multipath routing at circuit level can be used to counter *website fingerprinting (WFP)* attacks. Thus, one of the major contributions of this dissertation comprises an effective defense technique to counter WFP attacks based on splitting traffic over multiple paths. In WFP, the adversary — located in a position where the user's IP address is visible — analyses patterns of the exchanged traffic in order to identify the content (i.e., visited website) of the anonymous communication. To run WFP in Tor, an attacker only requires to participate as volunteer node at the entry position to the network. This means that anybody without extraordinary resources is able to evaesdrop the traffic needed to mount the attack. Our countermeasure is built upon the idea of employing multiple entry nodes instead of one at the overlay level. This implies to craft a traffic distribution to significantly disturb patterns that an attacker (i.e., a malicious entry node) can use to perform WFP. Our defense drastically reduces the effectiveness of WFP attacks from more

than 95% accuracy to less than 16% under real-world Tor conditions. In addition to the offered protection, our defense is incrementally deployed in Tor and its real-world adoption is feasible because the required changes are backwards compatible and the introduced overhead is marginal. Even though our defense does not cause severe performance degradation, we conduct a further refinement of our traffic distribution mechanism to improve the throughput experienced by Tor users. This is achieved by considering a metric that responsively reports the best conditions of the paths on which a certain portion of traffic must be sent. We show that this enhancement to the traffic-splitting technique improves web download times to an extend that Tor users do not notice differences compared to the regular unprotected system. With this, we aim to propose an integral system that can boost the privacy Tor clients while offering attractive performance.

4. The last contribution comprises an analysis to determine whether our effective WFP defense can extend their protection against *end-to-end correlation attacks*. In this attack, the adversary has a global view of the traffic entering and leaving the Tor network. With this, he can infer correlations to discover the user and the destination of a certain communication. We analytically determine that our traffic-splitting mechanism has the potential to also mitigate end-to-end correlation. Then, we conduct an evaluation to assess the effectiveness of the multipath scheme, initially proposed as WFP defense, to counter the most recent and powerful end-to-end correlation attack. We show that by splitting traffic through only two paths, the correlator observing a single exposure point on the entry side, and the corresponding exiting traffic, drastically looses the ability to perform correct correlations.

1.4 Outline

The structure of this thesis is organized as follows. Chapter 2 introduces the foundations of anonymous communication systems. Specifically, we present the relevant design aspects of Tor and the existing anonymity threats.

Chapter 3 presents the prior work related to the contribution of this thesis. Then in Chapter 4, we introduce our taxonomy for multipath anonymous communication systems and present our alternative transport design to increase fairness in Tor.

In Chapter 5, we propose a novel countermeasure against WFP attacks. We start by evaluating several traffic-splitting strategies for our defense. Then, once the best splitting scheme is found, we conduct an extensive realistic evaluation to assess the effectiveness of our defense against state-of-the-art attacks. We also include in this chapter our preliminary evaluation to determine whether our WFP defense can also provide security against end-to-end correlation attacks.

Chapter 6 covers the challenges and limitations of the presented contributions. Furthermore, we sketch further possibilities of multipath within other design aspects of Tor. This thesis

concludes in Chapter 7 with a final discussion, analysis of the response to the research question, and a description of future work.

1.5 Publication List

Most of the content of this dissertation is based on the specific author's work presented in the following scientific publications.

1. **Wladimir De la Cadena**, Daniel Kaiser, Asya Mitseva, Andriy Panchenko, and Thomas Engel. Analysis of Multi-path Onion Routing-Based Anonymization Networks. *In Proceedings of the 33rd Anual IFIP Conference on Data and Applications Security and Privacy (DBSec)*. Springer, Charleston, SC, USA. 2019.

2. Jan Pennekamp, Jens Hiller, Sebastian Reuter, **Wladimir De la Cadena**, Asya Mitseva, Martin Henze, Thomas Engel, Klaus Wehrle, and Andriy Panchenko. (Poster) Multipathing Traffic to Reduce Entry Node Exposure in Onion Routing. *In Proceedings of the 27th annual IEEE International Conference on Network Protocols (ICNP)*. IEEE, Chicago, IL, USA. 2019.

3. **Wladimir De la Cadena**, Asya Mitseva, Jan Pennekamp, Jens Hiller, Fabian Lanze, Thomas Engel, Klaus Wehrle, and Andriy Panchenko. POSTER: Traffic Splitting to Counter Website Fingerprinting. *In 26th Conference on Computer and Communications Security (CCS)*. ACM, London, UK. 2019.

4. **Wladimir De la Cadena**, Asya Mitseva, Jens Hiller, Jan Pennekamp, Sebastian Reuter, Julian Filter, Thomas Engel, Klaus Wehrle, and Andriy Panchenko. TrafficSliver: Fighting Website Fingerprinting Attacks with Traffic Splitting. *In Proceedings of the 27th ACM Conference on Computer and Communications Security (CCS)*. ACM, Virtual Event, USA. 2020.

5. **Wladimir De la Cadena**, Daniel Kaiser, Andriy Panchenko, and Thomas Engel. Out-of-the-box Multipath TCP as a Tor Transport Protocol: Performance and Privacy Implications *In Proceedings of the 19th IEEE International Symposium on Network Computing and Applications (NCA)*. IEEE, Cambridge (MA) - Virtual Event, USA. 2020

Work done in collaboration. The listed publications are result of collaborative work with the other co-authors. This dissertation does not present other author's contributions of the listed publications. Evaluations for Chapter 5 leverage implementations and data resulting from a wide collaboration. Please refer to the original paper [13] for further details.

Human beings aren't forever born the day their mothers bring them to life; instead they must be reborn time and time again.

Gabriel García Márquez

2 Background

2.1 Introduction

For a communication to be anonymous, the identities of one or both parties must be kept hidden from the respective counterpart or external agents. Here, the value to preserve is not the content of the communication (it can be protected via encryption) but the identities of the involved parts. To this end, data exchanged between two ends is primarily done through intermediate hops. The simplest anonymization technique consists on using a single hop (e.g., a VPN service) between a user and its destination. The anonymity provided by this method, however, depends on the integrity of the intermediary hop. Therefore, effective anonymization techniques employ multiple hops between users and their destinations. The resulting hop-based overlay structure is known as *anonymization network*. Despite this method prevents any hop to directly discover who converses with whom (each hop knows only the predecessor's and successor's IP addresses), an adversary — a malicious intermediary or an external agent with wiretapping capabilities — can perform *traffic analysis* to infer the identities of senders and receivers. For this reason, anonymization systems are designed to withstand traffic analysis at different levels and performed by distinct adversarial models. In the following, we summarize the existing anonymization techniques and their main properties.

Mix Networks. Proposed by Chaum in 1981 [14], mix networks relay traffic through a chain of intermediate nodes known as *mixes*. When forwarding data from several users, a mix delays, shuffles, and re-encrypts the data before passing it to the next node. This prevents that an adversary correlates, bit-wise or timing-wise, the traffic entering and exiting a mix. Thus, even

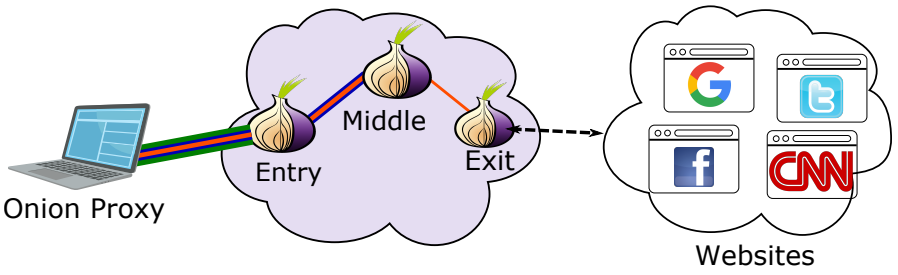


FIGURE 2.1: Overview of Tor operation

a global adversary located along all links of the network is unable to break the anonymity. In order to protect against malicious mixes, a user can send the message through multiple mixes because one honest mix along the path is enough to achieve anonymity. The effectiveness to withstand powerful traffic analysis motivated the implementation of this technique in systems like Mixmaster and Mixminion [15]. Unfortunately, the number of operations performed within each mix produce unacceptable latency for daily online activities such as web browsing, and therefore, their application is limited to high-latency use cases such as e-mail.

DC-Nets. Dining cryptographers (DC) [16] networks rely on cryptography to prevent traffic analysis for discovering the users' identities. Each member of the network uses a secret key to encrypt his message, and then all messages are distributed via a broadcast channel in rounds. This technique is highly resistant to traffic analysis, but unfortunately requires large computational capabilities, which, in turn, are not easy to achieve for low-latency applications (e.g., web browsing from a mobile device). Hence, this technology has not been widely implemented and developed over the years.

Onion Routing. This the most popular and deployed anonymization technique. Its most widespread implementation, Tor [17], serves more than two million daily users. Tor is designed to provide anonymity for low-latency applications, e.g., web browsing and interactive applications. Considering its acceptance, and that the vast majority of online activities demand low latency, Tor has become the de-facto tool for providing anonymity on the Internet. Unfortunately, Tor faces several challenges from both the performance and privacy perspectives. For the former, some designs aspects of the used transport protocol cause degraded performance [18]. For the latter, several anonymity attacks have been demonstrated to be effective against Tor users [19–22]. Thus, proposals to improve the performance and privacy properties of Tor are crucial for attracting more users in order to enlarge the anonymity set and comprise a more secure system. Given the relevance of Tor, the scope of this thesis is primarily delimited to this system. Nevertheless, our proposals are applicable to other anonymization systems that share design details with Tor. In the following of this chapter, we detail the relevant design aspects of Tor and the existing privacy threats.

2.2 Tor Overview

Tor is the most popular low-latency anonymization network designed for TCP-based applications. The main objective of Tor is to hide 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 to the destination. This virtual path is referred to as a *circuit* and is typically composed by three nodes known as *onion relays (ORs)* [23]. 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 [23]. While forwarding user traffic, each OR on the circuit removes (or adds, depending on the direction) a layer of encryption (see Figure 2.1). This ensures that none of the ORs on the circuit knows the source, the destination, and the content of the communication. Along a circuit, user traffic travels encapsulated in fixed-size units referred to as *cells*. Based on the data they carry, Tor cells can be categorized in two main groups: *control* and *relay* cells. While the control cells convey management information to the ORs to build and maintain circuits, the relay cells transfer the user data.

The ORs are run by volunteers who determine the amount of bandwidth they are willing to share. The position of an OR in a circuit has implications about the type of information that can be accessed (e.g, knowing destination at exit OR) or be more attractive for attackers. Therefore, there are specific requirements to run a relay as entry, middle, or exit OR . When an OR becomes active in Tor, it starts by default as middle OR. The only requirement for this is to be reachable from any host on the Internet. Thus, ORs in the middle position are currently the most numerous and have an overall larger bandwidth capacity than entry or exit ORs [24]. Since the ORs at exit position make the connection directly to the web server, their Internet service providers (ISPs) can recognize the websites visited by Tor users, which involves a disclosure of high-sensitive data. For this reason, an OR operator must explicitly set this option in the relay's configuration if he desires to run the relay at this position. On the other hand, to become an entry OR, higher requirements are necessary because this node has direct access to the client's IP address. In this case, the OR must be stable for a longer period of time and offer more bandwidth than ORs in other positions.

For censorship circumvention, Tor offers a set of alternative ingress points to users that are blocked to make connections to IP addresses of public ORs . These non-published ORs are referred to as *bridges*, and their access can be provided out-of-band (e.g., via e-mail or friend-to-friend).

Given its complex design, Tor is studied under several research directions [18]. We next detail the Tor design from two perspectives: privacy and performance.

2.3 The Tor Design

As for any other privacy enhancing technology, the anonymity offered by Tor comes at the cost of a reduced performance compared to a non-anonymized communication. During the communication, the overhead is introduced by the multihop-based data forwarding (ORs can be geographically distant), the costly multi-layer encryption and decryption operations, and the limited bandwidth capability offered by the ORs. For this reason, Tor requires an optimized transport design, a tailored control and congestion management, and an efficient OR selection algorithm. We next elaborate on each of these aspects.

2.3.1 Relay and Circuit Selection

The topology of Tor is complex and dynamic [25]. There are ORs located all around the world offering highly diverse bandwidth capabilities to Tor traffic. Therefore, the criteria to choose and use ORs for building circuits are critical for providing acceptable performance. A Tor client chooses the ORs for a circuit based on their availability, flags, and offered bandwidth. When a volunteer decides to run an OR, it takes around 40 minutes until their status and capabilities are validated. After launching the Tor application, the OR must wait that a set of special nodes, called directory authorities, update a consensus document where all ORs are listed. This document contains various details of each OR such as allowed flags (e.g., the exit OR flag), offered bandwidth, geographical location, and IP address. Once the validation round of the directory servers is performed, all fresh ORs are included in the consensus and they earn the *active* flag. Further flags are assigned as more conditions are fulfilled. The *stable* flag comes when an OR has been active at least the median time among all other ORs. And the *fast* flag is assigned to ORs that are in the top of relays with a certain speed defined by the authorities.

Tor aims to distribute traffic as balanced as possible among all ORs in the network. For this reason, the relay selection is done in a weighted-random manner where the OR's bandwidth gives it more chances to be chosen by a client. Besides considering bandwidth, as a result of emerging attacks and investigations [26], further constrains to the relay selection process have been included. These conditions vary depending on the relay's position within the circuit. Firstly, a router can only participate once in the circuit (e.g., it cannot be entry and exit simultaneously). Furthermore, any pair of ORs in a circuit must not be on the same /16 subnet, nor be maintained by the same operator. Middle ORs are the easiest-to-run relays. They only need to offer the minimum allowed bandwidth of 75 KBytes per second. For becoming an exit ORs, the operator must specifically activate the *ExitRelay* option. This position is sensitive as it directly access to potentially-monitored websites on behalf of anonymous clients. Therefore, these ORs are commonly under constant watching of prosecutions agencies, and it is thus recommended that they are run by universities or research centers. Special attention require the entry ORs since they are in direct connection with the clients (i.e., entry ORs know users' IP addresses). By default, a Tor client chooses the same entry OR from a list of candidate relays with the *fast* and *stable* flags.

This selection remains unaltered for a long period of time (around 60 days). The list of possible entry ORs (known as *primary* list) remains fixed after filtering out slow and geographically-closer ORs.

With this OR selection criteria, a Tor client creates preemptively — even before streams need to be attached — several circuits and records the corresponding circuit build times (CBTs). Then, when user data is ready to be sent, the Tor client discards the circuits with the slowest CBTs and attaches the stream to the circuit that is expected to have the best performance. Generally, any circuit has an overall lifetime of 10 minutes. After that, a new circuit is used for stream attachment. It is also possible that a Tor client leverages a Tor control framework (e.g., the Tor control protocol) to ignore the standard OR selection process and use customized circuits with specific desired nodes.

2.3.2 The Tor Transport Design

At the transport level, Tor operates as an overlay service for forwarding TCP traffic between the user and its destination. To achieve this, each OR in the circuit maintains a TCP connection with its successor and predecessor nodes. On circuit creation, the client establishes a first TCP connection with the chosen entry OR. After that, an instruction cell to extend the communication towards the next OR is sent. Then, next ORs, if not yet established, start their own TCP connections with their subsequent node. Finally, the exit OR reads the destination address in the instruction cell, and performs a TCP connection with the destination. The user can then exchange data through the Tor network using the TCP connection with the entry OR. If multiple users choose the same ORs on their circuits, these ORs reuse the existing connections to multiplex the traffic of all other involved users. This design choice implicitly contributes to users' privacy because traffic of each user travels hidden in an anonymity set. However, the bandwidth of the shared link is unfairly limited as it was intended for one single user.

When data arrives to an OR, it is demultiplexed and stored into per-circuit queues. After the corresponding cryptographic operations, packets of each queue are multiplexed and placed into the output kernel-level TCP buffers for forwarding them to the next OR or OP. The priority and order in which traffic from different circuit queues is dispatched into the next TCP connection depends on the scheduling Tor configuration. Currently, relay operators can choose among three options. The vanilla scheduler, the first one implemented in Tor, assigns cells to each circuit in a round robin manner. A noticeable improvement was included with the kernel-informed socket transport for Tor (KIST) scheduler [27, 28]. This technique uses TCP-kernel information to define when and how to distribute the cells into circuits. For operative systems that do not allow to access to kernel-level information, the alternative KISTLite scheduler works as an approximation that artificially estimates the kernel status.

2.3.3 Reliability, Flow, and Congestion Control

Tor, per se, does not implement congestion control nor reliability for the transmitted data. The existing OR-to-OR connection is the primary means to regulate congestion and to guarantee end-to-end (between OP and the exit OR) correctness and in-order packet delivery. In addition to this flow control provided in the OR-link level by TCP, Tor implements two window-based flow-control mechanisms at circuit and stream level respectively.

At circuit level, the maximum allowed number of in-flight cells within a circuit at any moment is regulated using two windows: the package and the deliver window, both initialized by default to 1,000 cells. Every relay cell received in an edge node (the OR or the OP depending on the stream direction), decrements by one its delivery window, and when a decrement of 100 cells is registered (the delivery window reaches 900), a control cell called `SENDME` is sent back as acknowledgment to the corresponding counterpart. A `SENDME` received at the other edge (the OR or the OP) increments its package window by 100 and greenlights the delivery of other 100 cells. When a package window reaches zero, the OR or OP refrains to process more TCP packets associated to that circuit and to send more relay cells before receiving a `SENDME` cell.

An analogous mechanism operates at stream level. The differences lie on the window sizes (initialized to 500) and the `SENDME` sending pattern (after 50 received relay cells). With this, the maximum amount of in-flights cells within a stream is capped to 500.

2.4 Anonymity Attacks against Tor

Due to its popularity, Tor is an attractive target for attacks aiming to discover the identity of its users. Numerous attacks have been investigated and proposed. Although many threats rely on powerful adversarial models — not specified in the original Tor design, many others have shown successful results considering a weak adversary such as a local passive observer.

A comprehensive taxonomy of existing attacks is presented in [18]. One research direction comprises *end-to-end correlation* attacks [29–31]. Here, an adversary, with a global view of the Tor network, monitors traffic at ingress and egress points to correlate patterns and identify users and destinations of the communication. Other family of attacks focuses on congesting or blocking ORs [32, 33] in order to prevent them to serve Tor users. Consequently, the anonymity set size of both routers and users may decrease and the security of the overall system is diminished. Similarly, denial of service (DoS) attacks [24] also represent an anonymity threat that can affect Tor at different levels. This includes from sybil-based attacks targeting single ORs, to large-budget state-level adversaries overloading a large amount of ORs using cloud services. Another highly-relevant research field comprises *website fingerprinting* (WFP) attacks. Here, the adversary uses merely network-level traffic information to discover the visited website (i.e., the destination).

A major contribution of this thesis is the design and evaluation of an effective multipath-based countermeasure against WFP attacks. Moreover, we study whether this defense can extend the protection to end-to-end correlation attacks. Hence, we next detail these traffic-analysis threats and the respective adversarial models.

2.4.1 Threat Model

Tor was not designed to resist global adversaries with a general view of the entire network. Contrary, it is assumed that an adversary has access to a limited portion of the network, which does not comprise observing simultaneously both edges of a circuit . Thus, the anonymity of a Tor user is considered compromised when an adversary accesses to the traffic at the entry and exit ORs. This is the adversarial model considered in end-to-end correlation attacks. Here, the attacker can discover user and destination by only observing traffic patterns at both ends (*passive*), or by watermarking/injecting patterns on one side and reporting it on the other (*active*). Evidently, the former is more dangerous as it is harder to be detected and requires less capabilities.

Among all the possible adversaries applicable to the Tor's threat model, one particularly critical — because of the weak required capabilities — is the *passive local observer*. This is the adversarial model considered for website fingerprinting attacks. As passive, the adversary is not able to decrypt, modify, or interrupt the communication; and as local, he is located where he can eavesdrop the traffic exchanged between the user and the Tor network. This bounds the possible attackers to: local network administrators, ISPs, entry ORs operators, and any other network sniffer agent placed along the user-entry OR connection. It is important to mention that among the listed possibilities, malicious entry ORs are particularly dangerous because anyone with moderate technical capabilities can run a volunteer OR on that position and easily become an adversary. This dissertation focuses on defeating malicious entry ORs mounting WFP attacks. Though, we also discuss potential enhancements to also offer protection against ISP-like adversaries.

To sum up, we have described the two adversarial models considered in the scope of this thesis. First, for addressing WFP, we consider a malicious local passive observer (i.e., entry OR). Second, when analyzing end-to-end correlation, we assume an adversary capable to observe traffic at both ends of the communication (i.e., entry and exit ORs).

2.4.2 The Website Fingerprinting Attack

Despite Tor promises to hide users' identities from passive a local observer e.g., a malicious entry OR — one of the weakest adversaries in the attacker model of Tor [13], it cannot conceal information regarding size, direction, and timing of packets. This has opened the possibility to local adversaries for mounting *website fingerprinting* (WFP) attacks. In this attack, a passive local observer scrutinizes traffic patterns of the encrypted connection between the client and the entry OR of the anonymization network. WFP is a type of traffic analysis attack, which aims to identify

the content (i.e., the visited website) of anonymous user connections by passively observing patterns of data flows. Over the years, multiple studies have systematically demonstrated the continuously improved effectiveness of WFP attacks [34, 35].

An adversary develops this attack as a *machine learning (ML)* classification problem. First, he visits multiple times the set of pages he wants to monitor and logs the corresponding traffic records (i.e., the timestamps, sizes, and directions of sent/received packets), which are refereed to as *traces*. Then, he detects distinguishable traffic patterns of each website (*fingerprints*), and extracts *features* from all traces to train a *classifier* that will be later used for identifying the victim's eavesdropped traffic. A WFP attack is commonly deployed in two settings: *closed world* and *open world*.

In the closed-world scenario, the user is allowed to visit a limited set of websites and the attacker can train on all of them. This setting typically denotes a classic balanced multi-class classification problem. Thus, accuracy — the number of correct predictions among all predictions — is the metric used to evaluate the attack's performance. Although this scenario is unrealistic, it allows to make fair comparisons between attacks.

In the open-world setting, the set of websites that a user may visit is unrestricted (this reflects a more realistic Internet browsing) and the attacker trains only with the set of websites he is interested in. The set of pages that the attacker can train on is referred to as *foreground*, and the set of pages unknown for the attacker is referred to as *background*. In the open-world setting, the primary goal of the attacker is to identify whether a page is part of the foreground or background class (binary classification). To measure the success of predicting both classes, true positive rate (TPR) and false positive rate (FPR) are the commonly used metrics. However, when the background and foreground set sizes are heavily unbalanced, precision and recall are the preferred metrics [36].

For comparison with prior work, this thesis respects the assumptions commonly done when investigating WFP. We consider that the attacker can isolate the traffic corresponding to a single page load and that WFP is only applied to the index page of a website. Moreover, we assume that the adversary has sufficient computational power to train several fingerprinting techniques on large training datasets.

2.4.3 End-to-end Correlation Attacks

When an adversary accesses to traffic at the entry and exit ORs of a certain circuit, it is assumed that this connection is compromised because the adversary only requires to correlate flows entering and exiting the network, and then, pair those matches with origin (IP address is visible by the entry OR) and destination (IP address is visible by the exit OR).

In this attack, the adversary obtains a certain number of ingress and egress encrypted traffic flows. Then, he aims to match flows on both sides corresponding to the same origin-destination

pair. Encryption avoids that this is done by looking at the content. Therefore, the attacker can only compare characteristics such as packet sizes, timestamps, and traffic direction.

There exist several techniques to correlate traffic patterns obtained from two ends of a Tor circuit. One option is to rely on the mutual information [37] metric to assess the dependency between two random variables assigned to each traffic flow. Another possible statistical measure to use is the Pearson coefficient [38]. This well-known metric can be also extracted from a mathematical representation of the traffic flows, and thus, be compared to assess the correlation level. Analogously, the cosine similarity [39] and the Spearman correlation [40] metrics have been also used for finding correlations in traffic flows.

Although these statistical metrics are easy to compute and extract from traffic traces, they are not always optimal for Tor traffic because data flows in real networks are noisy, and thus, patterns for checking similarities become hardly identifiable. Therefore, a recent work [31] has proposed to use deep learning (DL) to find correlation matches. Here, a convolutional neural network (CNN) model classifies whether a certain pair of ingress/egress traffic flow was originated on the entry/exit ORs of a certain Tor user. Thus, this attack is conceived as a two-class classification problem where the model is fed with raw input traffic flows pairs represented by sequences of packets' sizes, times, and directions. This data is labeled as *positive* if a traffic pair belongs to the same user, and as *negative* if no correlation exists. DL-based models are known for their adaptability to learn patterns at different levels (i.e., big and small details of the input data can be recognized) and for not requiring sophisticated feature extraction processes (i.e., the model can use as input a raw traffic representation).

2.5 Summary

We have covered the relevant concepts used in this dissertation. After introducing the existing anonymization techniques, we presented the design details of the most popular system — Tor. We started describing the various components of Tor. We described the process for selecting relays and building circuits. This is an important element considered in the contributions of this thesis, since one of our proposals is to upgrade Tor to a multipath structure, in which more nodes are required to build extra circuits. Thus, it is essential to understand the implications from setting up a relay on any position.

Then, we detailed the Tor transport design and the employed flow and congestion control techniques. Another major contribution of this thesis proposes an alternative protocol to be used at the transport level. Hence, the concepts covered in this section are important to discover the weaknesses that can be mitigated with the novel proposed multipath-based transport design.

Lastly, we described the relevant aspects regarding attacks against Tor users' privacy. Specifically, we detailed the threats and adversarial models considered in the scope of this dissertation, the website fingerprinting and the end-to-end correlation attacks.

Have no fear of perfection; you'll never reach it.

Salvador Dalí

3 Related Work

3.1 Introduction

We next revise the relevant related work for this dissertation. For performance improvements, we focus on prior works addressing alternative transport designs and proposals to migrate from the current single-circuit scheme to a multi-circuit communication paradigm. Then, we focus on the specific considered privacy threats, the website fingerprinting and the end-to-end correlation attacks. Specifically for WFP, we also survey the state-of-the-art defenses, which will be then compared with our proposed countermeasure.

3.2 Improving Performance in Tor

Providing anonymity in Tor comes at the cost of performance loss for the users. For instance, the inefficient transport design, and the dynamic nature of the system — anybody can join the network by running an OR or leave the network at any time — cause that 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.

Several components of the design space of Tor can be improved for a better performance of end users. A significant amount of research has focused on exploring a variety of relay selection algorithms, e.g., by trying to avoid congested ORs [41], considering the geographical location [42] or bandwidth [43, 44] of chosen ORs. Another group of works [45–47] 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 [48, 49], propose improved congestion control algorithms [47, 50] or even replacing the underlying transport protocol [51, 52] to optimize the utilization of available bandwidth in Tor. In contrast to this thesis, these works did not evaluate the effect of multipath techniques in Tor. Nevertheless, these proposals complement our work and their coexistence can further improve the performance and harden the security of Tor.

The scope of this dissertation is oriented to apply the multipath routing paradigm for improving privacy and performance of Tor users. As our investigation will show in Chapter 4, this can be done in several components of the design space. Specifically, we address the benefits of using multipath routing at transport and circuit level. Thus, we next describe the relevant previous works on these two research areas.

3.2.1 Alternatives Tor Transport Designs

As stated before, Tor, as part of its privacy-preserving design, specifies that all traffic from different circuits coexisting between a pair of ORs is multiplexed within a single TCP-secured (TLS) connection. Besides preventing one-to-one linkability between users and TCP sessions, keeping a single connection also avoids large processing times produced by multiple handshakes and cryptographic operations. This design choice has been identified as one of the main causes for the performance issues in Tor. Traffic from a single high-volume circuit is able to obstruct a fluid traffic flow from other circuits using the same TCP connection. For instance, a congestion event (e.g., a packet loss) on the high-volume circuit will produce that packets of other circuits are retained on the ORs' buffers until the lost packet gets retransmitted. Furthermore, the shared TCP connection offers bandwidth resources that without Tor would be allocated to a single user, and thus, are considered unfair when transporting multiplexed traffic.

The produced latency is particularly problematic for circuits carrying interactive traffic, in which delayed responses may significantly decrease the web browsing user experience. Another consequence of packets staying longer times in the buffers, is the degradation of the router's capacity for packet scheduling decisions and serving other applications. The intrusion produced by one circuit over another is known as *cross-circuit interference*. And the specific effect of packets retention of one circuit due to the congestion experienced by another one is known as *head-of-line (HoL) blocking*. Although HoL always refers to packets from different sources queued at

buffers, it is important not to confuse the HoL effect produced in Tor by several coexisting circuits, with the blocking that out-of-order delivery or shared buffers on network switches produce. The latter arises from traffic originated from multiple TCP sessions, while the former occurs due to the circuit multiplexing used in Tor.

As alternative Tor transport protocols, several works [47, 53–55] suggest increasing the number of TCP connections between ORs to minimize the coexistence of traffic from multiple circuits in a single connection. Another group of studies [52, 56, 57] proposes to replace the underlying protocol (e.g., with UDP), and profit from alternative congestion control mechanisms implemented outside the transport layer.

Attempting to reduce the unfairness of sharing a single TCP connection between ORs, and to simplify the tasks that each relay performs, Viecco proposed UDP-OR [52]. Here, any pair of ORs communicate using UDP instead of TCP. Reliability and congestion control are achieved by tunneling the data through an OP-to-exit OR TCP session. Since acknowledgments must transverse the entire circuit to reach an end point, an enormous latency is expected especially for congested scenarios.

Reardon and Goldberg [47] were the first that experimentally demonstrated at small scale the negative effects of cross-circuit interference. The authors suggested to use a secured datagram-based protocol to tunnel TCP (TCP-over-DTLS) communication between ORs. This design avoids that packets get blocked on buffers because in UDP the OR does not wait for acknowledgments to continue sending data. For providing reliability and congestion control between ORs, a separate user-space TCP connection is created for each OR-to-OR link. Two major concerns impede deploying this approach in Tor: *(i)* there is, to the best of authors' knowledge, no TCP user-space implementation with a licence compatible to be used in Tor. And *(ii)*, the unacceptable high CPU capacity required for running the transport protocol at user-space.

A more tailored proposal for Tor, Torchestra [55], opens two TCP connections between ORs, one intended for circuits with high-volume traffic, and another for lightly-loaded circuits. Unfortunately, the presented small-scale evaluation and a suboptimal effectiveness of the algorithm to classify the traffic load on the circuits as heavy or light, are major concerns to include this design in Tor.

Al Sabah and Goldberg [54] proposed to assign a kernel-level TCP connection per existing circuit (PCTCP). However, this proposal increases the aggressiveness of the traffic on parallel connections, and opens the door for socket exhaustion attacks [53] due to the one-to-one circuit-to-connection mapping. PCTCP is not backwards compatible because channels are secured with IPSec instead of TLS and a new Tor cell format is required.

Tackling specifically HoL blocking, Nowlan et al. [56] disabled the in-order delivery of TCP packets between ORs. With this, packets are sent regardless of possible missing data. Reordering is implemented at circuit-level and the channel is secured with an unordered version of TLS (uTLS). Unfortunately, the still shared I/O buffers in the ORs, and the effort for providing reliability on an upper layer, cause that performance is marginally improved.

Geddes et al. [53] proposed IMUX, an approach where the number of TCP connections between ORs is dynamically determined by the number of circuits. The objective is to avoid socket exhaustion attacks possible in PCTCP and Torchestra. Since the connections are created or closed depending on the existing circuits together with a configured threshold, a mapping 1:1 between connections and circuits is not always possible. In general, there are n circuits to occupy m connections. Therefore, IMUX implements three scheduling mechanism to dispatch cells of each circuit among the available connections. Although round robin scheduling outperformed other techniques, the obtained improvements were still modest. Furthermore, since the maximum number of connections is capped, it is likely to have more circuits than connections between a pair of ORs, and it is still possible that cross-circuit interference appears.

The latest proposal to refurbish the Tor transport design is BackTap [57]. The authors considered that the main problem for performance are the cells stored for long time on the per-circuit queues. In response to this, BackTap implements per-hop congestion control on the user space and relies on UDP as transport protocol. This work reported promising improvements, however, as for other similar proposals, the incompatibility with the current system, and the difficulties of migrating to UDP, discourage Tor developers to adopt this approach.

Despite the numerous alternatives proposed to redefine the Tor transport design [18], none of them can be practically deployed in Tor. We identify the following major concerns for their practical adoption: *(i)* replacing the underlying transport protocol is backwards incompatible and requires updating the entire system (e.g., BackTap). *(ii)* proposals cannot be used out-of-the-box as they usually require major adjustments for running in Tor (e.g., uTor). *(iii)* showed performance gains do not merit the proposed complex update to the entire system (e.g, TCP-over-DTLS, Torchestra). One of the main contributions of this thesis is to propose a transport design that can provide performance gains without raising these concerns (see Section 4.7.2).

3.2.2 Multipath in Anonymous Communication Systems

The notion of employing multiple paths is not entirely new in anonymous communication systems. This technique has been also considered in previous theoretical analyses, simulations, and non onion routing approaches. The objectives pursued by those works were: passive attack resilience [58–60], multipath as a means of anonymity [61], and performance improvements [10, 11, 62, 63].

Karaoglu et al. [63] investigated the feasibility of multipath techniques within the Tor network in order to solve performance issues related to congestion and low throughput. The authors propose a multipath routing scenario which emulates the operation of unidirectional multipath TCP [64]. 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.

Applied to high-latency mix-based systems, Serjantov and Murduch [58] proposed that while a packet transverses a mix node, it can also be forwarded to a next node belonging to a different route. The goal is to minimize the exposure to adversaries correlating traffic on two points on the route.

Engelman and Jukan [60] presented a simulation where multiple circuits were used to evade censors blocking the access to the Tor network. Using linear network coding, this approach extends with redundancy the original data, and send chunks of data through several circuits. Then, in case the censor blocks traffic from one circuit, the decoding process allows recovering the full data on the exit OR.

The multipath paradigm was used as a means of anonymization in [61]. The authors designed a system where a sophisticated data fragmentation technique prevents an observer to know the content and identities of user and destination of the communication. In order to know the route of each packet, the messages' fragments contain on their header information about the next node in the path. With this, anonymity and secrecy is provided without using cryptographic keys.

To the best of our knowledge, there exist only three systems have been fully developed and implemented as multipath onion routing-based approaches. Two of these, Conflux [10] and mTor [11], are extensions to vanilla Tor that adapt its traffic management design to utilize multiple circuits; the third, MORE [65], comprises a multipath design over UDP where each cell travels along a different circuit.

In Conflux [10], 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 multipath 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` 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. The authors of Conflux presented results from an implementation that supports only *two* circuits. They showed that performance improvements are particularly noticeable for bridges because they usually have a larger bandwidth than middle and exit ORs. Thus, the capacity of a multipath circuit structure is the sum of the individual capacities of each circuit.

Following a similar approach, mTor [11] was proposed. Here, the multipath structure and cell merging procedure is similar to Conflux. However, the end-points (exit OR or OP) choose one of the multiple circuits according to its current stream-level window value. The *end-point* drops cells to the circuits in a first-in-first-out manner, while their stream window is greater than zero. Furthermore, in mTor, clients tend to choose less-used low-bandwidth middle ORs. The idea is to use those relays with low demand and contribute to load balancing the Tor network.

In a contrasting approach, MORE [65], a new path is freshly created and used for each data cell. Here, 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[1] 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 multipath 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 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.

While Conflux and mTor were designed to improve client-side performance raised from using a possibly single slow circuit, MORE pursued to be resilient against traffic-analysis attacks by switching on-the-fly to a new circuit for each cell.

An important contribution of this thesis is a comprehensive evaluation of the design space of multipath for onion-routing anonymous communication systems. Thus, these approaches are further evaluated and analyzed in Chapter 4.

3.3 Security and Privacy in Tor

Tor promises to hide the relationships between the user and its destination from third parties. Nonetheless, several attacks have been demonstrated to be effective against the anonymity of Tor users. The study of attacks in Tor is a wide research field, there are various adversarial perspectives that may harm Tor. An extensive taxonomy of attacks against Tor is presented in [18]. We next summarize the existing attacks making special emphasis on those within the scope of this dissertation.

3.3.1 Routing Attacks

This family of attacks [40, 66–68] aims to deviate from the secure manner in which Tor clients select and use the routes to their destinations. Specifically, malicious agents gain control over one or more nodes to fake their characteristics in order to make them more likely to appear in target users' circuits. With this, adversaries gain confidence to perform further attacks (e.g., website fingerprinting) or correlate traffic among a compromised path.

3.3.2 Congestion and DoS Attacks

These attacks [24, 32, 33, 69] pursue to block or congest target ORs in order to weak the entire system making it unattractive for existing and new users. Furthermore, tearing down ORs is a mechanism to also deviate from a legitimate route to a potentially maliciously controlled path.

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

3.3.3 Traffic-Analysis Attacks

These attacks have been widely investigated in Tor. The technique used is the exploitation of traffic characteristics to infer source, destination, or content of the communication without breaking the encryption. Usually, these attacks are performed in conjunction with the previous ones. For instance, routing attacks force a client to user certain ORs, and then, traffic-analysis discovers the destination by scrutinizing traffic patterns exchanged over the Tor network.

Depending on the adversarial capabilities, we can categorize traffic-analysis attacks performed by local or global observers. At the same time, they can be passive — they only observe traffic — or active — they can insert patterns (e.g., watermarks) to the traffic flow to ease the analysis. While it is true that global adversaries are not considered in the Tor threat model, and therefore Tor does not offer protection against them, traffic-analysis can be also mounted by a local observer. In this regard, the most prominent threat is the *website fingerprinting attack* [20–22, 70–75]. A major contribution of this thesis comprises protection mechanisms against this threat. Hence, we cover more in the detail the related work regarding WFP attacks and defenses in the next sections.

Another important type of traffic-analysis attacks are the *end-to-end correlation attacks*. Here, an adversary with global capabilities, observes traffic entering and leaving the Tor network in order to pair matching patterns originated by a certain user and its destination. In this thesis, we also assess the effectiveness of our technique — initially indented to withstand WFP — against end-to-end correlation. We aim to demonstrate that our WFP defense can also serve as mitigation against these attacks. Hence, we also survey the corresponding related work.

3.3.3.1 Website Fingerprinting

Next, we briefly review previously proposed WFP attacks and defenses.

Website Fingerprinting Attacks. The first work in applying machine learning for website recognition in anonymization networks was done by Herrmann et al. in 2009 [74]. The authors evaluated WFP in Tor and other single-hop systems such as OpenSSL and OpenVPN. They showed that the recognition rate of a multinomial naive Bayes classifier within a closed-world dataset of 775 sites is more than 94% for single-hop systems and 2.96% for Tor. The main features used in this attack are the occurrences of packet sizes. This explains the low effectiveness in Tor due to its traffic encapsulation in fixed-size cells.

In 2011, Panchenko et al. [70] applied support vector machine (SVM) to the dataset of Herrmann et al. [74] and achieved a classification rate of 54.61% for the Tor system. The authors extracted a diverse feature set that included: number of packets, percentage of incoming and outgoing packets, and traffic direction changes. Furthermore, this work includes an open-world evaluation where a set of five pages was detected within a set of thousands of unknown pages. The recognition rate in this scenario achieved 73% (TPR). Results of this work drew the attention of

the research community, and caused that subsequent works focused on increasing the recognition rate and studying the feasibility of this attack on real-world conditions [71, 76, 77].

In 2012, Dyer et al. [78] presented a comprehensive comparison of classifiers, datasets, and countermeasures using an enhanced feature set. Unfortunately, the newly introduced features did not significantly improved the attacks' effectiveness.

Two disruptive attacks [22, 72] based on the k-nearest neighbors (k-NN) classifier were presented by Wang et al. After data preprocessing, the authors extracted a large set of around 2,000 features to achieve a recognition rate of more than 90% within a closed world of 100 websites. In the open-world setting, the authors achieved a TPR of 85% and a FPR of 0.6%, recognizing a foreground set of 100 websites within a background set of 5,000 sites.

To ease the feature engineering involved in a WFP attack, Panchenko et al. [75] presented a feature set that represents a traffic trace as a vector of the cumulative sums (CUMUL) of the packet sizes, considering the sign of each packet size according to its direction. Then, this feature set was enhanced with the number of packets and the sum of packet sizes on both directions. In order to have the same vector length for traffic traces with different number of packets, this cumulative vector is linearly interpolated to 100 equidistant points. The authors achieved an accuracy of more than 91% for a closed-world setting of 100 websites. In the open-world setting, the authors achieved a TPR of 96% and a FPR of 10% recognizing 100 foreground websites within a background set of 9,000 sites. This work also studied the feasibility of recognizing a page at Internet scale by evaluating different background set sizes, and explored how to fingerprint the internal web pages members of a single website.

Hayes and Danezis [21] proposed to use random forests to extract the fingerprint of a traffic trace. They calculated a diverse set of 175 features that included: sizes, direction, concentration and ordering of packets, data rate, and timing statistics. This vector is then associated to a leaf of the forest, which finally composes the fingerprint of the web page. In the closed world, the random forests' output directly provides the detected website label. Conversely, in the open-world case, these features leafs are input to a *k*-NN classifier. This attack is known as k-FP, and on the dataset provided by Wang et al. [72], it achieved an accuracy of 91% in the closed world, and a TPR of 88% with a FPR of 5% in the open world.

Following the latest techniques to solve classification problems, several works proposed to use *deep learning* (DL) to perform a WFP attack. The main advantage of these techniques is that feature engineering is not necessary, which means that the input to the classifier (*model*) can be a matrix representation of the raw traffic packet sequence. This sequence is usually represented as a vector of ± 1s that corresponds to each in/out packet of the trace. In order to keep the same length for all inputs, packet sequences are either zero-padded or truncated to a maximum length.

In the first DL-based approach, Abe and Goto [79] used stacked denoising autoencode (SDAE) as the model, and achieved 88% accuracy for a closed-world setting of 100 websites. In the open world with a background set of 9,000 sites, the authors achieved a TPR of 86% with a FPR of 2%.

Later on, Rimmer et al. [80] evaluated the following three DL models in the context of a WFP attack: SDAE, CNN, and long short-term memory (LSTM). In the closed-world setting of 100 pages, SDAE reported 95.46% accuracy, CNN achieved 97%, and LSTM reached 94%. In the open-world setting with a foreground set of 200 pages and a background set of 400,000, the best model (SDAE) achieved a TPR of 80% with a FPR of 9%. The main disadvantage of the models presented in this work is the requirement of a large number of training instances, which significantly increases the training time.

In 2018, Sirinam et al. [20] designed an improved CNN-based architecture as the model of the neural network classifier. This attack, known as deep fingerprinting (DF), achieves high accuracy without requiring a large number of training instances. DF reported 98.3% accuracy in a closed-world setting of 95 websites. In the open-world setting, this attack achieved 95.7% TPR and 7% FPR for a foreground set of 95 pages and a background set of 20,000 sites. Thus, DF outperformed all previous DL-based attacks. Furthermore, due to its fast training runtime, accuracy, and robustness against countermeasures, DF is considered one of the state-of-the-art WFP attacks (together with *k*-NN, k-FP, and CUMUL).

A hybrid approach, called Var-CNN [35], that combines DL and ML was introduced by Bhat et al. Here, a CNN model called ResNet is fed with: the raw packet sequence, seven precomputed cumulative features (relative to number of packets in both directions), and inter-packet timing features. This ensemble model reported 97.8% accuracy in the 100-websites Rimmer et al.'s closed-world dataset. In the open-world, with a foreground set of 100 sites and a background set of 10,000 sites, this attack achieved a multi-class TPR of 89.2% with 1.1% FPR. Due to the adaptation of two kinds of inputs to the model (the raw packet sequence and the precomputed features), this attack requires significantly larger training times for achieving slightly higher detection rates than DF.

Lastly, Sirinam et al. [73] focused on other DL methods to deal with changing content. The authors showed that it is possible to train on old datasets and still be able to achieve remarkable attack accuracy. Hence, this work demonstrated that the WFP attack has become more realistic and alarming for Tor users.

Website Fingerprinting Defenses. Along with the research of attacks, concurrent works have proposed several mechanisms to counter the recognition ability of a WFP adversary. Fundamentally, the idea of a defense is to modify the packet sequence in a way that the extracted features do not significantly contribute to the classifier's task. To achieve this, a basic idea is to intentionally insert *dummy traffic* on the transmission flow. Following this principle, Wright et al. [81] proposed *traffic morphing* as a mechanism to make that a website's packet sequence looks as a different one. This work outperformed previous packet-level padding countermeasures [82]. However, Dyer et al. [78] demonstrated that none of those padding techniques are effective against a WFP attack.

Panchenko et al. [70] proposed to introduce noise within the website's traffic flow by simultaneously loading an additional web page in parallel. Unfortunately, the required amount of introduced traffic noise is too high for an acceptable defense [72].

Another defense technique consists on generating a continuous data flow. Dyer et al. used this principle to present BuFLO [78]. This defense sends fixed-length packet units in bursts at a constant data rate. To achieve this, the defense delays and inserts a large amount of dummy traffic. As a proposal to reduce the bandwidth and latency overhead produced by BuFLO. Cai et al. presented two defenses, CsBuFLO [83] and Tamaraw [84], in which websites are associated in groups of similar size and padding packets are added to reach the maximum website size of each group. Likewise, Glove [85] groups web pages of similar size and adds cover traffic to make them indistinguishable within a cluster; and Supersequences [72], builds longer traffic sequences by padding similarly to Tamaraw. The a-priori knowledge required to protect a page load trace represents a complex usability problem that hinders the deployment of these defenses. Furthermore, bandwidth and latency overheads are still too high for considering the adoption of these defenses in Tor [20].

As a proposal to avoid the substantial changes that the development of a packet-level defense may require, other works have proposed countermeasures deployed at the application layer. As a response to the attack of Panchenko et al. [70], Tor developers experimented a technique called *randomized pipelining* [86] within the Tor Browser. This defense randomizes the order and number of requests produced by a website object. Similarly, HTTPOS [87] (HTTP Obfuscation) achieves the randomization by altering some parameters of the HTTP requests and responses (e.g., the accepted-range field header). Later on, Wang and Goldberg [22] showed that both application-layer defenses were not effective, and in some cases, leaded to higher recognition rates. Cherubin et al. [88] proposed client- and server-side defenses, LLaMA and ALPaCA. LLaMA reorders outgoing HTTP requests by randomly delaying them and adds dummy HTTP requests. ALPaCA applies morphing by padding the web objects of a page and inserting invisible dummy web objects.

The Tor research community has considered two defenses as candidates for adoption in Tor [89, 90]: adaptive padding [36] (WTF-PAD), and Walkie-Talkie [91] (WT). In WTF-PAD, the network point (client or router) inserts padding following a preloaded histogram that changes according to the current transmission state (burst or gap). This causes that the observed traffic for all websites follows the statistical properties of a single set of histograms. Padding according to the transmission state produces low bandwidth overhead and does not introduce latency on the traffic. Tor developers have shown a preference for this defense, and therefore have included in the recent Tor versions (from v.0.4.0.1) the framework for implementing histograms, protocols, and adaptive padding mechanisms. Lastly, WT enables the web browser to communicate in half-duplex mode. With this, each burst (in or out) is molded via padding in order to imitate the bursts of a decoy page. Despite the practicability and moderate overhead of WTF-PAD and WT,

Sirinam et al. [20] showed that none of them are enough effective against the DF attack. Thus, these defenses do not appear as suitable candidates for adoption in Tor.

As a defense for multihomed clients, Henri et al. [92] proposed strategies to split TCP traffic between the client and an additional Tor bridge over *two* independent network connections. The considered adversary is an ISP mounting WFP only on one of the two existing connections. Thus, this defense fails to provide any protection against malicious entry ORs (an adversary weaker, and thus easier to deploy, than an ISP).

There is, as of yet, no defense that effectively counters modern WFP attacks, while being easily deployable and producing unnoticeable overheads.

3.3.3.2 End-to-end Correlation Attacks

The objective of this attack is to assess the similarity of a pair of traffic flows observed at the two ends of a Tor circuit (e.g., at entry and exit ORs). To measure this similarity, several attacks considered traditional statistical correlation metrics such as the Pearson coefficient [38] or the mutual information [37]. This attack aims to predict whether a certain pair of traffic flows was originated or not (binary decision) on a single Tor connection. Thus, its performance can be assessed with the true positive and false positive rates.

There are two approaches that an attacker can employ to mount end-to-end correlation. First, by running malicious relays at entry and exit positions and logging the traffic activity on both sides. In this scenario, the attacker can leverage other routing attacks [24, 30, 67] to increase the chances to be selected by more clients. Second, the adversary can control an autonomous system (AS) or ISP to record traffic of the entry and exit ORs he has access to [40]. While this is more expensive for the adversary, it considerably increases the likelihood to control more ORs.

Among all existing attacks [31, 37–40], the latest proposed work, DeepCorr [31], has shown to be the most powerful attack. It achieved a TPR above 0.8 for a FPR below 2×10^{-3} correlating more than 50,000 traffic flows. This attack uses a CNN model to learn the level of correlation of a data flow. This DL model is highly resistant to changes on time and position of traffic patterns, which makes it robust to identify correlation on noisy realistic Tor traffic. For learning the model, DeepCorr represents each instance as a vector of inter-packet delays and packet sizes of ingress and egress traffic on both directions. Two classes are defined, the *positive* class determines that the instance shows high correlation (i.e., traffic belongs to the same Tor connection), and the *negative* class denotes low or no correlation (i.e., traffic belongs to entry and exit ORs from different Tor connections).

The ignorance the people live in leads them to commit mistakes against their own happiness.

Simón Bolívar

4

Multipath Routing in Tor

4.1 Introduction

Considering the complexity of its design, it is not trivial to directly determine which Tor aspects are subject to be redesigned with multipath. Therefore, we next introduce a comprehensive taxonomy in order to identify the possibilities that the multipath routing paradigm has for Tor. Then, we evaluate existing approaches that provide multipath in onion routing and, in addition, we enhance the design space with further alternatives. Particularly, we propose two novel design variations aiming to provide better performance to Tor clients. The specific contributions of this chapter are as follows:

1. We provide a systematic survey of currently-existing multipath approaches for Tor and other similar onion routing-based anonymization systems, as well as techniques that allow adding the multipath capability. To this end, we introduce a taxonomy for onion routing-based low-latency designs with a focus on multipath 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.

3. We introduce two novel design alternatives for onion routing anonymous communication systems. The former, mUDP-OR, is an incremental design to an existing UDP-based onion

routing system, which aims to boost — via multipathing — the advantages of removing the congestion control mechanisms from the transport layer. The latter, MPTCP-Tor, proposes to create multiple transport connections between ORs in order to provide a fairer bandwidth allocation to the inter-OR multiplexed traffic.

4.2 Considered Multipath Onion-Routing Approaches

Before proceeding with the analysis of multipath in the Tor design space, we first recapitulate the existing multipath onion-routing-based systems (see Chapter 3 for more details).

From the existing fully-developed multipath onion-routing-based approaches, two of them (Conflux and mTor) are extensions to vanilla Tor, and thus, operate with TCP traffic and using fixed circuits. On the other hand, MORE tunnels the traffic over UDP using a different circuit for each cell. To our knowledge, there is no fully-developed multipath 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 multipath approaches, we consider closing this gap in the design space to be necessary and so added multipath support to UDP-OR [52] as a 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 4.3).

We next summarize the main characteristics of the existing systems and describe our enhanced system, mUDP-OR.

Conflux: This extension to Tor allows a client to use two circuits with a common exit OR. Cells are dispatched based on the perceived congestion inferred from the time required by the `SENDME` cell to travel between the OP and exit OR. For reordering, a sequence number is appended to the cell's payload. In order to allow a further extensive evaluation, we implemented the generalization of Conflux to operate with m circuits.

mTor: This approach operates similarly as Conflux. The difference is the scheduling technique. Instead of using the `SENDME`-based RTT estimation as congestion metric, the stream-level window triggers the signal for switching between circuits.

MORE: In this system, each packet is sent through a different freshly-created circuit. To allow that packets are well received out-of-order — due to the diverse employed paths, MORE uses UDP for tunneling TCP data, and so, provide reliability and correctness between the end points.

mUDP-OR: For extending UDP-OR with multipath, we based the multipath structure and the circuit identification 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 and acknowledgment 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(s) 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.

4.3 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 4.1 illustrates our taxonomy. We focus on the multipath aspects and the effect of adding multipath capabilities. Based on the structure of our taxonomy, we classify and discuss the multipath OR approaches introduced in the previous section.

4.3.1 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 [47].

We classify traffic management into *OR-link layer*, *circuit layer*, and *multipath 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. While it would be plausible (albeit very inefficient) to fully address these aspects on each layer, not addressing them at all would not fulfill the requirements of a system for anonymizing reliable communication (e.g. a TCP connection). Inter-layer dependency gives rise to problems, the most prominent of which is cross-circuit interference. 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.

4.3.1.1 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

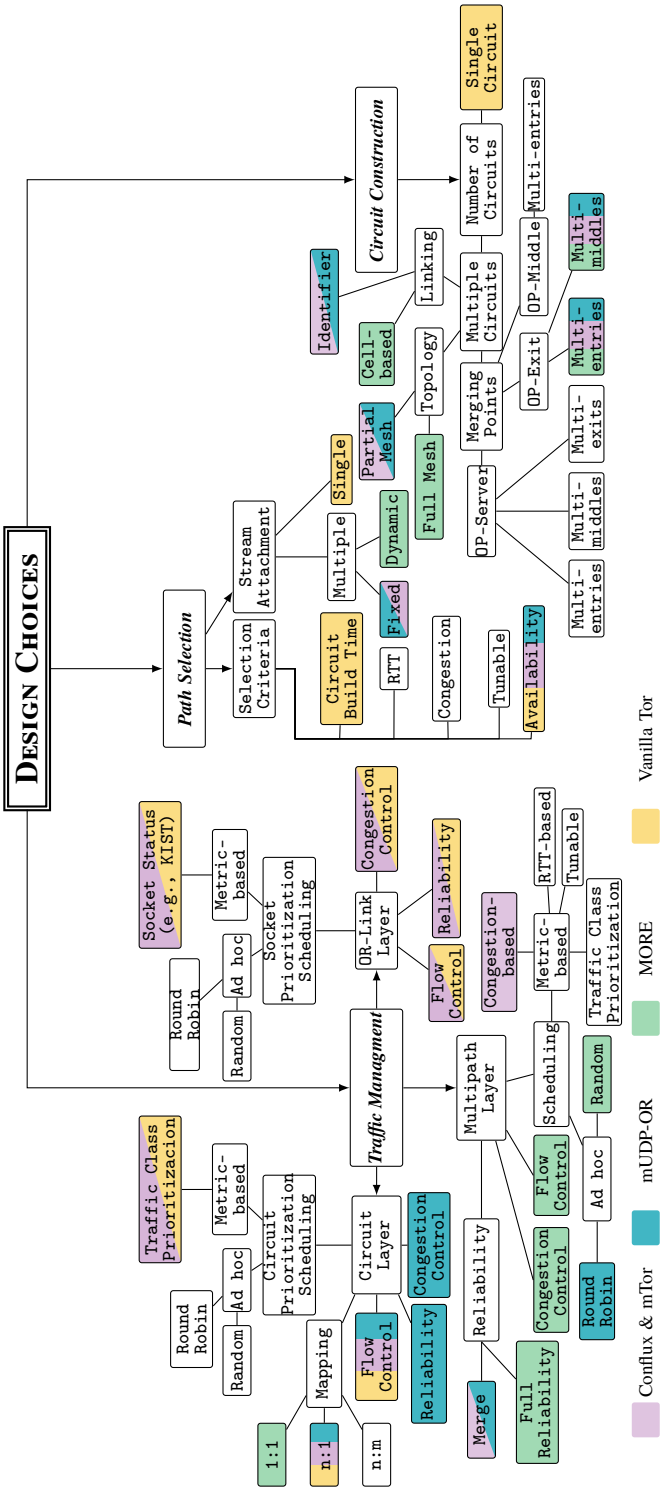


FIGURE 4.1: Taxonomy of design choices for onion routing-based approaches

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[1]; 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[2] 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 [54]; and secondly, using a simple transport protocol, e.g., UDP [52]. Conflux and mTor are Tor extensions that add the multipath 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[3], which might congest the network. This issue has been addressed in [57].

A multipath 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 multipath TCP [64] as a transport protocol. Since multipath 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 [93].

Lastly, we classify the OR-link by the employed *socket prioritization scheduling* methods. An OR typically keeps TCP connections with several other ORs. Thus, an OR-link level scheduler is required to decide the priority for serving the open sockets. We identify two subclasses of scheduling methods: (1) *ad hoc* (e.g., round robin or random), and (2) *metric-based*. In 2017, the KIST scheduler technique replaced the round robin scheduling in Tor. This method monitors the kernel-level state of pending sockets to decide which one is first served. In vanilla Tor, Conflux, and mTor, the KIST and round robin scheduler are available. MORE and mUDP-OR

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

do not implement any socket prioritization method, and each socket is served as it has available packets to send.

4.3.1.2 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 [23] 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 [57], 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 [18]. Prioritization of interactive connections on the circuit-level has been proposed by Tang et al. [48] 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 [47].

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 multipath TCP based solution described above is an example of (3).

We also classify the *circuit layer* by its *circuit prioritization scheduling* methods. This scheduler decides from which of the existing circuit-level output queues the next ready cell must be taken for its immediate allocation onto the corresponding OR-link transport connection. We classify *circuit prioritization* 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 [18], (2) provides superior overall performance.

Prior to 2012, Tor used *round robin* as circuit prioritization scheduler. Then, an improved scheduler (EWMA), based on the recent circuit's activity was implemented [48]. This scheduler works in conjunction with the socket prioritization scheduler. The recently introduced KIST scheduler improved the efficiency of EWMA by reducing the number of queued outbound packets. The integration of KIST in Tor improved the efficiency of EWMA by reducing the number of queued outbound packets. 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 prioritization*.

4.3.1.3 Multipath Layer

The *multipath layer* incorporates sets of circuits jointly building communication channels. We classify the *multipath layer* design by which of congestion control and flow control it considers. While it is a feasible design choice for the multipath layer to be agnostic to both flow control and congestion control, the realization of reliability for a multipath approach always includes the multipath layer. The subclasses of multipath 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 multipath layer allows for sending control information on less-congested circuits to reduce feedback time.

While Tor does not offer multipath capabilities, both Conflux and mTor can be seen as multipath 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 multipath layer. Since mUDP-OR is a multipath 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 *multipath layer*.

Another multipath layer design choice is *multipath scheduling*. While *circuit scheduling* decides from which circuit-level queue the next cell is put into the transport-level queue, *multipath 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 multipath 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, 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.

Multipath TCP [64] could be used not only on the OR-link layer, but also on the circuit and multipath layers, tunneling multipath TCP's sub-streams on the circuit layer and using its scheduling and merging strategy on the multipath 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 [94].

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[4] on the multipath 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., [57]) have not as yet been used in the context of multipath onion routing.

[4]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.

4.3.2 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 ORs) 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 4.5). 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 [95].

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 multipath layer, and choosing it properly results in faster multipath build times, and a more secure multipath structure.

4.3.3 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 round trip time (RTT), congestion, or a tunable combination of these may be also considered. The subclass *stream attachment* comprises special choices for multipath 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 *multipath 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.

Our taxonomy has explored and described the design points in which multipath for onion routing (particularly Tor) can be implemented. We observe several components of the design space in which it is worth to evaluate the anonymity and performance implications of integrating the multipath paradigm.

In the following of this chapter, we evaluate the design space considered by existing approaches. From this analysis, we sketch considerations for novel multipath-based approaches. These recommendations serve then as base for our novel proposed designs — the main contributions of this thesis — that improve privacy and performance for Tor users.

4.4 Performance Evaluation of Multipath Onion Routing-based Approaches

In this section, we present an extensive performance evaluation of the existing multipath onion-routing systems and our mUDP-OR enhancement. To this end, we have built a testbed where all networks can run under the same conditions. Setting up a realistic environment for evaluating anonymization systems, such as Tor, is challenging by itself and, thereby, has been deeply investigated [96, 97]. For our evaluation, this task becomes even more complex as the studied systems utilize different transport protocols, various client interfaces, and were written in different programming languages. For instance, the preferred Tor evaluation tool — the Shadow simulator [97] — does not support other transport protocols than TCP. Hence, two of the studied approaches cannot be simulated with this tool. On the other hand, the ns-3 simulator[5] requires that applications are rewritten as modules part of this simulator. Thus, we prefer instead to run the applications in real machines, and when needed, we emulate network effects such as latency and packet drop rate.

4.4.1 Experimental Setup

We conduct our experiments in two settings: a private local network and an emulated larger-scale scenario. The former setting allows us to verify the functionalities of each system and assess their performance upper bounds. With the latter scenario, we aim to show the effects expected under realistic conditions.

[5] `https://www.nsnam.org/`

When performing experiments in the local network, unless otherwise specified, we do not introduce any artificial network effect. We implemented a set of scripts to coordinate and log the events and results of our experiments.

Emulating a larger-scale network is more challenging. Besides running several hundreds of parallel applications (e.g, Tor, web servers, and web clients), the testbed must introduce realistic network effects between each emulated network node. To achieve this, we leverage an integration of two frameworks: the NetMirage[6] emulator and the Chutney[7] testing platform. NetMirage is a tool to emulate any IP-based application that can be bound to a desired IP address. This emulator uses the Linux network namespaces to create several virtual network spaces to which multiple instances of an application are attached to. To simulate network effects, NetMirage uses the `netem` functionality to set latency, bandwidth, and packet drop rate values taken from a pre-loaded realistic topology. This process is carried out in two physical hosts, the *core* machine runs all the applications, and the *edge* simulates the network effects. To avoid external influences, both hosts are connected on an isolated network with uncapped bandwidth. On the other hand, the Chutney tool is used to launch several Tor instances (both as OPs and ORs) in the core host and to setup the Tor configuration options accordingly.

We report three metrics in our experiments. The time to first byte (TTFB), which is defined as the time between the content is requested to the server until the first data byte arrives. The download time (DT), defined as the time to get the full content from the server. And the percentage of CPU utilization on the anonymization nodes (i.e., the ORs).

4.4.2 Private Local Network Experiment

We use this experiment to understand the differences between all designs without external influence. In our private 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. We repeated the measurements 200 times to present them with with 95% confidence 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 multipath scheduler. This also ensures that multiple circuits will be equitably used. Effects of congestion-based schedulers are evaluated in the second scenario.

4.4.2.1 Multipath Circuits and Load Balancing

In Table 4.1 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.

[6] https://crysp.uwaterloo.ca/software/netmirage/
[7] https://gitweb.torproject.org/chutney.git

TABLE 4.1: CPU usage percentage on the onion routers for the maximum number of clients.

		CPU Usage on the OR		
Approach	Paths	Entry	Middle	Exit
Conflux	1	57.87 ± 2.16	49.85 ± 1.82	31.74 ± 1.15
	2	33.22 ± 2.25	31.01 ± 2.21	31.51 ± 2.22
	3	27.24 ± 1.94	22.78 ± 1.56	32.94 ± 2.28
mTor	1	57.87 ± 2.16	49.85 ± 1.82	31.74 ± 1.15
	2	25.10 ± 2.21	23.19 ± 1.73	32.51 ± 2.26
	3	13.45 ± 0.97	14.62 ± 0.85	32.90 ± 2.27
mUDP-OR	1	88.31 ± 1.51	86.89 ± 1.48	71.33 ± 1.24
	2	36.75 ± 2.51	32.67 ± 2.79	73.48 ± 3.09
	3	22.73 ± 3.11	24.45 ± 3.27	75.30 ± 3.21
MORE	1	6.73 ± 0.14	6.93 ± 0.19	75.92 ± 2.34
	2	6.65 ± 0.13	6.93 ± 0.20	70.09 ± 2.24
	3	6.57 ± 0.11	6.52 ± 0.08	68.19 ± 2.48

It is clearly observable that the load assigned to each OR is decreased by using multiple circuits simultaneously. An important effect to notice is the different level of load created on the ORs by each approach. While in the systems which fixed circuits — Conflux, mTOR, and mUDP-OR — exit ORs are more loaded than entry ORs, the dynamism of MORE significantly alleviates entry ORs and translate the heavy computation effort to the exit OR. Since our clients mostly retrieve data (outgoing packets are minority), the backwards flow is expected to be more computational intensive for entry ORs because they are handling data with three encryption layers.

Regarding load balancing among entry ORs, our mUDP-OR enhancement drastically reduces the amount of CPU consumption on entry ORs. This decrease is less noticeable when a third entry OR is added.

In general, we observe that middle ORs remarkable capitalize the benefits of using multiple paths. Furthermore, we identify that despite mTor and Conflux are similar designs, the implementation and scheduling techniques of mTor require less capacity for the ORs, which, in turn highlights the importance of developing efficiently the multipath design for a future real-world adoption.

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.

4.4.2.2 Client Performance

Performance metrics are presented in Table 4.2. 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). Since the local network

TABLE 4.2: Performance metrics for the maximum number of clients.

		Client Performance Metrics		
Approach	Paths	TTFB [s]	DT Web [s]	DT Bulk [s]
Conflux	1	0.027 ± 0.005	0.059 ± 0.003	0.113 ± 0.008
	2	0.016 ± 0.002	0.052 ± 0.0005	0.082 ± 0.002
	3	0.014 ± 0.003	0.049 ± 0.0007	0.079 ± 0.0024
mTor	1	0.027 ± 0.005	0.059 ± 0.003	0.113 ± 0.008
	2	0.012 ± 0.002	0.055 ± 0.001	0.079 ± 0.003
	3	0.017 ± 0.003	0.049 ± 0.0007	0.081 ± 0.002
mUDP-OR	1	0.002 ± 0.0003	0.031 ± 0.0018	0.076 ± 0.005
	2	0.002 ± 0.0001	0.034 ± 0.0017	0.081 ± 0.001
	3	0.0024 ± 0.0001	0.035 ± 0.001	0.113 ± 0.014
MORE	1	0.014 ± 0.004	1.41 ± 0.007	N.A
	2	0.014 ± 0.004	1.37 ± 0.19	N.A
	3	0.016 ± 0.006	1.39 ± 0.014	N.A

provides almost ideal links between all nodes, effects of packet loss and latency are not present, therefore the mUDP-OR approach performs better than the TCP-based approaches. However, more paths do not contribute to better performance in UDP-OR because in such case reordering must be handled end-to-end with a long feedback that reacts slower than the hop-level TCP of mTor and Conflux. In contrast to Conflux and mTor, our multipath enhancement to UDP-OR did not produce the desired improvements due to the still-existing very long feedback for retransmissions and acknowledgments.

4.4.3 Larger-Scale Experiment

Currently, the Shadow [97] 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. This emulator uses the virtual network spaces facility from Linux to create a large virtual environment, where any application(s) can be bound to each virtual network space. In order to simulate the effects of a real network, NetMirage uses a topology file (GraphML) to reproduce effects such as packet loss, latency, bandwidth between the links in the virtual network. Within this emulator we built a test bed to perform a fair performance comparison among all considered approaches. Since Tor is the only one network deployed at large scale, we tried to reproduce a scenario enough large to recreate realistic conditions for all approaches.

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.

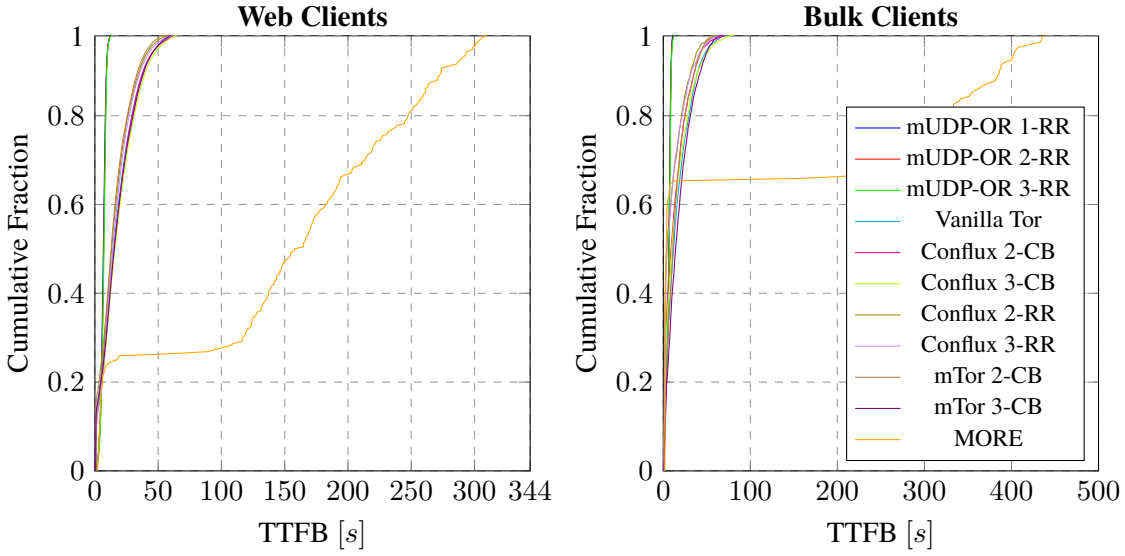


FIGURE 4.2: Time to first byte for web and bulk clients

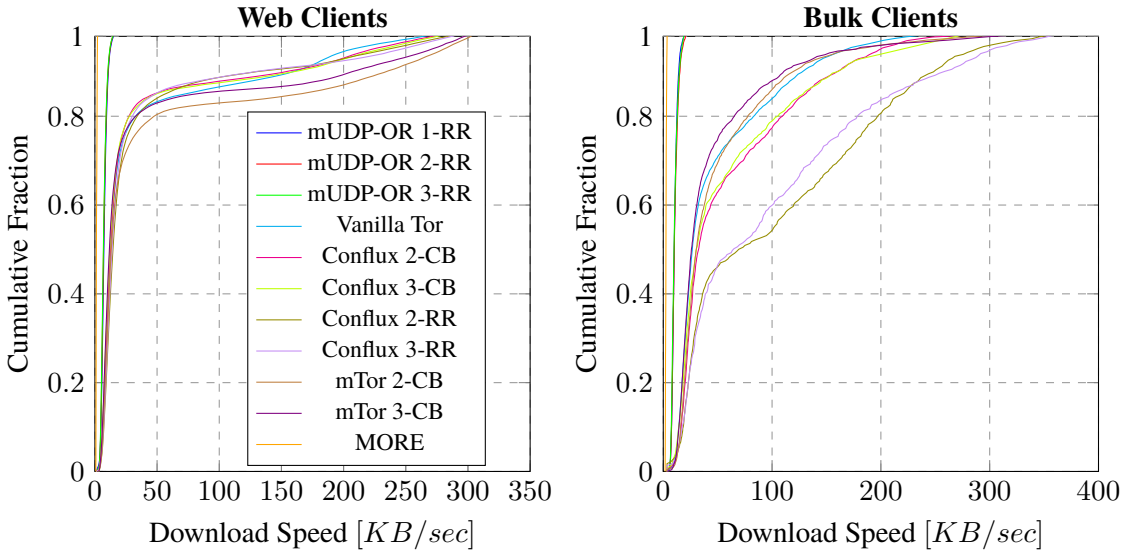


FIGURE 4.3: Download speed for web and bulk clients

4.4.3.1 Client Performance

For every approach, two design variations were evaluated: the number of paths (labeled as 1,2,3) and the *multipath* scheduler (labeled as RR for round robin and CB for congestion based). We run our experiments for two hours (see Figure 4.2 and Figure 4.3). We observe that, in a congested environment, mUDP-OR only outperforms other approaches in the TTFB metric. This is a clear advantage of using UDP as transport protocol, because data can be received before any acknowledged is confirmed.

On the other hand, the congestion-based scheduling techniques of mTor and Conflux do not profit completely from the utilization of multiple circuits; this may explain why the round robin scheduler performs better, particularly for bulk downloads. Thus, it is necessary to develop

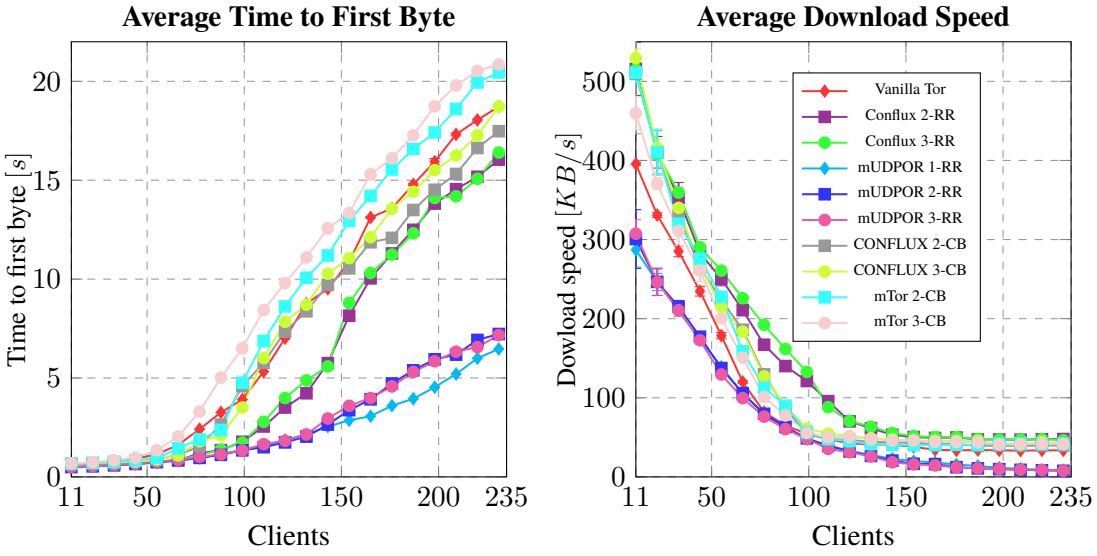


FIGURE 4.4: Performance metrics for different number of clients

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.

4.4.3.2 Network Scalability

In this experiment, we incrementally introduced up to 228 clients (always keeping the proportion 10% for bulk and 90% for web clients) and measured TTFB and download speed for each iteration. In Figure 4.4, we show the variation of TTFB and download speed while more clients participate in the network. For both metrics, we observe, as expected, that for all systems performance is degraded as more clients participate on the network. For TTFB, we notice that mUDP-OR outperforms other systems in terms of scalability, while mTor and Conflux download data faster when more clients are present.

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

4.5 Anonymity Implications of Multipath Routing

In this section, we address the anonymity implications produced by using multiple paths in the context of the evaluated approaches. We aim to shed light on the privacy risks that emerge from using multiple paths in order to avoid unfavorable design choices for future proposals.

4.5.1 Client Multipath 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 $P(c) \approx f_g f_x$ [10, 98]. For multipath 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. For instance, 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 $m = 3$. Hence, we consider that clients using even five or six entries do not significantly loose their anonymity protection against adversaries with realistic bandwidth capabilities.

4.5.2 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 so called priority-ordered *primary* list. 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.

Since a multipath 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 [26] together with metrics and adversary models presented in [99]. 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 [100]. The second, defines a compromised client if and only if all m entries are controlled, which may be valid for *website fingerprinting* attacks [19, 101–103]. We aim to cover with these two models the security bounds of compromise rate for multipathed clients. 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 4.5 shows the mean compromise rate

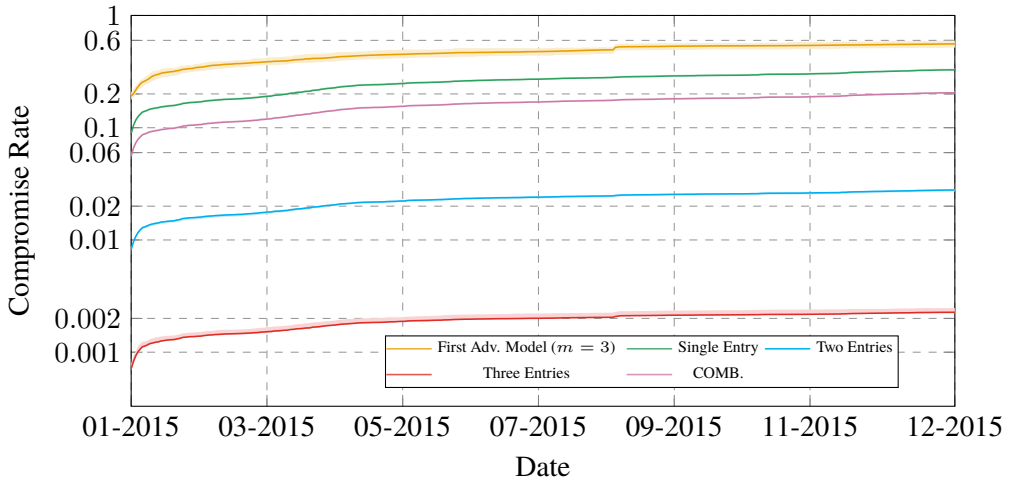


FIGURE 4.5: Fraction of compromised clients (Compromise Rate) during one year: Single, two, and three entries refer to the second adversary model. In the scenario labeled as COMB, 60% of the clients use a single entry, 20% two entries and 20% three entries.

(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. 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. In this attack, 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:

$$\overline{A} = \frac{2 \times 10^6}{\binom{2000}{m}} \tag{4.1}$$

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).

4.5.3 Selective Denial of Service Attack and Circuit Reliability

Another threat that affects to Tor clients utilizing multiple paths is the selective denial of service (SDoS) [67]. Contrary to a general denial of service attack, the SDoS is particularly dangerous to systems such as Tor because it is cheaper, less detectable, and more destructive since denying service on specific ORs would not drive users away, but making their communication less reliable. The reliability of a Tor circuit is defined as the probability that *any* of its ORs does not fail. In other words, if an OR has a probability of being reliable of f, the reliability of the circuit is

$R = f^3$ (entry, middle and exit OR are reliable). Non-reliable circuits lead to users reattempting to restart the communication and increasing the chances to get compromised (i.e., an adversary aiming to control both edges of a circuit).

In Tor, the SDoS operates as follows: an adversary controlling a certain number of ORs on the entry or exit position monitors for a correlation between traffic at both points to compromise a client. In case such match does not occur, or the adversary is at the middle position, the attacker refuses to forward traffic. This kills the circuit and forces the client to build a new one. Ultimately, the attacker gets a new chance to correlate the traffic of the client. Tearing down a circuit by refusing forwarding traffic makes it less reliable. Therefore, under SDoS, the reliability of a Tor circuit depends on the reliability of the ORs and the fraction of honest nodes (t) in the network[8]. Borisov et al. [67] defines the reliability of a single Tor circuit under a SDoS attack as:

$$R_{DOS} = (1-t)^2 + (tf)^3 \tag{4.2}$$

In other words, the circuit is reliable if all nodes are honest (the component $(tf)^3$) or both entry and exit OR are compromised (the component $(1-t)^2$). For multipathed clients, the reliability on this attack can however change as an entire multipath circuit is not necessarily destroyed if one OR denies service. On one hand, $(1-t)^2$ changes as described in Section 4.5.1 because the compromise likelihood on the entry side is slightly higher. On the other hand, $(tf)^3$ changes for two cases: *(i)* if one failing OR destroys the entire multipath circuit, and *(ii)* if a failing OR destroys only the circuit it is part of, but not the entire multipath tunnel. In general, one would prefer the option *(i)*. However, if the usage of multiple circuits is the means for anonymity boost (e.g., see Chapter 5), it is preferable to close the tunnel rather than allowing the reduction to a single-circuit communication. Hence, the generalization of R_{DOS} for a multipath client using m circuits for the both cases is[9]:

$$R_{DOS_{m(i)}} = (1-t)(1-t^m) + (tf)^{2m+1} \tag{4.3}$$

$$R_{DOS_{m(ii)}} = R_{DOS_{m(i)}} + (m-1)[(tf)^m(1-tf)^{2m-2} + m\sum_{k=1}^{m}(tf)^k(1-tf)^{m-k+1}] \tag{4.4}$$

Where t is the fraction of honest nodes, f is the probability of a OR being reliable, and m the number of entry ORs. Conflux and mTor operate under the option *(i)*, and mUDP-OR and MORE with the option *(ii)*.

Figure 4.6 shows the graphical representation of the reliability formulas for single- and multipath-circuit scenarios for $m = 3$ and $f = 0.99$. We observe that multipath systems are more reliable when the minority of nodes are honest. It means that adversaries can compromise easier mulltpathed clients. Nonetheless, this difference is not drastic for realistic adversarial

[8]This is an approximation for an adversary controlling a fraction $(1-t)$ on both entry and exit sides. See Section 4.5.1 when bandwidth fractions on entry and exit ORs are different.

[9]Formulas are based on the case: multiple entries, multiple middles, and merging at the OP and exit OR.

scenarios. It is important to notice that in the multipath operation, the system can also decide to rebuild the lost circuit even if the communication is still alive. For the attacker, this ends up with a similar effect as in the single-circuit scenario, because he has a new chance to participate in the client's circuit. A good compromise would be to define a minimum number of required circuits to operate, and, in case of lost circuits, not to create new ones while the multipath tunnel is still alive.

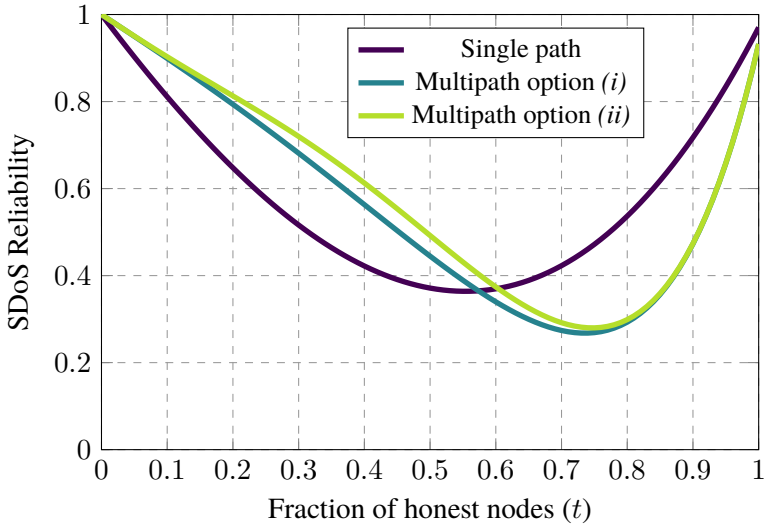


FIGURE 4.6: Reliability under the SDoS attack for single- and multipath-circuit scenarios in Tor

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 multipath capabilities. The main constraint is the fact that using multiple entries is not considered in the latest Tor specification, however future research directions [26] aim to give more flexibility in this regard.

4.6 Design Recommendations

From the performed evaluation, we identify relevant design aspects where multipath may provide improvements for performance and anonymity. Regarding performance, we identify that any design (single or multipath) based on UDP, provides a fast first response. This feature comes however, at the cost of a slower total download time. 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 multipath scheduler. Here, we notice that the multipath-level scheduler must be optimized to use the path with the best characteristics. This, in turn, has to be implemented without creating novel surfaces for further anonymity attacks.

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. We also notice that an unexplored design option comprises the implementation of multiple transport connections between ORs to replace the single shared one. With this, a fairer bandwidth allocation for Tor traffic could be achieved.

With respect to anonymity, a client using multiple circuits — particularly multiple entry ORs — could withstand traffic-analysis attacks such as website fingerprinting, while the information available to a certain malicious entry OR does not disclose enough meaningful patterns to mount the attack. Thus, we identify the usage of multiple entry ORs as an opportunity to design a protection mechanism against website fingerprinting attacks.

4.7 MPTCP-Tor — A Novel Tor Transport Design for Fairer Bandwidth Allocation

One remarkable outcome from our analyzed taxonomy is that the design of Tor at transport level is suboptimal. We noticed that using a single TCP connection for traffic that carries packets from multiple users leads to unfairness and performance degradation. In response to this, and following the main motivation of this thesis, we next investigate whether the multipath paradigm can be applied to this design aspect in order to improve the performance of Tor users.

4.7.1 Problem Statement and Proposed Solution

In Tor, between any pair of ORs, cells of different circuits (typically originating on different clients) travel multiplexed in a single TCP connection. This protects users' privacy because it prevents adversaries from mounting traffic confirmation attacks over circuit-specific traffic, and minimizes information leakage about circuits' activity. However, this design choice causes an unfair allocation of bandwidth for Tor traffic, as TCP congestion control gives all the multiplexed Tor traffic as little of the available bandwidth as it gives to every single TCP connection that is competing for the same resource. Additionally, congestion events affect *all* circuits, even if those events were caused by a single circuit. In response to this, several transport alternatives have been proposed [18]. Although some of these designs have shown noticeable performance benefits, none of them can practically be deployed in the real Tor network because they are backwards incompatible, and require structural changes to the entire system.

In response to this, we propose creating multiple transport connections between ORs using an out-of-the-box solution — multipath TCP (MPTCP) [64]. This aims to better utilize the available resources and to increase the bandwidth intended for Tor traffic to a fairer extent.

As of May 2020, from the 6,469 relays listed within the Tor network[10], 5,947 run Linux, 464 run BSD-based systems, 8 run Unix for MacOS, 46 run Windows, and 4 run Solaris. Hence,

[10] All relays that are or were stable and operative were registered

beyond the advantages of using an out-of-the-box and backwards-compatible protocol, we consider the study of MPTCP applicability in Tor as highly relevant, because its integration as default feature in Linux-based systems (used by most of the ORs) is foreseeable in the near feature [104].

4.7.2 MPTCP as a Tor Transport Protocol

Proposing changes to the Tor transport design is a challenging task [46]. There are approaches that offer attractive improvements and address the root problem of the Tor transport design [57]. However, such upgrades require structural changes on the Tor network — an upgrade for practically the entire system. Thus, and despite the promised benefits, the Tor community still believes that those alternatives require more research to determine optimal benefits that merit such an impacting change on Tor. Alternatively, we propose a design that offers acceptable benefits while being compatible and easy to deploy in Tor. We believe that such light upgrades on the system can provide significant improvements without noticeable implementation and deployment overhead. We next define the design goals (DG) for MPTCP as a Tor transport protocol

- **DG-1:** Provide fairness to the bandwidth allocated to Tor traffic. Consequently, Tor clients should experience faster downloads.
- **DG-2:** For easing deployability, MPTCP must be used out-of-the-box — no further deployment over the protocol must be required.
- **DG-3:** The integration in the real Tor world must be incremental and compatibility to legacy TCP traffic must be supported by ORs.

Next, we survey MPTCP, analyze its integration in Tor, and describe our implementation.

4.7.2.1 MPTCP Overview

MPTCP enables two hosts to communicate using multiple TCP connections (called *subflows*) simultaneously. To this end, MPTCP provides mechanisms to open extra connections, distribute traffic along different paths, and associate packets from different routes to a unique MPTCP connection. We next describe the relevant design aspects.

Connection Initialization. A MPTCP connection is initiated by a three-way handshake that includes the flag `MP_CAPABLE` in the `SYN` packet. If both hosts support this flag, handshakes for new subflows can be exchanged, otherwise the connection is treated as regular TCP. With the parameter *path-manager*, MPTCP offers three options for the creation of subflows. In *(i) fullmesh*, each host announces all available network interfaces, and all possible subflows in full-mesh topology are created. The *(ii) ndiffports* option allows that between a single pair of IP addresses, multiple ports are used to open a configurable number of subflows. The option *(iii) binder* initiates subflows among all available gateways on the hosts. MPTCP limits the maximum number of simultaneous open subflows to 32.

Association of Subflows. While establishing the first subflow, both hosts exchange cryptographic material to calculate a common token. Then, the handshake of each extra subflow that includes a matching token is associated with the same MPTCP connection.

Data Transfer. The *multipath scheduler* has three options to decide the route for each outgoing packet. In the *(i) default* mode, the path with the lowest RTT is selected while the congestion window is not full. Then, packets are routed to the subflow with the next higher RTT. The *(ii) round robin* scheduler sends a configurable number of packets on each subflow sequentially. Lastly, the *(iii) redundant* mode sends replicated traffic on all subflows. In order to reorder packets coming from different subflows, an additional sequence number relative to the subflow is introduced in the packet's header.

Congestion Control. A design goal of MPTCP is that it should not be more aggressive than TCP. To achieve this, congestion should not be regulated for each subflow independently or uncoupled. Therefore, the *lia* algorithm — the default MPTCP coupled congestion control algorithm — keeps track of all per-subflow congestion windows, and regulates them with a common factor to avoid all subflows becoming more aggressive than a TCP connection. On the other hand, MPTCP also allows the use of legacy congestion control as in TCP (e.g., cubic). In this case, congestion is treated uncoupled per subflow, which increases the throughput of the MPTCP connection proportionally to the number of subflows.

4.7.3 Integrating MPTCP in Tor

In order to fulfill our design goals, we next analyze how each parameter and configuration must be set up when using MPTCP as a Tor transport protocol. We identify two options for integrating MPTCP into Tor: *(i) externally* or *(ii) internally*. *(i)* is the de facto setting for most applications. Here, all MPTCP options are configured for all applications in the OR host and Tor's source code remains unmodified. In *(ii)*, MPTCP is configured from Tor's source code, offering flexibility, adaptability, and interaction with Tor events. For both options, ORs maintain the capability to serve legacy TCP connections. Next, we suggest settings for the integration of each MPTCP option in Tor.

Enabling MPTCP. If MPTCP is enabled externally, it becomes available to any TCP service operating concurrently with Tor. For a more flexible operation (i.e., the OR operator can decide when the relay uses MPTCP) we suggest incorporating it internally into Tor.

Path Manager. Since ORs are typically not multihomed, the *ndiffports* setting allows for a wider adoption of MPTCP in Tor. Moreover, it has been shown that *ndiffports* performs better in symmetric scenarios [105] (e.g., when a single interface is used for MPTCP). However, *fullmesh* could still be used by multihomed ORs which, besides offering increased bandwidth, provides communication resilience due to the diversity of the employed routes (e.g., 4G and Wi-Fi). For supporting MPTCP to any OR regardless of the available network interfaces, we suggest setting this parameter internally in Tor.

Number of Subflows. To achieve a fair bandwidth allocation, n circuits in a MPTCP connection should optimally have $m = n$ open subflows at their disposal. Otherwise, either excessive ($m > n$), or deficient ($m < n$) bandwidth would be unfairly allocated to Tor traffic. Thus, it is required that Tor and MPTCP synchronize the number of employed subflows. To this end, each pair of ORs must coordinate to demand MPTCP to maintain open a certain number of subflows. In a Tor transport connection, both ORs keep track of the number of *active circuits*. Hence, the ORs can communicate this parameter to MPTCP (integrating it internally into Tor) to open/close subflows on demand, and so, provide fair bandwidth resources to the existing circuits. It is worth mentioning that although $m = n$, this does not imply that a subflow carries traffic of exclusively one circuit, because the multipath scheduler still selects independently the subflow for each packet.

Multipath Scheduler. Performance improvements are expected when TCP packets are transmitted through the route with the best characteristics. Thus, and as recommended in the MPTCP specification [64], we consider the *default* scheduler for integration into Tor.

Congestion Control. Using an uncoupled congestion control algorithm causes overly aggressive traffic, which should be avoided. Typically, a MPTCP connection represents a single logical connection of two communicating applications. However, for interconnecting ORs, this multipath TCP connection carries a variable number of logical connections — the Tor circuits. Because each Tor circuit carries the packets corresponding to a communication that without Tor would be transmitted over a separate TCP connection, having a separate congestion window should be considered for better fairness. For this reason, we suggest using an uncoupled (e.g., cubic) congestion algorithm for interconnecting ORs. Since ORs would keep as many open subflows as active circuits, MPTCP will not cause more aggressiveness than what could be considered fair.

To sum up, we have addressed the most relevant design aspects of incorporating MPTCP in Tor. We noted that integrating MPTCP parameters internally in Tor offers fair performance benefits, high flexibility, and adaptability.

4.7.4 Implementing MPTCP as a Tor Transport Design

An important advantage of MPTCP over other Tor transport designs is that it can be used *out-of-the-box* (i.e., it is ready to be used without additional deployment). Only Tor may require modifications for the case that MPTCP is integrated internally (e.g., to open or close subflows from Tor). We next describe the implementation details for each integration option. We use the most-deployed MPTCP Linux implementation [106] and its available API [107].

External Integration. Here, MPTCP operates transparently for *all* applications running on the hosting machine. Thus, this does not require any modification in Tor but only to configure the host as specified by the Linux kernel MPTCP implementation. This comprises setting the

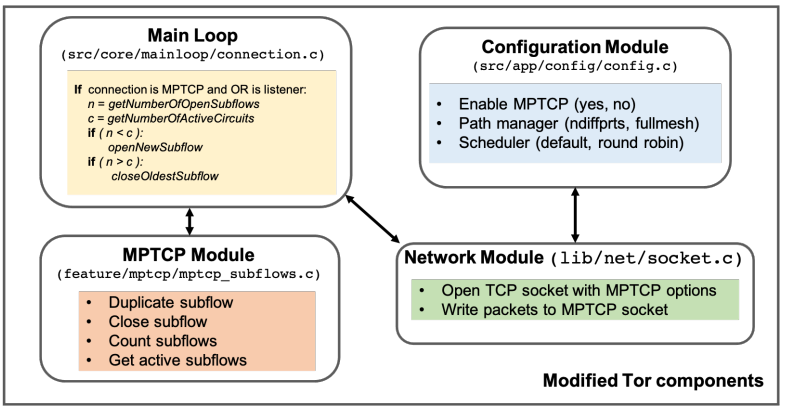


FIGURE 4.7: Components modified to implement MPTCP as a Tor transport protocol

`sysctl` calls to configure the MPTCP options, the congestion control algorithm, and the fixed number of subflows.

Internal Integration. Our previous analysis determined that this option provides fair performance benefits to Tor traffic. Besides setting the `sysctl` calls to allow that MPTCP can be used by any application, this option requires to implement the control of subflows from Tor. To this end, we leverage the latest available MPTCP API. This includes socket-level functions that can be called from Tor to: *(i)* enable and configure MPTCP, and *(ii)* open or close subflows. For *(i)*, we reimplemented the TCP socket functions of the Tor network module and enabled all MPTCP parameters (i.e, path manager and scheduler) as Tor configuration options. For *(ii)*, we implemented a new feature module that handles the calls to the API functions. This new module monitors the number of *active circuits* that the OR maintains with each other OR it is connected to. For each MPTCP connection, when the number of active circuits does not match the number of subflows, the module calls the corresponding API function to open or close subflows. In order to avoid that two ORs in a certain MPTCP connection perform a redundant regulation of number subflows, the calls to the API functions are made only by the OR acting as listener in the connection.

Figure 4.7 shows in detail the modules that we modified and created to integrate MPTCP in Tor. In the main loop of an OR, we constantly monitor the number of active circuits that the listener OR maintains. Then, with help of our MPTCP module, we call to the necessary functions to fulfill our goal of keeping the same number of active circuits as subflows in a OR-to-OR connection. It is important to emphasize, that we did not modify MPTCP and that the changes implemented in the Tor source code — for the internal integration — are minimal (less than 500 lines of code).

4.7.5 Performance Evaluation of MPTCP-Tor

We now evaluate the performance gains for Tor users when MPTCP is used as the Tor transport protocol. First, we set up a local Tor network to validate the functionality of MPTCP and assess

TABLE 4.3: Evaluated scenarios in the private local network for MPTCP-Tor.

	Web clients	Bulk clients	Latency	Packet drop rate
Scenario I	1	1	0 ms	0%
Scenario II	3	1	40 ms	0.5%
Scenario III	9	3	40 ms	0.5%

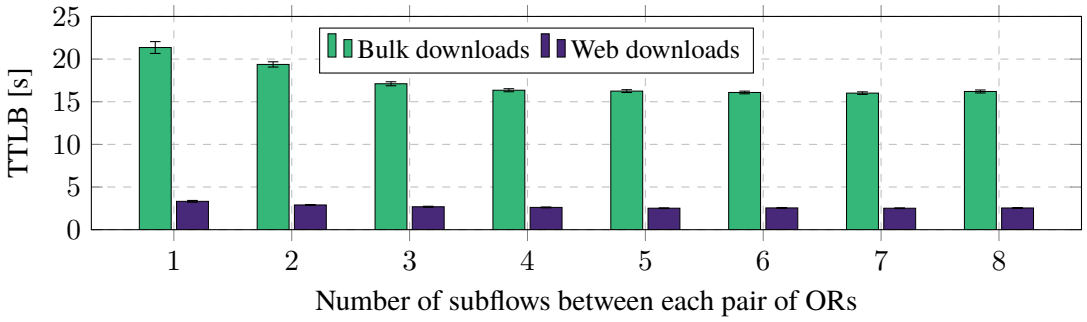


FIGURE 4.8: Time to last byte for different number of subflows between ORs

the upper bounds of the performance benefits. Second, we conduct a larger-scale evaluation considering real-world Tor settings.

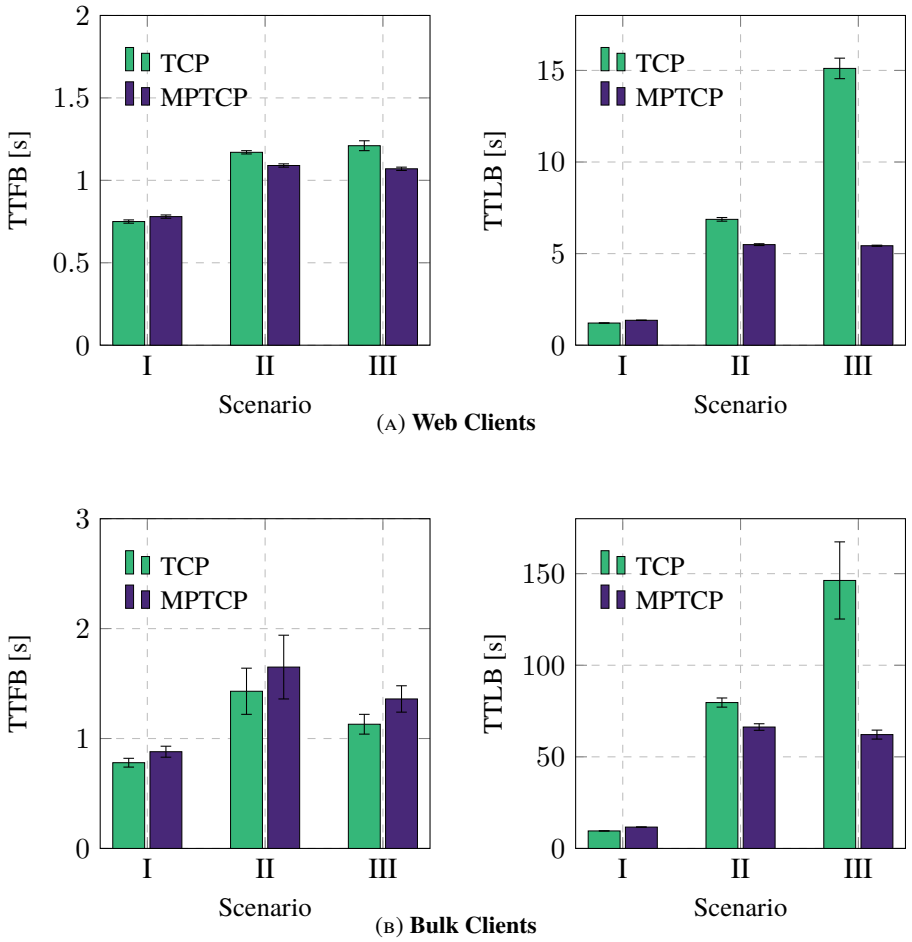
4.7.5.1 Experimental Setup

Similarly as for our experiments with multipath routing onion-routing based approaches, we leverage the NetMirage emulator to build both the small and large-scale scenarios. We measured two client-side metrics in our experiments: time to first byte (TTFB), and time to last byte (TTLB). We modified Tor v0.4.3.4 to use all options of the most-deployed MPTCP Linux implementation, v0.94. Specifically, we enabled all MPTCP parameters as Tor configuration options, and implemented the capability for opening and closing subflows according to the active circuits using the MPTCP API. Based on our previous analysis (cf. Section 4.7.2), ORs were configured to operate the *ndiffports* path manager, the *default* scheduler, and the *cubic* congestion control algorithm.

4.7.5.2 Private Local Network Experiment.

The objective of this experiment is to benchmark the differences between MPTCP and TCP as transport protocols in Tor. Moreover, we seek to verify the effectiveness of the design choices described in Section 4.7.2.

First, in order to fulfill the goal **DG-1**, we need to verify whether having the same number of subflows as coexisting circuits indeed provides performance improvements. To this end, we conduct a preliminary experiment, where we repeated 100 runs of six web and two bulk clients downloading data over circuits with the same ORs and using different number of subflows on each OR-to-OR link. From Figure 4.8, we confirm that indeed having as many subflows as



(A) **Web Clients**

(B) **Bulk Clients**

FIGURE 4.9: Performance evaluation of MPTCP and TCP as transport protocol between ORs in the local Tor network. .

circuits enlarges the bandwidth to an extend that TTLB gets noticeably faster. When there are less subflows than circuits, not all resources that would be considered fair are used.

Once confirmed our critical design choice, we next compare the performance of TCP and MPTCP as a Tor transport protocol. We conduct an experiment in a local Tor network composed by six ORs and 12 clients concurrently downloading web (320KiB) and bulk (5MiB) data from four HTTP servers. Specifically, we evaluated three scenarios with different numbers of active clients (cf. Figure 4.9) using the same ORs as entry, middle, and exit during 1,000 seconds[11]. The Tor network runs over a star topology with 100 Mbit/sec bandwidth capacity per link, and with similar packet drop rates and latency values as in previous evaluations [47, 54, 56]. In Table 4.3 we detail the characteristics of each evaluated scenario.

In Figure 4.9, we notice that for Scenario I (the simplest comparison between MPTCP and TCP) TCP slightly outperforms MPTCP because a single connection does not get fully utilized during the data transmission, and MPTCP requires extra subflow handling, which introduces a

[11] all results are presented with 95% confidence intervals

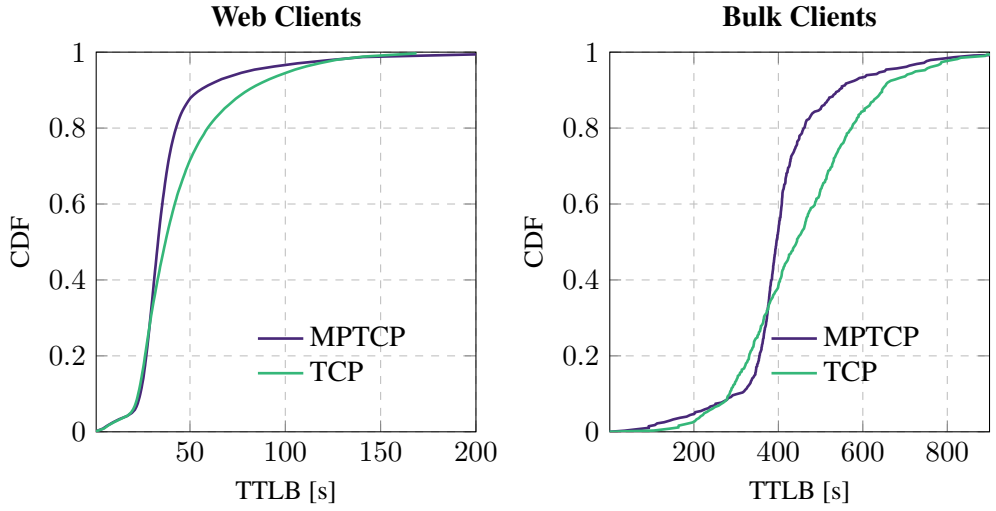


FIGURE 4.10: Cumulative distribution function (CDF) of time to last byte for web and bulk clients for scenarios when MPTCP and TCP are used as transport protocol between ORs.

small overhead. Once more clients are active, and packet drop and latency are induced (Scenarios II and III), we noticed that MPTCP takes advantage of the extra bandwidth provided by the multiple subflows working together with the uncoupled congestion control algorithm. In a more congested environment (Scenario III), we observe that for TCP the single existing connection gets saturated and all clients unfairly suffer performance degradation. On the other hand, MPTCP allows clients to possess the resources of 12 subflows, which, in turn, results in favorable improvements for both web and bulk clients. Regarding TTFB, TCP and MPTCP behave similarly. However, for the specific cases of no congestion, or large downloads, MPTCP introduces a small overhead due to the subflow management operations.

4.7.5.3 Larger-Scale Experiment

We next evaluate MPTCP as the Tor transport protocol in a larger-scale scenario. In realistic conditions, several links between nodes may not have enough bandwidth to satisfy all Tor traffic. This experiment allows us to understand whether the cases where bandwidth resources are plentiful represent a noticeable improved performance in the whole network.

In this experiment, we used the PlanetLab topology to build a network with nodes located around the world with realistic values for bandwidth, latency, and packet drop rate. To keep the same proportions as the real Tor network, we set up 44 relays, 11 web servers, 30 Torperf clients, 359 web clients, and 39 bulk clients [108]. Geographical locations and bandwidth characteristics of the ORs were set up using data obtained from the real Tor network statistics from 04/2020. Web clients retrieved 320 KiB with random pauses of between 1 and 20 seconds between downloads, bulk clients downloaded 5 MiB without pausing, and Torperf clients sporadically downloaded 50 KiB, 1 MiB, or 5 MiB. We ran this experiment for two hours for scenarios when ORs use either MPTCP or TCP as the Tor transport protocol.

In Figure 4.10, we notice that MPTCP speeds up the web downloads on average 15% compared to TCP. We also observe that the 20% fastest web downloads showed a decrease of 50% in TTLB. For bulk clients, we observe a similar performance gain when MPTCP is used. However, a small fraction of clients experience slower downloads than with TCP. This may be explained by the existence of slow connections between ORs, where more subflows cannot get more capacity because MPTCP introduces the overhead of managing multiple subflows. We did not include the results of TTFB, as we did not observe any noticeable difference between MPTCP and TCP in this experiment. Overall, we notice that MPTCP allows a fairer allocation of bandwidth resources, which leads to performance improvements for the majority of clients.

4.7.5.4 Fairness achieved for Tor Traffic

We next analyze how the demonstrated performance boost arises from a fairer bandwidth allocation to Tor traffic. To this end, we differentiate two types of fairness. First, the intra-fairness between coexisting circuits in the MPTCP connection. Since the multiple subflows transverse paths with similar characteristics (ORs use the ndiffports setting to communicate using a single network interface), the independent congestion control acts similarly for all subflows. Furthermore, a subflow does not carry traffic from exclusively one circuit (the kernel scheduler is agnostic to the origin of the Tor traffic). Hence, we can conclude that with MPTCP in Tor, traffic from different circuits would have a fair bandwidth allocation on the shared connection.

The second type, the intra-fairness, is related to other concurrent applications. Here, we need to analyze whether the bandwidth allocated to Tor traffic comes at the cost of unfairly limiting the resources of other TCP applications on the same channel. Although the suggested uncoupled congestion control causes that Tor traffic of n circuits receives $n-$times more bandwidth than other applications, it is important to emphasize that this traffic is carrying data originated on multiple logical connections — the Tor circuits. Thus, we consider inter-fairness is even higher than with regular Tor, while the number of subflows equals the number of circuits. As part of our anonymity analysis, we measured how accurately the creation of subflows corresponds to the number of circuits (see Figure 4.12). Thus, we can affirm that MPTCP in Tor is consistently fair with other TCP applications because the creation and subflows *only* relies, and is highly responsive, on the active circuits at any moment. Again, this is fairer for Tor traffic than when using TCP, because in that case, Tor circuits do not achieve the deserved amount of bandwidth.

4.7.6 Anonymity Evaluation of MPTCP-Tor

Beyond the demonstrated performance gains, it is necessary to study whether using MPTCP as the Tor transport protocol leads to degradation of users' anonymity — the main feature pursued by Tor users. We now evaluate how MPTCP may create attack surfaces that allow further anonymity leaks. We first analyze the *socket exhaustion attack* [53], which is applicable to designs that employ multiple transport connections between ORs. Second, we evaluate a novel attack vector that

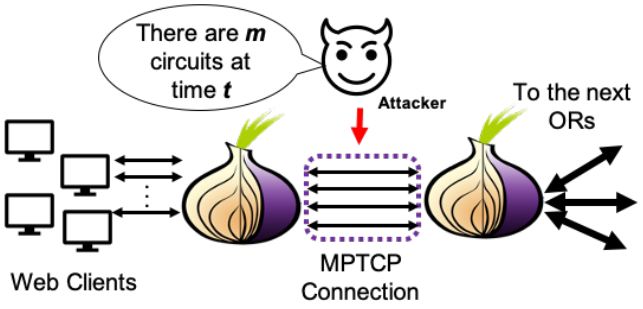


FIGURE 4.11: Connection profiling scenario

emerges when using MPTCP. We call it *connection profiling*. We evaluate the effectiveness of this procedure, describe the potential exploitation strategies, and suggest mitigation techniques.

4.7.6.1 Experimental Setup

We conduct experiments to estimate the effect of anonymity threats on an isolated local Tor network in which we started a limited number of clients and ORs. Contrary to performance evaluations, here we do not aim to realistically represent the Tor network, but to find worst-case scenarios and identify the impact of the attack over specific ORs connections. Furthermore, with this we do not menace the security of the real Tor network.

For our anonymity evaluations, we set up a similar network as the one used for our performance evaluations. We also developed an tool to dissect MPTCP traffic. This tool analyzes sequences of packets collected via tcpdump and separates the packets corresponding to each identified subflow. With this, we can infer the number of subflows at any time, and can isolate packets traveling through each subflow.

4.7.6.2 Socket Exhaustion Attack

In designs where the number of circuits determines the number of transport connections, it is possible that an attacker, acting as a client, maliciously creates a large number of circuits to surpass the maximum number of sockets. In Linux-based system , this limit is 4,096 open sockets, and for other operative systems is 15,000. Once reached that limit, a new petition to open a socket will ultimately block an OR from serving legitimate traffic. Tearing down ORs with this attack may increase the chances for a user to use a malicious OR instead. MPTCP is resilient to this threat because the path manager restricts the creation of new subflows when the per-connection limit is reached.

4.7.6.3 Connection Profiling

This process can be seen as a first step for anonymity attacks. The goal is to infer circuit-to-connection mapping information — currently hidden in Tor for privacy reasons — by only observing network-level traffic. This becomes feasible in MPTCP because fields that indicate subflow associations (e.g., ports and token) are visible to any agent sniffing the traffic exchanged by an OR. Thus, the attacker can easily infer the number of circuits by counting the number of subflows (while the number of subflows has not reached the limit of 32). We describe the operation of this procedure in Figure 4.11. Specifically, a non-global eavesdropper — as assumed by the Tor's adversary model — that can count how many circuits exist in a Tor transport connection, could conduct the following attacks: *(i)* a cheaper and more destructive denial of service (DoS) attack, by addressing connections with more clients. *(ii)* a refined traffic-analysis attack such as website fingerprinting [19, 103, 109]. The middle OR's ISP can eventually discover the website visited by Tor users when traffic flow originated from a single circuit is detected. As a consequence of *(i)* and *(ii)*, censorship and blocking can be adapted to ORs carrying more active circuits, or based on the destination.

We evaluated an adversary conducting connection profiling in our Tor local network testbed. We started nine Tor clients sharing an MPTCP connection and downloading data at different times and durations. Using *tcpdump* and the tool we deployed to dissect MPTCP traffic, the attacker eavesdrops this shared connection, and aims to create a profile of the number of circuits that are active on the connection over time. This profile is, in turn, compared against the real circuits' activity logged on the target OR.

In Figure 4.12, we present the real and detected number of circuits for the target connection. We observe that, excepting a marginal number of misdetections, in this preliminary experiment, the adversary consistently detects the correct number of circuits with less than one second delay. Such offset is expected as there is a time difference between the subflows' events are logged in the OR and their detection by the attacker. However, as the time span analyzed by the adversary increases, the connection profiling becomes more accurate and useful for the intended adversarial purposes (e.g., a directed DoS attack). Still, this attack is trivially possible only until the maximum number of subflows is reached (by default 32).

4.7.6.4 Countermeasures

We identify three strategies to protect against adversarial connection profiling. *(i)* using IP-level encryption (e.g., IPSec) to hamper subflows recognition. This encryption should not expose new vectors for traffic-analysis attacks. *(ii)* inserting noise, in the form of random delays, into the subflow closing/opening events. The inserted delays should be inside an acceptable range to maintain the bandwidth fairness. And *(iii)*, allowing clients to use a variable number of circuits [13, 110] or subflows, to minimize the leakage of the clients-to-circuit relationships. Here, it is necessary to find a trade-off between fair bandwidth allocation and privacy protection.

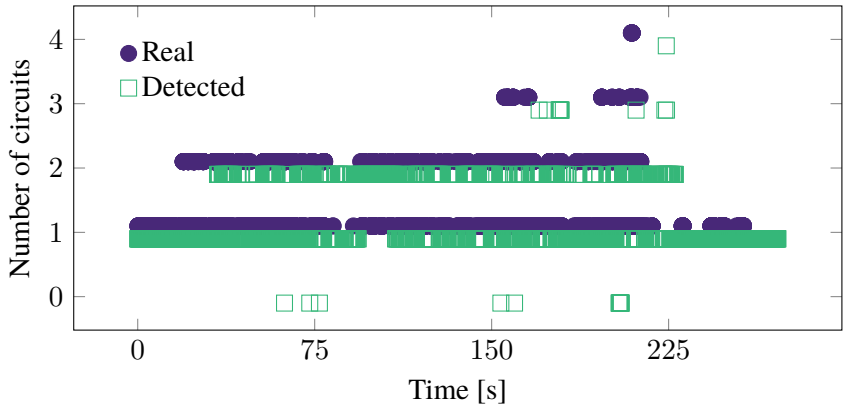


FIGURE 4.12: Evaluation of connection profiling

4.7.6.5 Comparison with Prior Designs

Once we have shown the performance benefits and the anonymity implications of using MPTCP as a Tor transport protocol, we next discuss how our approach differs from the designs that also propose using multiple transport connections in Tor. We aim to highlight the deployability benefits of MPTCP compared to prior designs.

TCP-over-DTLS & PCTCP. Conceptually, both designs operate similarly because they provide per-hop reliability and open one TCP connection per circuit. PCTCP claims to be highly efficient because it is fully implemented at kernel level. We argue that MPTCP is easier to deploy because, besides being at kernel level, it does not require additional external modifications (e.g., IPSec in PCTCP). Moreover, contrary to PCTCP, MPTCP is resilient to socket exhaustion attacks.

Torchestra. This approach is similar as MPTCP operating with two fixed subflows. While it is true that MPTCP cannot assign circuit's traffic to a subflow based on its volume, the scheduler can decide at kernel level what the path with the best RTT for each packet is. We believe that this responsive reaction is more advantageous than the slower and non fully-accurate traffic classification algorithm done at circuit level by Torchestra.

IMUX. The main difference between IMUX and MPTCP is that the scheduling decisions are done at kernel level in MPTCP, and in the user space (inside Tor) in IMUX. Thus, IMUX implements an additional scheduler to decide how to allocate packets among several independent TCP connections. Since network conditions would be recognized more responsively in the kernel space, the MPTCP's scheduler could detect more accurately than IMUX the best path for each packet.

To sum up, we believe that using MPTCP as an out-the-box protocol for Tor has significant design advantages compared to other proposals that also employ multiple transport connections.

4.8 Summary

This chapter started with an extensive analysis of existing anonymous communication systems that use multipath routing. To this end, we presented a taxonomy that revises all the design choices. Besides providing a clear representation of the different elements of Tor, our taxonomy sheds light to further unexplored options that can lead to performance and privacy improvements (e.g., implementing flow control at the multipath layer). Particularly, we showed the distinct options to split and merge traffic when using multiple circuits. We also described the possible techniques for handling data streams arriving from diverse paths and associate them to a single user. We identified that the existing Tor-based approaches limit their field of application to the utilization of multiple circuits, splitting and merging traffic at the OP and exit OR, for improving performance. Furthermore, we observed that the strategies used by those approaches rely on modifying the Tor cell format to include an extra sequence number. We believe this is may hinder a further adoption, as it would become infeasible to enable ORs to deal with multiple cell formats.

A special insight found through this exploration was the opportunity to use multiple routes at the transport layer. In response to this, in the second part of this chapter, we presented a novel transport protocol for Tor. We proposed to replace TCP with MPTCP and provide fairness to the Tor traffic that transverses a shared link between two ORs. With this out-of-the-box and easy-to-deploy transport design, Tor clients experience noticeable faster download times. Lastly, we have identified a possible leakage — the connection profiling — of using MPTCP as transport protocol in Tor. To counter this, we introduced countermeasures to minimize adversarial usages of the circuit-to-connection relationships.

I am very seldom interested in applications. I am more interested in the elegance of a problem. Is it a good problem, an interesting problem?

Claude Shannon

5 Improving Privacy with Multipath Routing

5.1 Introduction

The taxonomy explored in the previous chapter shed light on the design aspects that can be enhanced to boost the anonymity of Tor users. Specifically, we have identified that multipathed Tor clients distributing traffic via multiple entry ORs may resist WFP attacks performed by a malicious entry OR. While it is true that other more powerful adversaries placed in the client-to-entry connection (e.g., an ISP) would still be able to conduct WFP against multipathed Tor clients, we believe our proposed technique is relevant because becoming an entry OR is easier and cheaper (and, thus, more dangerous for the user) than an ISP.

Over the years, multiple studies have systematically shown the continuously improved effectiveness of WFP attacks [20, 21, 72, 75] and their applicability in real-world settings [34, 73]. This growing number of powerful WFP attacks and the lack of effective and feasible WFP defenses highlight the need to design easily-deployable and efficient countermeasures against WFP. In contrast to the existing defenses (see Section 3.3.3.1), our proposal does not intentionally add or delay packets to achieve distortions of the traffic patterns that lead to the obfuscation of WFP features. We argue that a well-crafted distribution of traffic through several paths can destroy repeatable patterns useful for a WFP attack at a marginal overhead cost. Additionally, using traffic splitting does not require any a-priori knowledge about the websites to be protected, which eases the adoption and deployability of our proposal. In the following of this chapter, we conduct an extensive analysis of several strategies to achieve this, and then, we evaluate the most effective option under real-world Tor settings.

The contributions presented in this chapter are as follows:

1. We design a novel and effective WFP defense based on the idea of traffic splitting over multiple entry ORs.

2. We explore several traffic-splitting strategies, which can serve as candidates for adoption in our defense. Our comprehensive analysis comprises simulations and real-world evaluations considering today's state-of-the-art WFP attacks and advanced strategies that an adversary may use.

3. We conduct an extensive analysis of the effectiveness of our defense using the best splitting scheme. We show that our defense outperforms all prior works in terms of accuracy reduction (below 16%) while offering attractive performance (inserted overhead is minimal).

4. Lastly, we explore whether traffic splitting can contribute to mitigate end-to-end correlation. To this end, we present a preliminary experiment where we apply the most recent and powerful end-to-end correlation attack against traffic protected with our WFP defense. We show that splitting traffic through only two entries is enough to make practically infeasible this attack when the adversary observes one entry and the respective exit OR.

5.2 Splitting Traffic over Multiple Entry ORs

Since we aim to use traffic splitting to counter malicious entry ORs mounting WFP, we first need to determine the design changes required in Tor. With this, we can then study how the traffic must be distributed to prevent a single malicious entry OR mounting a successful WFP attack.

As depicted in our taxonomy (see Chapter 4), when the multipath approach utilizes multiple entry ORs, the splitting/merging points must be on one side the client, and on the other side the middle OR, exit OR, or web server. Although for achieving WFP protection the second merging point can be placed at any position after the entry ORs, a proper selection of this point leads to an easier further adoption of our defense and produces less load on the Tor network. If the web server is the merging point, the defense would be limited only to those websites supporting the multipath capability. Thus, we discard this option for our traffic-splitting defense. We could also leverage the designs of Conflux or mTor and splitting/merging traffic on the exit OR. However, this option would load with extra multipath-related operations the scarce — in quantity and bandwidth — exit ORs [24]. Therefore, we prefer to merge traffic at the middle OR because they are easier to deploy (i.e., each OR can act as as middle node by default), and their position in the circuit is not sensitive (i.e., neither the origin nor the destination are visible by the middle OR).

Figure 5.1 depicts the implementation of our multipath Tor design. This figure represents how a single malicious entry OR (denoted in red color) would observe only a certain portion of all exchanged packets in with the Tor network. We argue that the observed fraction of packets is not

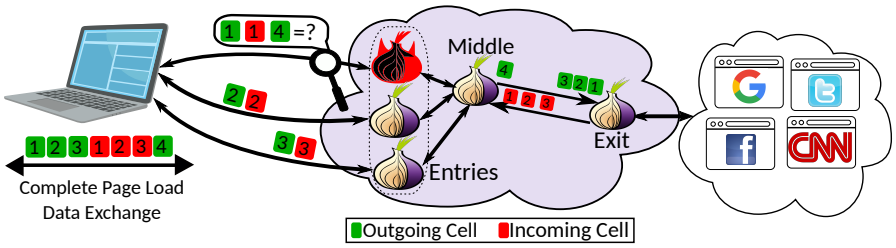


FIGURE 5.1: Design overview of our traffic-splitting defense.

sufficient for the malicious entry OR to determine which website the user is visiting. Our defense requires that the middle OR collaborates with the client to follow a certain traffic distribution technique and to associate traffic originated on multiple circuits. These aspects were considered in the implementation used for our experiments. This contains the necessary modifications on the Tor source code to split and merge traffic on the client and middle ORs for a reliable web browsing.

5.3 Traffic Splitting Strategies

The most important design choice for our defense to be effective against WFP attack is the way how traffic is split and distributed through multiple entry ORs. In other words, the splitting scheme should produce highly diverse traffic distributions among different loads of the same page in order to prevent an adversarial entry OR in identifying (repeatable) patterns. To find such a strategy, we analyze the influence of *(i)* the *number of distinct entry ORs* used to establish multiple paths through Tor, and *(ii)* the *percentage* and *diversity* of traffic observed at each of the entry ORs.

5.3.1 Number of Entry ORs Used for Traffic Distribution

To apply our defense, we first need to determine the number m of distinct entry ORs utilized by the Tor user for a multipath communication. While a large number of entry ORs decreases the amount of information available to each entry OR, it also increases the likelihood of selecting a malicious entry OR for a circuit belonging to a single multipath user connection (see Section 4.5). Thus, we explore how m influences the user's protection against WFP attacks and propose a trade-off between privacy protection and performance overhead.

5.3.2 Distribution of Traffic over Multiple Circuits

Having selected the number of entry ORs, we need to define how to distribute the web traffic. We explore several techniques, from trivial to more sophisticated possible strategies, to find the most effective one against WFP attacks.

The first considered strategy, and the most basic one, is *round robin*. Here, we switch to the next circuit for each Tor cell. Our goal is to analyze the level of security provided by such a

simple splitting scheme and assess whether only the multipathing, or the use of a sophisticated splitting strategy, is more important to build an efficient WFP defense. A notorious disadvantage of this strategy is that it is deterministic, and thus, a traffic fragments may be distinguishable among different page loads of the same website.

We also analyze *random* splitting, in which we randomly select a circuit for each Tor cell. Although this strategy inserts some randomness, portions of traffic traces will keep proportionality in the long term, which may still be useful for an attacker.

We further consider to split traffic *by direction*, i.e., we use one circuit for incoming and another circuit for outgoing Tor cells. The intuition suggests that the incoming/outgoing relationships contain significant characteristics of a website. However, as many WFP attacks consider the number of packets per direction as independent features, observing one-direction traffic may still contribute to the classification.

To increase the diversity of the traffic distribution for repeated page loads of a common website, we also evaluate a *weighted random (WR)* scheme. Here, we pursue to make less distinguishable long-term traffic patterns. In WR, for each page load, we create a separate vector $\vec{w}$ consisting of m probabilities for each splitting point, which, in turn, are computed from a m-dimensional Dirichlet distribution. We use these probabilities to weight the selection of an entry OR for each cell transmitted between the user's OP and the middle OR. With this weighted-random selection, we seek to conceal size-related patterns that may remain among several page loads. Through this biased random selection, each web page load tends to show diverse lengths and therefore the number of packets and their sizes are more obfuscated than with a uniform random choice. We used the Dirichlet distribution as it is a common way to model random probability mass functions for finite sets. As required by our strategies, it outputs directly m random positive values that add up to one. Moreover, this distribution is statistically more precise and faster than first generating m independent random values and normalizing them.

Finally, we consider a *batched weighted random (BWR)* splitting strategy. Contrary to WR, here, the vector $\vec{w}$ is utilized to weight the choice of an entry OR for a *batch* of n Tor cells. We aim to create more confusions to the WFP classifier by creating chunks of cells highly diverse but belonging to the same web page load. As we do not insert any dummy traffic and constant-size batches, traveling across distinct circuits may still reveal useful traffic patterns. Thus, we update n constantly during a single page load, i.e., after each batch. While a small n converges BWR to the WR strategy, a large n may not produce a sufficient level of traffic randomness, e.g., for small websites, as the splitting of traffic will be limited or even may not occur at all. Based on our empirical analysis (see Section 5.4.3), we adjusted n to be uniformly sampled from the interval $n \in [50, 70]$. Algorithm 1 shows the detailed process to select the entry for any cell of a certain web page load. The BWR algorithm is followed independently on both directions, for incoming (middle-to-client) and outgoing (client-to-middle) cells respectively.

Algorithm 1 Batched Weighted Random (BWR) strategy for each cell within a web page load (for each direction)

Input: Number of entry ORs (m), low limit interval (a), high limit interval (b)
Output: Index of the selected entry OR for each cell ($selectedEntry$)
$\overrightarrow{weights} \leftarrow Dirichlet(m)$
$batchSize \leftarrow RandInt(a, b)$
$randomEntry \leftarrow RandomWeighted(m, \overrightarrow{weights})$
for each cell **do**
    $selectedEntry \leftarrow randomEntry$
    if (Number of sent cells = $batchSize$) **then**
        $randomEntry \leftarrow RandomWeighted(m, \overrightarrow{weights})$
        $batchSize \leftarrow RandInt(a, b)$
    end if
end for

5.4 Finding the Most Effective Splitting Strategy

In order to determine the most effective parameters for our defense. We first conduct an extensive evaluation of the described splitting strategies within a simulative framework. With this, we can test several options and design variations of the proposed designs. We assume that the attacker controls one malicious entry ORs in the victim's multipath connection. For all experiments, the adversary is aware of the applied defense and its splitting scheme. We further assume that the attacker has enough resources to collect a representative training dataset. Intuitively, an attacker would prefer to train with traces protected with the same defense and splitting scheme as the client would use. Still, we also conducted preliminary evaluations to know whether training with non-defended traces may help for classifying protected traces. These results (accuracy below 2%) confirmed our intuition, and demonstrated that an adversary would not opt for training with non-defended traces (this is also observable in our experiment in Section 5.5.5). In particular, the best strategy for the attacker is to use all collected subtraces of each page load. The adversary represents these subtraces as separate input vectors belonging to the same class and generates features for these vectors to train the WFP classifiers.

We also ensure that all subtraces of the same page load are used either for testing or for training only. Using only single, randomly-chosen subtraces from each page load leads to lower accuracy and will not be the dominant strategy of the adversary. We believe to have addressed the logical advancements for the attacker in the learning phase and deem it unlikely that further improvements can be made without extensive efforts (which would also require enhancements to today's WFP classifiers).

Once the best strategy has been found, we further conduct an extensive evaluation under real-world Tor settings.

5.4.1 Simulation of Traffic Splitting

To initially evaluate the efficiency of the presented traffic-splitting schemes, we developed a simulator that artificially splits real-world traffic traces based on the selected scheme. We refer to the order of all cells that are assigned to the same path after splitting as a *subtrace*. In total, our simulator creates m subtraces for a single page load. To obtain realistic results, our simulator further takes the latency of the different circuits between the user and the middle OR into account when employing our multipath transmission scheme. To this end, we measured the RTT of several circuits consisting of the same middle and exit ORs but different entry ORs in the real Tor network. As in previous work [111], we measured the RTTs by sending a *relay connect* cell to *localhost* and triggering the reply time. In total, we gathered RTTs for 4,073 successfully built circuits, which we integrated into our simulator. The source code of our simulator is available in [112].

5.4.2 Experimental Setup

For this simulative analysis, we conduct several closed-world evaluations. We rely on a dataset consisting of the 100 most popular sites [113]. First, we collected 100 traces for each website without applying our defense and refer to this *non-defended* dataset as `ALEXA-NODEF`. Then, we replayed these traces through simulator using each traffic splitting scheme varying the number of utilized entry ORs. We rely on these closed-world experiments for finding the best strategy, which will be used in further realistic evaluations presented in next sections.

During crawling of our datasets, as in related work [20, 75, 80], we excluded websites that deny user traffic coming from Tor, show a CAPTCHA, have no content, or redirect to other sites that are already present in our dataset. We also removed page loads indicating a client or server error as the attacker is not interested in fingerprinting broken page loads. We applied the automated approach presented in [75] to collect all traces. For each page load, we recorded metadata such as the size and direction of the transmitted TCP packets by using a toolbox containing the Tor Browser 9.0.1 and *tcpdump* and then reconstructed the Tor cells by applying a previously-used data extraction method [75]. For our evaluation, we focus only on Tor cells, as the different layers for data extraction (e.g., TCP packets, TLS records, or cells) only have a marginal influence on the classification results [75]. Hence, our results are comparable for other extraction formats.

For our evaluation, we considered four state-of-the-art WFP attacks: k-NN [72], CUMUL [75], k-FP [21], and DF [20]. For all following experiments where we do not explicitly mention a different methodology, we apply 10-fold cross-validation with respect to the total number of collected page loads, i.e., the data is split into 10 evenly large parts, i.e., *folds*. Then, the entire process of training and testing is repeated 10 times, using one of the 10 folds as test data and the remaining nine folds as training data in turn. For this closed-world analysis, we computed the *accuracy*, i.e., the probability of a correct prediction (either true positive or true negative).

TABLE 5.1: Accuracy (in %) for different intervals of the batch size n needed in our BWR strategy.

Batch size n	[30, 40]	[30, 90]	[50, 70]	[60, 80]	[90, 120]
k-FP	16.05	17.98	**13.46**	18.00	17.76
DF	8.36	7.95	**6.58**	8.20	6.70
CUMUL	6.70	6.50	**4.63**	8.44	7.31
k-NN	4.75	4.50	**3.15**	5.20	4.80

We assume the attacker to be a passive observer that does not decrypt, modify, or interrupt transmitted packets. The attacker has a limited view of the Tor network by controlling a restricted number of entry ORs. Thus, he can monitor traffic exchanged with a Tor user and is aware of the user's identity, but does not know which website the user is visiting. Moreover, we assume that the adversary has sufficient computational power to train several fingerprinting techniques on large training datasets.

5.4.3 Determination of Optimal Splitting Scheme

To identify the optimal traffic splitting scheme for our defense, we use the simulator presented in Section 5.4.1 to artificially split the non-defended traffic traces in `ALEXA-NODEF`. For each traffic splitting scheme, we generated a separate dataset containing artificially created defended traces. Each splitting scheme was evaluated for a certain number of desired entry ORs between two and five for all page loads, and for a variable number of entries between two and five that is randomly decided for each page load.

Before presenting the results for all variations of each splitting scheme, it is first necessary to determine the complete settings for BWR. As previously described, this strategy selects an entry OR to send a batch of cells of variable size between two values. Therefore, we first conduct an empirical evaluation of this specific strategy against all WFP classifiers to determine the optimal range. Since some state-of-the-art WFP attacks rely on features extracted from consecutive sequences of 30 to 40 Tor cells within a given traffic trace [21, 72], we argue that the number n of cells in a single batch should lie around those values in order to disturb useful features. Based on this, we investigated different intervals for n by using five entry ORs for each user's multipath connection and summarize the classification accuracy obtained for each of these intervals in a closed-world scenario in Table 5.1. We present the experimental results for four state-of-the-art classifiers, k-FP, CUMUL, k-NN, and DF. As we can see, the most promising results were obtained when uniformly sampling n from the interval $n \in [50, 70]$. Hence, we consider this interval as a good choice and use it for all experiments.

Table 5.2 details the accuracy of each WFP classifier in a closed-world scenario without defense (row "Undefended") and against our evaluated splitting strategies for varying numbers m of entry ORs, where the attacker controls one of them. We next discuss the effectiveness the number of utilized entry ORs and the different splitting schemes.

TABLE 5.2: Accuracy (in %) of state-of-the-art WFP attacks in scenarios without defense and against our splitting strategies, where m indicates the number of entry ORs used in user connections.

Our Splitting Strategies	m	k-NN	CUMUL	k-FP	DF
Undefended	1	98.20	98.50	98.40	98.75
Round Robin	2	91.29	96.30	96.73	97.56
	3	89.52	86.88	94.83	95.91
	4	87.25	85.04	93.21	94.64
	5	86.59	82.21	92.22	93.01
	⟦2,5⟧	75.54	80.07	83.66	90.25
Random	2	87.74	95.51	94.48	96.01
	3	82.51	92.64	91.51	94.77
	4	78.09	89.71	89.13	93.43
	5	72.09	87.02	86.41	90.31
	⟦2,5⟧	62.28	79.34	76.59	84.05
By Direction	In	37.05	37.43	59.07	29.99
	Out	32.45	26.71	56.17	26.15
Weighted Random	2	16.14	63.05	55.01	65.94
	3	9.66	53.16	45.76	54.21
	4	4.32	47.90	42.74	47.77
	5	4.38	41.62	40.55	42.33
	⟦2,5⟧	4.57	47.92	41.31	47.79
Batched Weighted Random	2	6.89	21.22	33.37	31.82
	3	4.49	9.11	21.33	31.82
	4	3.62	4.76	16.48	6.91
	5	**3.15**	**4.63**	**13.46**	**6.58**
	⟦2,5⟧	3.22	8.60	18.24	11.44

5.4.3.1 Number of Entry ORs Used

First, we analyze how the number of entry ORs used influences the accuracy of WFP attacks, as summarized in Table 5.2. Independent of the chosen strategy, we observe that all WFP attacks become less effective when the user utilizes a larger constant number of entry ORs to fetch a website. We further notice a slight decrease of the classification accuracy for a variable number of entry ORs (rows "⟦ 2,5 ⟧") and a constant $m \geq 4$ regardless of the splitting strategy.

In case of a variable number of entry ORs, the adversary is challenged by the uncertainty of the applied splitting strategy as website-specific patterns are less deterministic. Additionally, our most effective scheme, BWR, drops the accuracy of all classifiers to less than 14% when a constant number of five entry ORs is utilized. Therefore, we consider $m = 5$ as a good choice and argue that this choice neither significantly increases circuit establishment times (current versions of Tor already build three circuits preemptively [114]) nor dramatically increases the probability of selecting a malicious entry OR (see Section 4.5). Our experiments confirmed our initial intuition that the data observed on a single client-to-entry connection by a malicious entry OR is not sufficient to perform WFP attacks.

5.4.3.2 Efficiency of Different Splitting Schemes

To find the most suitable splitting method, we now explore the efficiency of each strategy, as summarized in Table 5.2.

Round robin: Overall, we notice a slow decrease in accuracy with the round robin strategy as the number of entry ORs used increases. Since this strategy is completely deterministic, subtraces' features among different page loads remain similar and thereby useful for the attacker. Moreover, this scheme cannot completely hide the total size of a given website — one of the most important features for WFP attacks [21] — even when observing only a fraction of the page load. Although varying the number of entries further contributes to accuracy reduction, the successful rate for the attacker remains significantly high.

Random: We observe a similar trend with a steeper decline for the random strategy. Still, the accuracies of CUMUL and DF remain comparably high, corresponding to the correct identification of most page loads. Although the inserted randomness lowers the accuracy more than for round robin, such high values are still unacceptable for a WFP defense.

By direction: A simple scheme, which only splits the traffic by direction, already delivers a significant decrease in accuracy for all WFP attacks. Even though the number of transferred cells per direction remains unaltered, most classifiers only recognize a third of the page loads. This drop might be caused by the classifiers' inability to retrieve information about the relationship between incoming and outgoing cells. However, despite the absence of this characteristic, k-FP profits from other features that use available information on timing and data rate per direction, contributing to a comparably high classification rate for this attack.

Weighted random: When applying a WR circuit selection, we observe a significant decrease in the accuracy compared to the previous schemes. All evaluated WFP attacks achieve less than 43% accuracy. For the worst-performing classifier, k-NN, the rate of reliably-detectable page loads drops below 5%. We believe that this significant decrease is caused by the diversity in total size among the different subtraces of a single website.

Batched weighted random: In case of BWR, we further reduce the accuracy of all WFP attacks down to 14%. In particular, the accuracies of CUMUL and DF cannot achieve a detection rate higher than 7%. For the worst-performing classifier, k-NN, the rate of reliably-detectable page loads drops below 4%. We believe that the significant difference in the accuracies between WR and BWR is caused by the fact that WR cannot fully destroy consecutive sequences of Tor cells within a given traffic trace, often exploited by WFP attacks to extract features. Another reason for the significant accuracy decrease achieved by BWR is the diversity in total size among the different subtraces of a single website. A notable observation is that a variable number of entry ORs (row "$[\![2, 5]\!]$") does not improve the efficiency of both splitting schemes, WR and BWR, as these schemes already introduce sufficient diversity by design.

Other evaluated schemes: Despite BWR already provides an attractive level of security

against WFP attacks, we also experimented with other variations and combinations of traffic-splitting schemes aiming to provide protection without requiring numerous entry ORs. Unfortunately, these schemes did not achieve the protection of our best strategy (BWR). Still, we next detail the obtained results to also show what direction should not be pursued in the future when designing further strategies for our defense. In *WR + by direction*, we tried to combine the low accuracy of one-direction traffic splitting with the randomness of WR. We experimented with $m = 4$ entry ORs with the restriction that each pair of entries transport only either outgoing or incoming traffic. The best classifier (k-FP) still reported accuracy superior to 45%. We also tried a variation of *WR with changing weights*. Here, the weights did not last for an entire web page load but were refreshed after each packet is sent. This method tends to converge to the simple random splitting technique, and thus accuracy remained high (> 70%). We also evaluated several variations of BWR in order to find out the reasons of its effectiveness and potential improvements. These variants are next described as part of our in-depth analysis of the batched weighted random scheme.

5.4.4 Success Factors for Traffic-Splitting Strategies

Our experimental analysis showed that BWR achieves a high protection level against WFP attacks. We next analyze the key factors for its effectiveness and present variations for particular use cases. As we mentioned before, two aspects determine the success of a splitting scheme, *(i)* the number of entry ORs used, and *(ii)* the percentage and diversity of traffic observed by the attacker. For *(i)*, the immediate effect is the creation of traffic chunks of certain size. Thus, it is important ensure that the strategy indeed guarantees that every traffic trace will be split. For the aspect *(ii)*, it is necessary to estimate the quantity of diversity introduced. Hence, we set a convenient metric to estimate this diversity for all methods. For instance, we can intuitively affirm that the round robin technique inserts a near-to-zero diversity value because the size-related features of the traffic splits remain similar among different web page loads. We next elaborate each aspect in detail.

5.4.4.1 Number of Slices Generated by Traffic Splitting

Generally, it is desired that for every web page load, the strategy splits the trace intro multiple slices. Otherwise, there might appear the risk that non-split (or even less-split) traces become more attractive for the attacker, which, in turn could target WFP successfully to specific traces of certain web pages.

To quantify this effect, we counted, from the universe of 10,000 traces of the `ALEXA-NODEF` dataset, how many of them were split into $1, 2, \ldots, m$ chunks. These counts are expressed in a vector H in which the $i-$th element is the number of traces that were divided into i splits. While for strategies such as round robin, we expect that this vector demonstrates that m splits were always generated, for WR and BWR it is possible that for certain page loads, weights vectors

TABLE 5.3: Number of splits generated by each strategy for $m = 5$.

	H	WFP attack	Accuracy (%)
Round robin	$[0, 0, 0, 0, 10000]$	DF	93.01
Random	$[0, 0, 0, 0, 10000]$	DF	90.31
By direction	$[0, 10000]$	k-FP	59.07
Weighted random	$[0, 4, 2, 4, 9990]$	DF	42.33
Batched weighted random	$[25, 138, 241, 1180, 8416]$	k-FP	13.46

TABLE 5.4: Number of splits generated by the BWR variations for $m = 5$.

	H	WFP attack	Accuracy (%)
BWR-Var	$[0, 45, 97, 4946, 4912]$	k-FP	24.19%
BWR-Strict	$[0, 44, 148, 692, 9116]$	k-FP	17.14%

$\vec{w}$ with unbalanced probabilities (e.g., $[0.95, 0.04, 0.01]$) are generated, and consequently some entries get never selected. Thus, traffic for certain page loads may not be split into the configured m parts.

Table 5.3 shows the vector H for each strategy when $m = 5$ (for the strategy by direction $m = 2$). As expected, for round robin and random all traces were divided in five parts. For WR, a marginal number of traces were split into less than five parts. This effect becomes notorious for BWR. We can count more than one thousand instances divided into four or less parts. Particularly, around 200 traces were not consistently protected as they contain, in several cases, all packets of the traffic trace. This effect is important to notice as a low degree of traffic division generates subtraces that contain abundant information of the total page load, which, in turn, may ease the classification task (see Section 5.5.4).

Although this leakage may seem harmless for the user, we believe it is important to propose alternative strategies to avoid that such leakages might be potentially useful for future attacks. For instance, an adversary can target WFP on certain pages that he knows are usually not split, and, consequently, be confident to achieve high accuracy. Therefore, we implemented two slight variations to BWR to avoid, at least, non-split traces. In the first one, called BWR-Var, we decreased the batch size to $n \in [10, 20]$ to force more random selections, and in the second one, BWR-Strict, we avoided that two consecutive batches select the same entry OR.

Table 5.4 shows the vector H and the highest reported accuracy for these BWR variations for $m = 5$. It can be noticed that we minimize the existence of less-split traces at the cost of a noticeably higher accuracy attack. Hence, the dilemma is to prioritize either low accuracy or non-split traces avoidance. Considering the main goal of WFP defenses (overall accuracy drop), and that the occurrence of less-split subtraces is marginal, we primarily used BWR without variations in our implementation. However, we also explore in 5.5.4 whether the less-split traces outbreak is beneficial for an adversary.

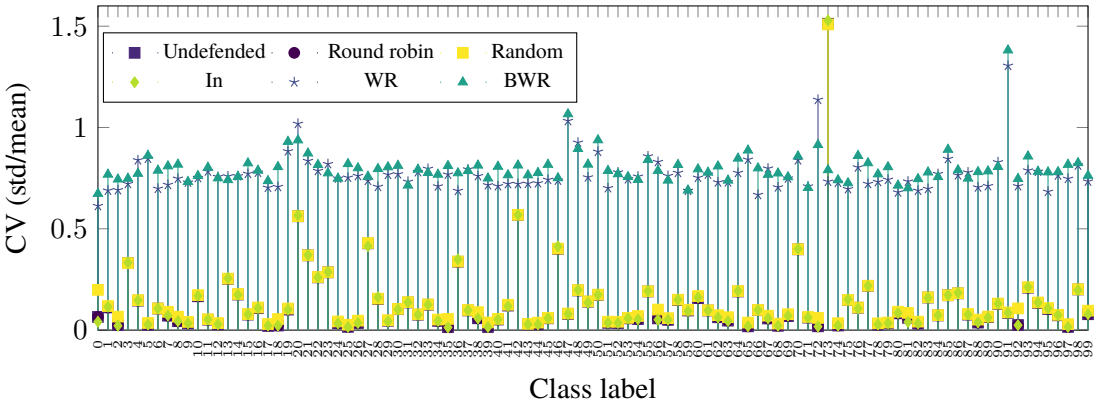


FIGURE 5.2: Coefficient of variation (CV) of the lengths of all instances for each class (web page) after applying each splitting scheme ($m = 5$)

5.4.4.2 Diversity Inserted over Traffic Traces

We now evaluate the amount of diversity inserted by our strategies. We aim to find a metric that explains the success of a splitting scheme. With this, we establish a parameter that can be considered when designing further techniques. For this purpose, we rely on the coefficient of variation (CV) to estimate the diversity of distinguishable features among several page loads of the same site. The coefficient of variation is a statistical measure that quantifies the diversity or deviation of a set of data points. Contrary to the single standard deviation, the CV generates values that are comparable between data series of different characteristics. Mathematically, the CV of a data series is the standard deviation divided by the mean.

We computed the CV for each strategy for one of the most important features considered in WFP attacks [101, 115], the total count of packets. As we are interested on the variation of these features over several instances of the same website, we calculate the CV for this feature over all instances of each class.

Figure 5.2 shows the coefficient variation of the total packet counts of all instances of each website (class). We notice how the CV for simple strategies (i.e., round robin and random) presents negligible differences with the undefended case. Most of the values lie under 0.2, which means that the dataset has a very low variation. This also explains why it is possible for the attacker to recognize useful patterns for successful classification. On the other hand, WR and BWR generate variation coefficients above 0.5. This means that different subtraces of the same page load have a very wide range of sizes. This, in turn, obfuscates all size-related features and complicates the WFP attack.

To sum up, it is important to guarantee high diversity among useful WFP features. Our strategies BWR and WR generate that the observable traffic patterns largely vary from one instance to another, leading to confusions for the WFP classifier. Thus, we suggest that any further scheme to be integrated into our defense should produce comparable CV values to those generated by WR and BWR.

5.4.5 Summary

In this section, we have conducted an extensive simulative analysis to find out the most effective splitting scheme to counter today's WFP attacks. The BWR strategy outperformed all other proposed schemes and reduced WFP accuracy to less than 14%. Thus, it will be considered for the rest of evaluations of this chapter.

Furthermore, our in-depth analysis showed the characteristics that a successful splitting strategy should have. We presented alternatives in case it is necessary to guarantee the desired number of splits and suggested the coefficient of variation as an adequate metric to assess the degree of diversity introduced by the splitting technique.

5.5 Evaluation of our Traffic-Splitting Defense

In this section, we present an extensive evaluation of our splitting defense — using the best found splitting scheme — under realistic Tor settings. To this end, we implemented all strategies in the Tor source code [112]. Additionally, when required, we started a set of middle ORs supporting our defense and a pool of clients for fetching web pages.

Specifically, we first present an open-world evaluation. Then, we conduct the closed-world evaluation using traces collected in the real Tor network. We complement this study with the evaluation of our defense against more powerful adversaries that access to splitting-related information or to multiple entry ORs.

5.5.1 Experimental Setup

For our realistic closed-world evaluation, we collected a new dataset, referred to as `ALEXA-DEF`, with our WFP defense running on the real Tor network. To this end, several relays were launched in the real Tor network.

For our open-world evaluation, we built our background set referred to as `ALEXA-NODEF-BG`. For this dataset, we visited, without applying our defense, the 11,307 Alexa most popular web-sites, excluding the first 100 sites used to build our closed-world dataset. Then using our simu-lator we generated defended traces to compute, similarly to related work [21, 72], the TPR, i.e., the fraction of accesses to foreground pages that were detected, and the FPR, i.e., the probability of false alarms.

If not otherwise specified, the crawling process, the adversarial training strategy, and the training/testing data set separation are similar as for our simulative evaluation (see Section 5.4.2).

5.5.2 Open-world Evaluation

Next, we evaluate the efficiency of our defense using BWR with five entry ORs in an open-world scenario. In particular, we focus on a two-class simplified open-world scenario where the adversary aims to detect whether a single subtrace of a testing page belongs to the foreground set

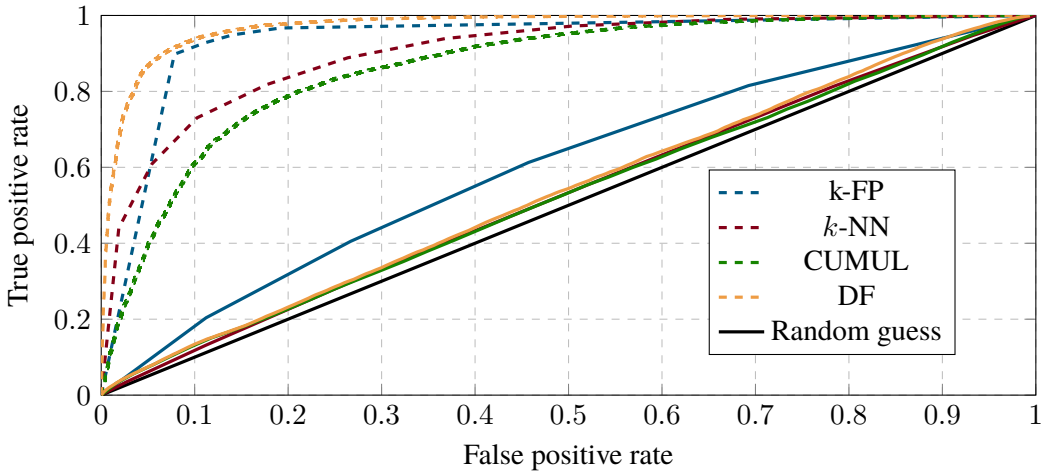


FIGURE 5.3: ROC curves for today's WFP attacks in open world. Dashed lines represent results for undefended traces and solid lines results for defended traces.

or not, without trying to identify the exact foreground website. Such a scenario is significantly more advantageous for the adversary. If we are able to reduce the success of the WFP classifiers in this setting, then their performance will be even worse in the scenario when the adversary aims to detect the exact foreground website.

As a baseline, we first used the non-defended datasets `ALEXA-NODEF` as a foreground set and `ALEXA-NODEF-BG` as our background set and computed the receiver operating characteristic (ROC) curve for each classifier. We then applied the simulator from Section 5.4.1 to artificially split the non-defended traces in `ALEXA-NODEF`, i.e., our foreground set, and the non-defended traces in `ALEXA-NODEF-BG` — our background set — using BWR with five entry ORs.

To quantify the open-world setting, we calculated the ROC curve for each classifier using these defended traces. As shown in Figure 5.3, we observe a clear proximity of the curves of all WFP classifiers to the random guess reference (black line), when TrafficSliver-Net is applied. While the area under the curve (AUC) indicating the detection of non-defended traces using all WFP attacks lies between 0.87 and 0.97 (AUC is one in case of a perfect classifier), the best-performing classifier, k-FP, reaches an AUC of only 0.60 when applying our defended dataset. Moreover, DF, k-NN, and CUMUL achieve a marginal higher AUC than random guessing (i.e., AUC = 0.5). Hence, the adversary is not able to conduct a successful WFP attack in an open-world scenario.

5.5.3 Closed-world Evaluation in the Real Tor Network

Once we have analyzed and identified the best traffic-splitting scheme and the optimal number of entry ORs, we deployed this strategy in the real Tor network and collected a real-world dataset (`ALEXA-DEF`). We then computed the accuracies of all state-of-the-art WFP attacks by using these defended traces as well as the non-defended traces from `ALEXA-NODEF`, which we use as a baseline. Table 5.5 summarizes the accuracy of each WFP classifier in a closed-world scenario.

TABLE 5.5: Accuracy (in %) of state-of-the-art WFP attacks against our defense.

	k-NN	CUMUL	k-FP	DF
Undefended	98.20	98.50	98.40	98.75
Our Defense	**5.02**	**5.18**	**15.44**	**8.07**

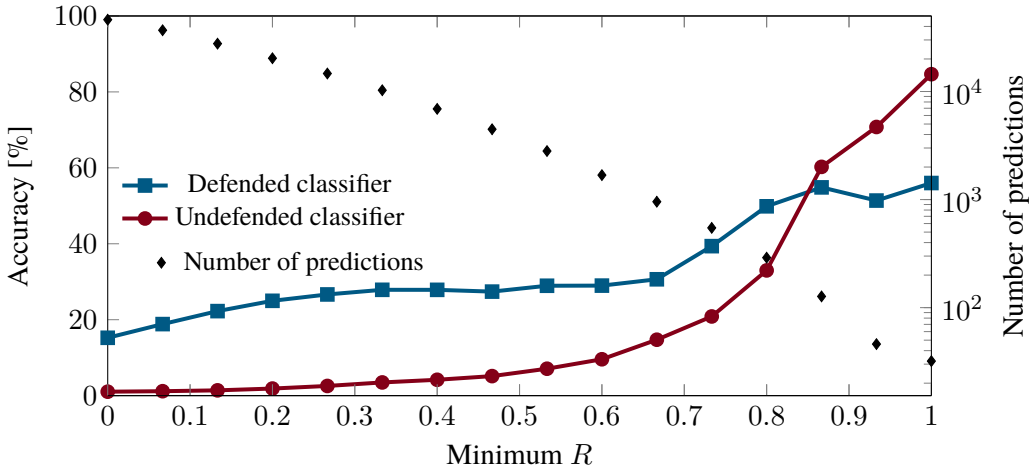


FIGURE 5.4: Accuracy achieved by both classifiers, one trained on defended subtraces and another trained on non-defended traces, when detecting *only* subtraces whose length is at least R of the complete page load. The number of evaluated predictions ♦ is referenced to the right log-scaled y-axis.

As we can see, the classification results obtained by using the defended traces gathered from the real Tor network are analogous to those achieved with simulated defended traces. In particular, we observe a dramatic reduction of the detection rate from more than 98% to less than 16% for all state-of-the-art WFP classifiers. Simultaneously, our defense does not introduce any artificial delays or dummy traffic to counter these WFP attacks. Thus, we can conclude the effectiveness of our network-layer defense in real-world settings.

5.5.4 Security against More Advanced Adversary

We further explore the performance of our defense against an unrealistically strong attacker that is even aware of the portion of traffic of a complete page load that is transmitted over an observed client-to-entry connection.

As the adversary uses own deployed Tor users running our defense in order to collect training data for WFP, he can further gather a number of traces representing various traffic distributions for the websites of interest. Based on a statistical analysis of the training data, the attacker can then assess the ratio R for real victim user subtraces. By using this knowledge, the adversary can build several classifiers trained on defended subtraces of different lengths or non-defended traces and, thus, improve the detection rate.

To analyze the level of danger for the users in this scenario, we assume that the attacker possesses two classifiers: one trained on non-defended traces and another trained on defended subtraces whose length represents a certain minimum ratio R from the corresponding complete

TABLE 5.6: Accuracy (in %) of WFP attacks for two malicious entry ORs during a single page load.

	Defended				
Training strategy	S_1	S_2	S_3	S_4	S_5
k-FP	5.85	25.23	30.16	28.14	35.90
DF	6.82	14.01	27.75	19.24	35.71
CUMUL	7.95	9.11	16.68	12.75	19.47
k-NN	5.78	6.56	8.39	13.47	9.11

TABLE 5.7: Accuracy (in %) of WFP attacks for n malicious entry ORs.

	Defended				Undefended
n	2	3	4	5	–
k-FP	35.90	55.92	80.62	96.52	98.40
DF	35.71	65.62	86.92	97.40	98.75
CUMUL	19.47	43.52	72.86	96.56	98.50
k-NN	13.47	29.94	52.11	94.29	98.20

page load. Then, we compute the accuracy achieved by these classifiers when the attacker tries to detect only subtraces whose lengths are at least a given minimum R. We executed this experiment by using both datasets `ALEXA-DEF` and `ALEXA-NODEF` and k-FP — the best-performing WFP attack against our defense. Figure 5.4 shows the results obtained for a closed-world scenario.

While the accuracy achieved by our classifier trained on defended subtraces is almost constant and remains below 25% for a minimum $R \leq 0.6$, the recognition rate of the other classifier trained on non-defended traces is close to zero for the same range of R. Hence, we can conclude that our traffic-splitting-based defense achieves good protection against WFP attacks as long as each client-to-entry user connection contains less than 60% of the total length of a given page load. In contrast, high detection rates are achieved only for subtraces comprising more than 80% of a complete page load.

Moreover, although the adversary can reach significantly higher classification accuracies when observing subtraces of larger length, the number of these subtraces that are generated by our defense is marginal compared to the total number of traces contained in our dataset. Despite this advantage is minimal, it emphasizes the necessity to keep at minimum non-split traces in certain cases. For instance, when those non-split traces correspond to a unique website, which we strictly desire to protect, any of the described BWR variations that prohibit non-split traces can be used.

5.5.5 Security against Multiple Malicious Entry ORs

Finally, we consider an even more-powerful adversary controlling n, $2 \leq n \leq 5$, malicious entry ORs in real victim multipath user connections and, thus, observing multiple client-to-entry connections utilized to load a single page. In this scenario, the attacker gains additional knowledge about the complete page load by merging the sequences of Tor cells that are transmitted

through the compromised client-to-entry connections into a single subtrace. However, in order to achieve high recognition rate, the adversary needs to also adjust the training strategy applied for the classifier. To find the best training strategy in case of multiple malicious ORs, we explore several alternatives that the adversary can use for the training process.

In our first training strategy, S_1, we assume that the adversarial classifier is trained on non-defended (i.e., non-split) traces. This is to evaluate whether a classifier trained with non-split traces can recognize a part of them represented by combined subtraces.

As in the previous sections, in the second strategy, S_2, we assume that the attacker uses all subtraces belonging to a single training page load as separate inputs for training the classifier. Here, we want to study if a classifier can recognize combined subtraces by learning only on parts of these combinations.

In the third strategy, S_3, the traces for training consist of the ordered sequences of Tor cells that were transmitted through n *consecutive* client-to-entry connections (e.g., if the total number of entry ORs used by a user is four and the number of malicious entry ORs is two for a given page load, a single training trace is the union of two subtraces traversing the *first* and the *second* entry OR).

We further study another strategy, S_4, where the adversary builds training traces by merging n *randomly-chosen* subtraces. In our last strategy, S_5, the adversary creates all possible combinations of training traces that consist of n merged subtraces (i.e., $\binom{m}{n}$ training traces in total). For all experiments, the testing traces consist of n randomly-chosen, merged subtraces (this represents what an attacker would observe in reality). First, we evaluated these strategies by using our defended traces in `ALEXA-DEF` against $n = 2$ malicious entry ORs.

Table 5.6 shows the classification results obtained for a closed-world scenario. In case of DF, k-FP, and CUMUL, the best strategy for the attacker is to use all possible combinations of two merged subtraces for training (S_5) and, thus, cover all potential variations of testing subtraces. Although the adversary achieves a slightly higher recognition rate for k-NN when using S_4, the difference in the accuracies between S_4 and S_5 is neglectable and might be caused by the use of a specific dataset. Thus, we conclude that S_5 serves as our most effective training strategy in case of multiple malicious entry ORs. Notably, as n gets closer to m, the training dataset contains fewer combinations of subtraces. The attacker can fully reconstruct the page load, if $n = m$.

We further analyze the susceptibility of our defense against WFP attacks for an increasing number of malicious entry ORs by applying S_5 for training. Table 5.7 shows the classification results obtained for a closed-world scenario and malicious entry ORs varying from two to five. As expected, the more traffic is observed, the higher accuracy the adversary achieves. Nevertheless, the accuracies of all classifiers remain below 36% in case of two malicious entry ORs. Although the detection rate increases dramatically (especially for DF) when three or more entry ORs are compromised, the probability to select an excessive number of malicious entry ORs belonging to a single adversary is statistically not feasible in the real Tor network. Moreover, when the attacker controls more entry ORs, DF outperforms the best-performing classifier, k-FP. Finally,

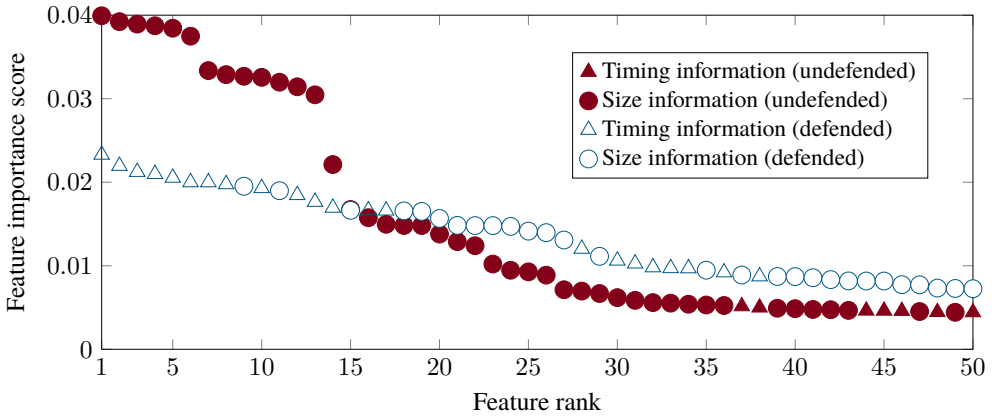


FIGURE 5.5: Feature importance score of the first 50 best-ranked features for defended and non-defended traces.

the accuracy achieved in case of five malicious entry ORs converges to that of non-defended traces.

5.6 Obfuscation of Most Informative Features

When designing a WFP countermeasure, it is important to ensure that this defense can obfuscate all discriminating features used by WFP classifiers. Otherwise, the adversary can discard non-relevant features and develop an optimized classifier that prioritizes other kind of features. Therefore, to better understand the influence of our defense on the features used in state-of-the-art WFP attacks, we computed the feature importance score of defended and non-defended traces by using the method presented in [21].

To do this, we categorized all generated features in two main groups: *(i)* features which are extracted based on information about *packet sizes* and *ordering*, and *(ii)* features that involve *timing* information. We restrict this analysis to k-FP, since it is the classifier that best performed against our traffic-splitting defense, and consequently has a chance to be redesigned.

Figure 5.5 shows the importance score of the first 50 best-ranked features for defended and non-defended traces. Non-defended traces are correctly detected since classifiers typically rely on features based on packet sizes and ordering. On the other side, the level of importance of size-related features reduces significantly when considering defended traces. Specifically, we obfuscate not only the size-related features but also the order of transmitting consecutive packets (due to the use of two splitting points for incoming and outgoing traffic and the weighted selection of an individual circuit for a batch of Tor cells). This, in turn, explains the success of our BWR splitting strategy over classifiers such as CUMUL and DF, which mainly rely on packet sizes and ordering to recognize websites. However, as our defense does not add any packet delays or dummy traffic, it cannot fully obfuscate features involving timing information and, thus, such features gain a higher importance score. This explains the better performance of k-FP, which also considers timing information. Nevertheless, those features are highly fluctuating depending

on the network conditions and, thus, require a much higher effort for an attacker to deploy a new effective attack.

To further confirm that, after applying our defense, timing-related features remain as the most representative ones for the classification, we repeated the k-FP attack but removing all timing-related features. In this experiment, the classifier reported an accuracy of 7.05%, which approximates to the accuracy achieved by other attacks. Thus, we confirm that these features are the main reason for k-FP to outperform other attacks. Looking specifically at the importance of only timing-related features, we found that the four most important features are: the 25, 50, 75, and 100^{th} percentile of the total page load time. When a trace is split into five subtraces, at least two out of five subtraces contain either the beginning or the end of the page load, which means that the 25 and 100^{th} percentile keep consistency among several subtraces of the same page. We believe these are the patterns distinguished by k-FP to achieve more than 15% accuracy in our experiments. However, those features are generally noisy, and consequently are less attractive when designing WFP classifiers.

To sum up, we have identified that after applying our defense, specific timing-related features remain useful for the k-FP classifier. However, since such patterns depend on the network conditions, they are not substantially reliable for mounting a WFP attack. Thus, we can affirm that our defense targets the most important traffic characteristics for WFP attacks — the size-related features. These, in turn, contribute to high-rate detection only if they contain a significant portion of the entire traffic trace (see Section 5.5.4).

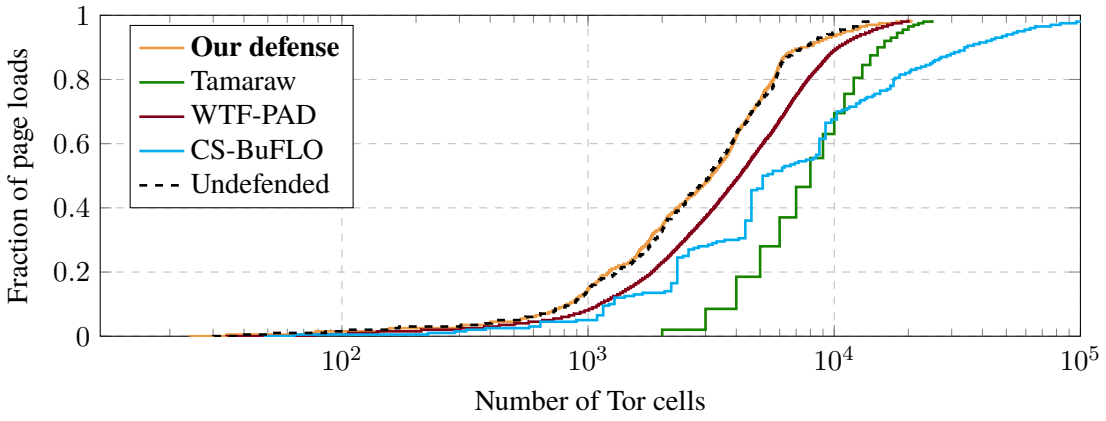
5.7 Comparison with Prior Defenses

We next compare our defense with the most popular previous WFP defenses CS-BuFLO [83], Tamaraw [84], and WTF-PAD [36]. We use the public implementations of each defense to replay the traces of our `ALEXA-NODEF` and obtain defended traces for our comparison.

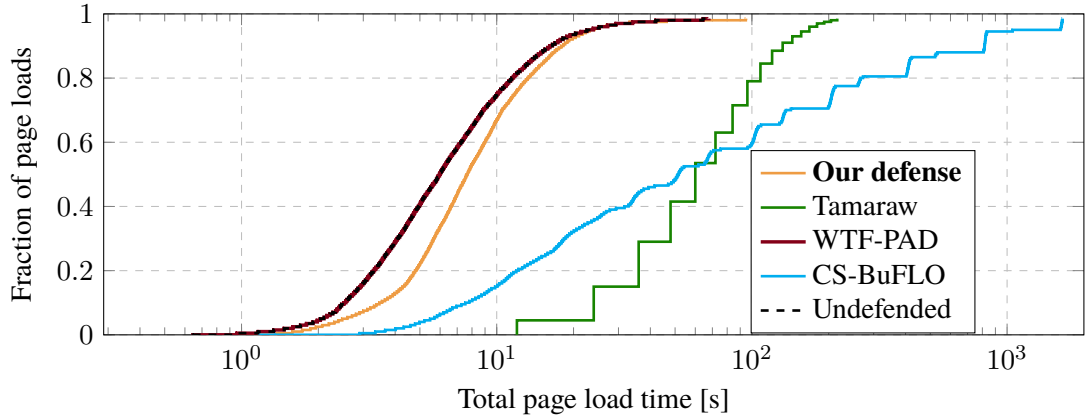
Figure 5.7 shows the classification results obtained in a closed-world scenario for a malicious entry OR. Our defense clearly outperforms CS-BuFLO and WTF-PAD. Although our defense and Tamaraw achieve similar accuracies in case of k-NN and CUMUL, Tamaraw is robuster against k-FP and DF compared to our defense. However, as we will show in the next section, this is achieved at the price of bandwidth and latency overheads that are several orders of magnitude higher than those of our defense (see Figure 5.6). We conclude that our defense achieves higher accuracy declines for all WFP attacks than the most former defenses.

5.8 Overhead and Performance Evaluation

We now evaluate the performance overhead introduced by our defense. As our defense does not introduce additional traffic nor intentional delays to obfuscate traffic patterns, the only overhead is caused by the multipath and reordering operations. As in previous works, we calculate bandwidth



(A) Bandwidth overhead.



(B) Latency overhead.

FIGURE 5.6: Cumulative distribution function (CDF) of bandwidth and latency overhead created by WFP defenses.

and latency overheads, and compare our countermeasure against previous defenses, and vanilla Tor (undefended). To do these calculations, we used our `ALEXA-NODEF`, `ALEXA-DEF`, and the undefended traces after applied the prior defenses as in the previous section.

As shown in Figure 5.6, our defense produces only a tiny bandwidth overhead in contrast to all former WFP defenses. In terms of latency, the created overhead is more conspicuous. However, it is still lower by several orders of magnitude compared to CS-BuFLO and Tamaraw. Although WTF-PAD does not introduce any time delays, it does not provide a sufficient level of security against WFP attacks as we showed previously. We conclude that our defense produce only a small overhead in terms of latency and a negligible one in terms of bandwidth consumption and outperforms all former WFP defenses.

Finally, we evaluate the load produced by our defense over the modified middle ORs. We want to assess whether the operations required by our defense (e.g., cell reordering and traffic merging) introduce a noticeable overhead on the processing of the middle OR. To this end, we set up a middle OR in the real Tor network and started hundreds of penetrators (clients downloading constantly data) to fetch data from a fixed web server and using, for each run, different entry ORs

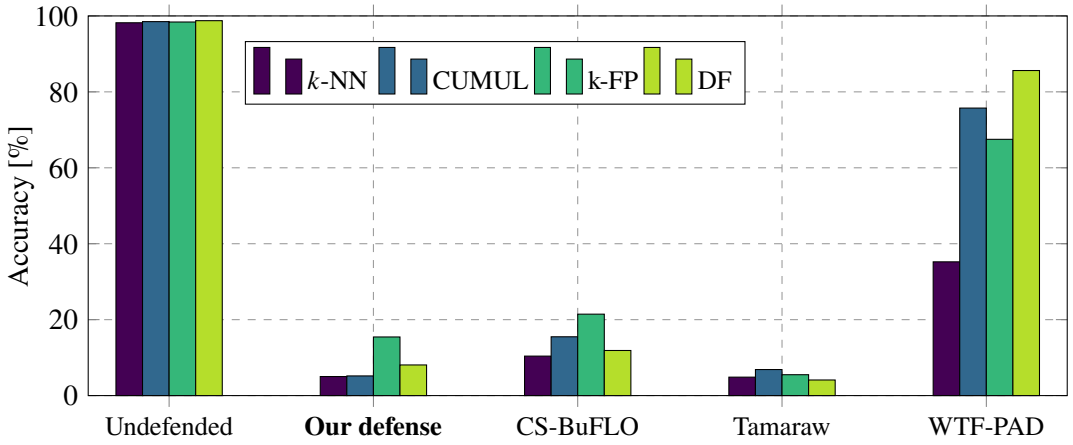


FIGURE 5.7: Accuracy of our and prior defenses against state-of-the-art WFP classifiers

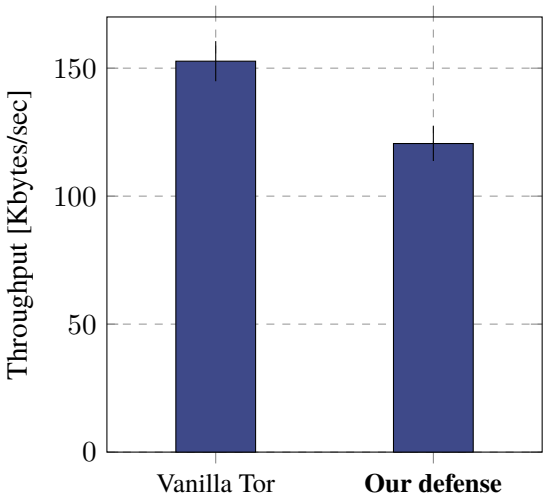


FIGURE 5.8: Throughput produced by our defense compared to Vanilla Tor

and exit ORs but the same middle ORs. We seek to overload the middle OR and compare its performance with and without the defense.

Figure 5.8 shows that the multipath-related operations indeed make the middle OR with our defense slower to forward traffic. Although we observe a decrease in performance of around 30%, it is important to highlight that this represent a worst-case scenario (i.e., hundreds of clients downloading data simultaneously over the same middle OR). In any case, we believe it is necessary to investigate further refinements in the splitting strategy (or even in other Tor components as in Section 4.7) to offer attractive performance together with the already demonstrated high security. In the following section, we explore some potential alternatives to reduce the load produced by our traffic-splitting defense.

5.9 Improving Performance in Our Traffic-Splitting Defense

When we designed and evaluated our strategies, the primary goal was to decrease accuracy of WFP attacks. However, this introduced some overhead due to operations such as buffering and reordering of cells traveling over different paths. We now analyze how these strategies can be refined in order to speed up the download times for Tor users.

In order to take advantage of multiple entry ORs to speed up data transfers, traffic should be forwarded using the path with the best characteristics. Since in our defense, the paths in the multipath structure share the middle and exit ORs, the only option to use faster nodes lies along the entry ORs. Thus, our splitting strategy should also consider the bandwidth properties of entry ORs and tend to distribute more traffic over relays with more capacity. It is important to mention, that the potential benefits of using properly faster entry ORs acquire more significance if the middle and exit OR are not the bottleneck of the overall communication capacity. Otherwise, we could at most, achieve similar performance levels as those provided by a single-path Tor circuit.

In our best splitting strategy (BWR), the amount of traffic exchanged over each entry is directly influenced by the weights obtained from the Dirichlet distribution. On each direction (outgoing or incoming), the weights generated by the strategy determine the probability to choose a certain entry OR to send the next batch of cells. Hence, an entry OR observes approximately that proportion of traffic with respect to the full page load. For this reason, modifications to this strategy for improving performance should be oriented to tailor the mechanisms to establish the weights for choosing an entry OR.

It is true that using a deterministic metric such as bandwidth to craft our strategy reduces the randomness that makes effective a splitting scheme. However, we aim to find a suitable trade-off where we can guarantee diversity among the protected traces, and still using optimally the multiple entry ORs. We next present modifications to our best splitting scheme — BWR — for utilizing more efficiently the available entry ORs.

5.9.1 Experimental Setup

For evaluating the variations to BWR, we rely on a similar experimental setup as the one considered in our previous performance evaluation. We started a middle OR in the real Tor network, which runs the original BWR strategy along with the alternatives here proposed. Then, we started hundreds of penetrators to overload the middle OR by downloading constantly data from a fixed server using different circuits (but using always the same middle OR). We recorded throughput on the client side and repeated the measurements more than 1,000 runs for each strategy.

Since we will rely, partially or fully, on the bandwidth of each entry OR for setting our strategy, it is important to ensure that we are using a metric that represents the real condition of an OR. Therefore, we use the bandwidth of the relay as reported in the consensus. This means that the obtained bandwidth value has been measured and approved by Tor authorities.

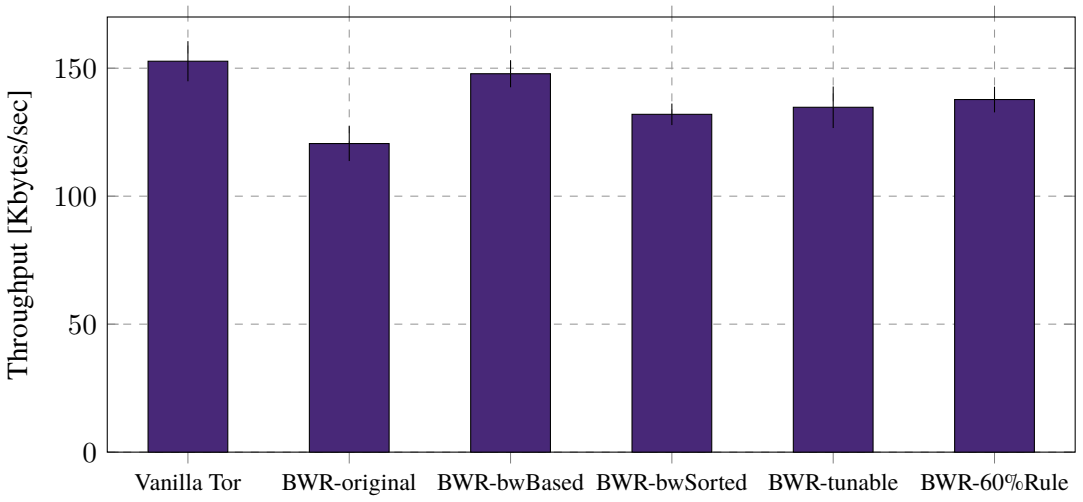


FIGURE 5.9: Throughput produced each of the splitting schemes variations compared to BWR and Vanilla Tor

5.9.2 Setting Weights Based on Bandwidth

We first analyze whether we can exclusively use bandwidth information for the splitting technique. The objective is to know what is the upper bound of achievable performance gains. In this scheme, we set the weights proportionally to each entry's bandwidth in relation to total bandwidth of all utilized entry ORs. This will force that entry ORs with more capacity forward more traffic. The advantage of this scheme is that we are optimally using each entry, and in the ideal case where middle and entry ORs are not the bottleneck, the overall capacity is the aggregate of each individual entry OR's bandwidth.

Nonetheless, this scheme has severe disadvantages against WFP. First, weights in both directions are the same, thus, similar traffic patterns travel forwards and backwards, which, in turn, could be detected as features by WFP classifiers. Second, high-bandwidth entry ORs will capture more traffic, in extreme cases the entire page load, making the WFP defense ineffective. Thus, we primarily evaluate the performance of this scheme as an upper-bound reference. Figure 5.9 shows that when faster entry ORs are preferred (scheme labeled as *BWR-bwBased*), throughput on the client side is visibly increased. It is still slightly lower than vanilla Tor since for some downloads, middle and exit ORs may have behaved as bottlenecks.

5.9.3 Sorting Weights Based on Bandwidth

In order to avoid that, in case of heavily unbalanced entry ORs, the node with higher capacity captures a large amount of traffic, we next present a technique where the weights values are not directly determined by the bandwidth. In this scheme, we use the bandwidths exclusively to sort the weights generated by the Dirichlet distribution. Thus, we still prefer to use a high-bandwidth OR for forwarding more traffic but the amount of traffic is still controlled by the WFP-effective

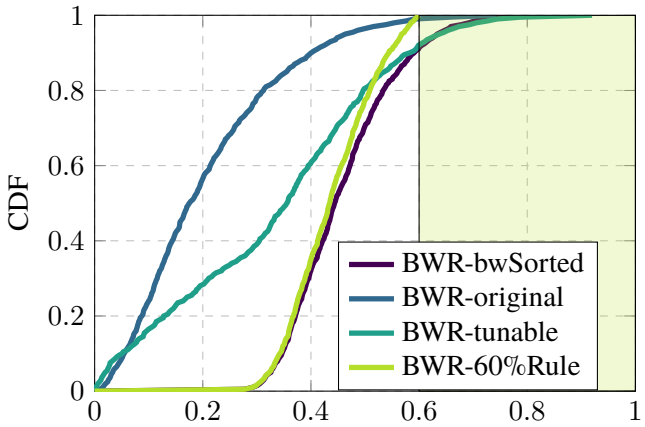


FIGURE 5.10: Cumulative distribution function of the highest weight assigned to the fastest entry OR.

splitting strategy. Evidently, performance improvements are less notorious, but WFP accuracy decrease should not changed as the strategy itself was not modified.

However, it is possible that specific entry ORs access to substantially more data than others. As we have learned from our analysis in Section 5.5.4, WFP classifiers might be effective while the attacker observes more than 60% of the traffic trace. Thus, we analyze how often that can happen in this strategy. For this, we generated one thousand weights vectors, and computed the cumulative distribution function of the highest weight — which approximates to the amount of traffic that the entry OR with the highest bandwidth would observe[1].

From Figure 5.10, we observe that with this scheme (labeled as *BWR-bwSorted*), the most performant entry ORs always observes at least 30% of the traffic. Although this is not particularly dangerous, we observe that in the other extreme, there is a visible amount of page loads where the attacker gets more than 60% of the total packets. Thus, this strategy gives him some more confidence to mount WFP attacks. Regarding throughput (see Figure 5.9), this scheme is less beneficial compared to BWR-bwBased. However, a subtle improvement to the original BWR design is noticeable. It is important to emphasize that WFP protection levels are not expected to change significantly, as we did not modify how weights are calculated, but only how they are assigned.

5.9.4 Tunable Weights based on Bandwidth

Even though the previous scheme seems promising to improve performance while maintaining the security levels, we now present a technique that aims to introduce more randomness to minimize the leakage of potentially large traces to high-bandwidth entry ORs (i.e., containing more than 60% of the page load). Our idea is to discourage entry ORs to offer more bandwidth with

[1]Recall that weights are independently calculated for each direction, thus, this analysis applies independently for incoming and outgoing traffic.

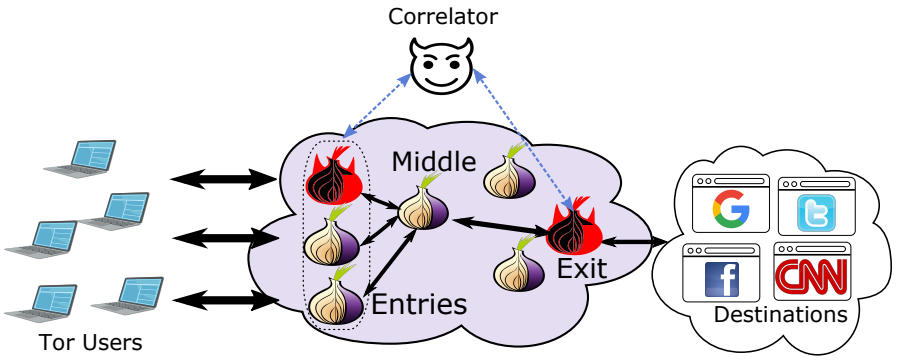


FIGURE 5.11: End-to-end correlation attack scenario.

the purpose of capturing more traffic to mount WFP attacks. To this end, we introduce a user-controlled parameter to decide how much of bandwidth or randomness to consider when setting the weights of BWR. The parameter α determines the probability to sort the weights, either according to the Dirichlet distribution or to the measured bandwidths. If $\alpha = 0$, weights are calculated and assigned as in the original BWR. If $\alpha = 1$, weights are sorted using the bandwidths. With this, WFP protection remains, as weights are still computed by the Dirichlet distribution, but there is space for the user to employ more often a faster entry OR. Figure 5.10 shows a compromise of this strategy (labeled as *BWR-tunable*) with $\alpha = 0.5$. We notice that the potentially dangerous limit of 60% is achieved similarly often as in BWR-bwSorted. However, slightly higher throughput can be achieved (see Figure 5.9), and malicious entry ORs can not assume that they always capture a noticeable minimum portion of traffic (i.e., they could statistically observe 0%).

5.9.5 Sorting Weights Bassed on the 60% Rule

To guarantee acceptable security against WFP attacks, we now consider the limit of 60% of a full page load as a hard threshold that must not be surpassed. Since none of the previous strategies solved this issue, we next propose a scheme that performs the sorting of weights similarly as BWR-bwSorted, but refuses to assign a weight higher than 0.6 to any entry OR. To this end, we repeat the Dirichlet computation if any of the weights reported a value higher than 0.6.

As expected, Figure 5.10 shows that the most performant entry OR never observes more than 60% of the page load traffic. Regarding performance, this method performs similarly as BWR-bwSorted. Thus, we consider this scheme as the simplest alternative to speed up data transfers for Tor clients without introducing any noticeable advantage for the WFP attacker.

5.10 Traffic Splitting against End-to-End Correlation

Tor users are susceptible to be deanonymized when an adversary has gained some control over both sides of the circuit, i.e., on the entry, and exit OR sides (see Figure 5.11). An adversary can passively observe and correlate traffic characteristics on both ends to disclose origin and

destination of the communication [31, 37, 38, 116]. Alternatively, he can intentionally delay or congest one side of the link [33], and observe the corresponding effects on the other side. If clients split traffic over multiple entry ORs, end-to-end attackers would access to a partial view of the traffic on each observed entry OR. While this would negligibly affect ASes or ISPs-level adversaries because they would still access to all client-to-entry links, other realistic attackers that only run ORs at both edges to capture clients, would have partial access of the entry-side traffic. Regarding gaining control over the entry ORs, the attacker is more likely to be part of the entries chosen by a multipathed client. Nevertheless, previous work has shown that the probability of picking a malicious entry OR is reasonably low even when the attacker has high bandwidth capabilities [10, 110]. For performing the attack, the adversary can — in his best-case scenario — control all m entries simultaneously, and the corresponding exit. In this case, the amount of traffic available to the adversary is not greater than in the single-entry scheme. Thus, employing more entries would not introduce new vulnerabilities. Moreover, considering a realistic scenario where an adversary controls — as it is currently possible in Tor — only *one* of the entry ORs of a circuit and the corresponding exit OR, we have identified that traffic-splitting may hamper the adversarial goal of finding visible patterns for correlation.

We have demonstrated that our traffic splitting defense is highly effective to counter WFP attacks performed by malicious entry ORs (a local adversary). We have validated our hypothesis that an entry-side adversary with a limited view of the traffic can identify less useful patterns for mounting this traffic-analysis attack. The intuition suggests that since an end-to-end attacker correlates from traffic characteristics present at the two edges (entry and exit OR), the disturbance via splitting of patterns on the entry side may difficult the attacker's goal to identify correlatable patterns present on the other end of the circuit.

In this section, we evaluate whether an end-to-end correlation attack is effective against traffic splitting. We consider the most up-to-date and powerful attack (DeepCorr), and compare its performance when a single malicious entry OR observes all the traffic versus one entry OR accessing to traffic split with the BWR strategy.

5.10.1 Experimental Setup

For this preliminary evaluation, we rely on the most recent state-of-the-art end-to-end correlation attack — DeepCorr. We chose this attack since it has shown to perform better than any other previous correlation attack. Hence, if our defense is able to mitigate this attack, we expect that previous approaches achieve even lower performance.

5.10.1.1 Traffic Traces Representation

DeepCorr is a DL-based attack that uses a convolutional neural network where each instance is represented as a pair of traffic flows, collected at entry and exit ORs. Each flow is, in turn, represented as two sequences, one per traffic direction, of packet sizes and their inter-arrival

times. Thus, each input to the model comprises a two-dimensional array with eight rows. Two classes are assigned for labeling the data. *Positive* when the pair of flows is originated from an entry and exit ORs of the same Tor circuit. And *negative* when the egress flow traversed an exit ORs from another circuit.

5.10.1.2 Dataset

We use the same dataset provided by authors of the original paper[2]. For capturing ingress traffic, a pool of clients visited the top 50,000 Alexa websites using 1,000 different circuits[3]. Approximately, each circuit was used to visit 50 websites. In turn, a proxy on the exit side was used to record egress traffic. We used the same amount of traces available in the public source code and dataset of this attack. Specifically, we evaluated the scenario that reported the best performance for the attack. I.e., all traffic traces collected at once without any gap in time. In total, we used traffic traces form 7,324 Tor connections. 6,000 were used for training, and 1,324 for testing (see next section for labeling details). If our defense can successfully counter the attack in this advantageous setting for the adversary, we can expect even higher effectiveness of our defense in more adverse conditions for the attacker.

Since we also evaluate the attack when our defense is applied, we used our traffic-splitting simulator (see Section 5.4.1) to replay the traces of the dataset, and so, composed a new defended set of traces *only* for the entry side. Particularly, we derived two defended datasets, using two entry ORs (i.e., with the BWR strategy and $m = 2$) and using five entry ORs (i.e., the best setting against WFP, BWR and $m = 5$ entry ORs).

5.10.1.3 Classifier and Evaluation Metrics

To allow for comparison, we delimit the classification similarly as in the original paper. The CNN is tuned according to the hyperparameters that reported the best results for the classification. For training, for each captured flow pair generated on the same Tor connection (i.e., from the positive class), 199 non-correlating traffic flows (i.e., instances belonging to the negative class) were created by pairing traffic of a certain entry OR with traffic from a random exit OR. For each instance, only the first 300 packets were considered to input the neural network.

In order to determine the effectiveness of the model, we calculated TPR and FPR at different threshold values for the testing instances. We computed the correlation value as the probability that a certain pair flow belongs to the positive class. A true positive (TP) is counted when the correlation of a flow pair originated in the same Tor connection is higher than the threshold, and a false positive (FP) is counted when this threshold is surpassed from a testing flow pair originated in non-related entry and exit ORs. From the total $N = 1{,}324$ Tor connections used in testing (recall these traffic traces belong to the positive class as were originated on a certain

[2] `https://github.com/SPIN-UMass/DeepCorr`

[3] The option to use persistent guards was deactivated to always get different entry ORs

Tor connection), we created negative-class flow pairs, in an all-against-all manner, to assess false positives. In total, $N \times (N-1)$ non-correlated instances were derived.

5.10.2 Evaluation of Traffic Splitting against End-to-end Correlation Attacks

We next evaluate how DeepCorr performs when traffic observed on the entry side is limited by our strategy. We first conducted the attack when users employ two entry ORs. We consider this scenario realistic, as it may be possible to connect to the different entries via unrelated networks (see Section 6.3.1), and extend the protection against global adversaries. Then, we mounted the attack over traces protected with BWR and using five entry ORs. This is the setting that best performs against WFP. Therefore, we expect a stronger defensive effect against DeepCorr.

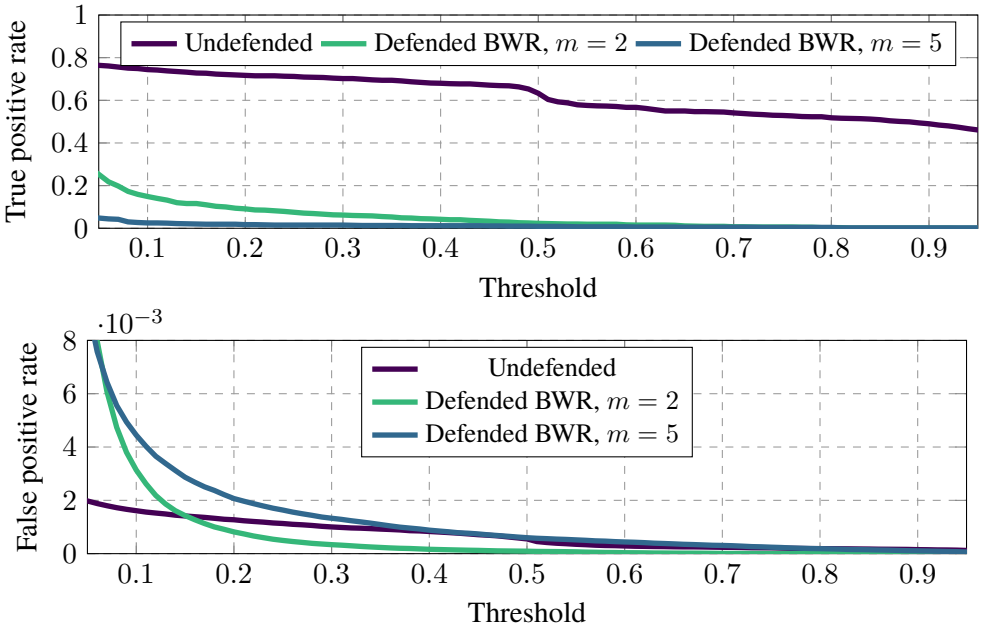


FIGURE 5.12: TPR and FPR for different threshold values

Figure 5.12 shows the TPR and FPR for various threshold values for the undefended and defended scenarios. For undefended traffic flows, we reproduce the results reported in the original DeepCorr paper. Overall, true positives are consistently higher and false positive are in a marginal level for the entire threshold spectrum. For instance, for a TPR close to 0.80, FPR is below 2×10^{-3}. Hence, we confirm high effectiveness of this attack to match correlations in real Tor connections. When correlating defended traffic flows, we observe that DeepCorr performance is drastically reduced. First, when two entry OR are used (BWR, $m = 2$), TPR never surpasses 0.25 for any threshold value. Additionally, false positives are alarming high for low threshold values, and when FPR lowers to desired levels, almost no traffic flow pair from a certain Tor connection can be correlated. Hence, splitting traffic into *only two* entry ORs is sufficient to reduce the attacker's ability and making almost unfeasible to confidently achieve real correlations. Second, using the best setting against WFP (BWR, $m = 5$) practically makes impossible

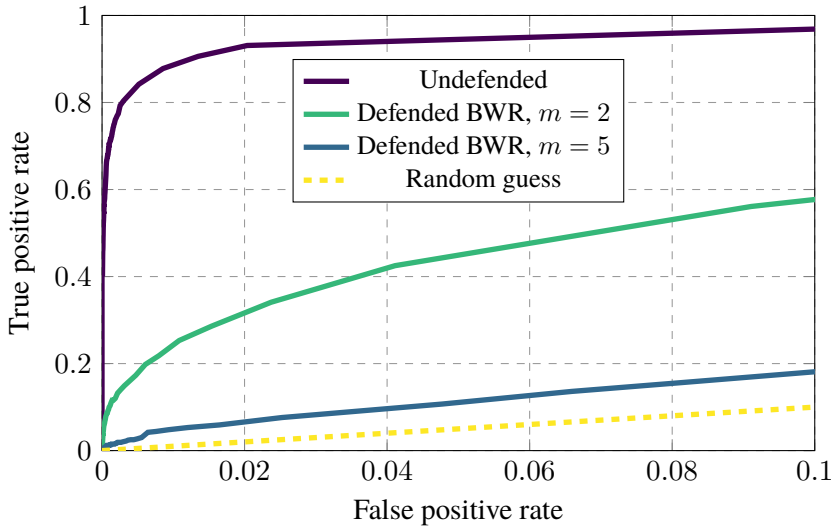


FIGURE 5.13: ROC of the DeepCorr attack in scenarios with traffic flows unprotected and protected with our traffic-splitting defense on the entry side.

to correlate traffic between an entry and exit OR. Calculated correlations are close to zero, and a TPR of only 0.1 is observable for a very large FPR value.

To better compare all the evaluated scenarios, we present ROC curves in Figure 5.13. Again, we reproduced the performance of DeepCorr against undefended traffic. For defended traffic flows, we observe that the attack is significantly less effective. When two entry ORs are used, TPR is halved for acceptable FPR values less than 0.04. This reduction is even more drastic when five entry ORs are used. Here, the model achieves only a slighter higher performance than the random guess. Thus, we confirm that our defense is also effective to mitigate end-to-end correlation performed by an adversary controlling one entry and one exit OR. Nonetheless, we believe these results are promising to be evaluated against more powerful adversaries. Particularly it would be interesting to use a multihomed client connecting to two entry ORs via unrelated connections. This would drastically complicate the attack for correlators such as ISPs, since we showed that this in-two splitting is enough to practically disable the attack.

5.11 Summary

In this chapter, we have presented an efficient and attractive defense against website fingerprinting attacks. The fundamental idea is to limit and destroy distinguishable traffic patterns observable to malicious entry ORs. We extensively explored several traffic splitting strategies in order to find the scheme that bests obfuscates useful features used in the most powerful WFP attacks. To this end, we implemented an accurate simulator that reproduces the behavior of the studied splitting strategies over traces collected in the Tor network. We determined that our strategy *batched weighted random* (BWR) using five entry ORs significantly decreases the accuracy of

all attacks to less than 16%. Our defense was implemented and ran successfully on the real-world Tor network, which makes it the most effective and deployable defense known as of today.

We evaluated several advantageous scenarios for advanced adversaries in which our defense still showed robustness and acceptable security levels. Since neither cover traffic nor artificial delays were introduced, our defense does not load the Tor network with unnecessary traffic nor impact noticeably the page load times experienced by the users. Still, we also presented some refinements to our best strategy (BWR) in order to enhance the client performance regarding web download time. We proposed to assign the weights calculated by BWR according to the bandwidth that each entry OR has. With this, faster download times are experienced and users do not notice difference between browsing with our defense and with regular Tor.

The last part of this chapter presented promising effects of our defense against end-to-end correlation attacks. When the attacker has a limited view of the ingress traffic, the observed correlation with another flow on the exit side does not allow the attacker to affirm with confidence that the traffic pair was originated on a certain Tor connection. Specifically, we showed that only using two malicious entry ORs with our traffic-splitting strategy already substantially reduces the effectiveness of the attack. It is important to mention that these effects are applicable to scenarios when the correlator has access to a limited number of entry ORs and the corresponding exit OR. We believe that these experiments motivate further investigations to clarify if the offered protection can be effective against more advanced adversarial scenarios.

What if before we start doing what we must, we started doing what we must have done?

Quino

6 Discussion

6.1 Introduction

In this chapter, we provide a motivating discussion about the limitations and further perspectives of each of our contributions. Specifically, we start analyzing the adoption road-map of our MPTCP-Tor transport design. We aim to clear up possible privacy concerns resulting from our proposal and provide preliminary insights about the presented countermeasures.

Then, we review the limitations of our WFP traffic-splitting defense. Particularly, we analyze the implications of using multiple entry ORs in the context of further attacks and sketch possible countermeasures.

Lastly, we survey other design components of Tor where multipath routing is applicable. We aim to provide future research directions that contribute to enhance the security and privacy properties of the entire Tor system.

6.2 Limitations and Perspectives of MPTCP-Tor

We have shown that MPTCP-Tor could be immediately adopted in Tor without requiring major changes on the system. Contrary to prior work, it is only required that some ORs install the MPTCP kernel and adopt our modifications in order to create multiple transport connections when connecting with other MPTCP-ready ORs. Furthermore, we also identified that the *connection profiling* anonymity leakage is solvable by breaking the direct relationships between

active circuits and connections. Although that would come at the cost of performance loss (because the number of subflows might not equal the number of circuits of all the time), still, we expect the degradation would be minimal while privacy will be boosted.

6.2.1 Limitations

Implementing MPTCP as a Tor transport protocol raises two main concerns. First, the connection profiling process needs to be addressed. We believe this can be solved if the opening and closing subflow actions are obfuscated with the insertion of delays. We conducted preliminary experiments where the MPTCP-API functions for opening or closing subflows reacted after a random time once a change on the number of active circuits was reported. We also tested ignoring certain changes on the number of active circuits in such a way that the number of subflows was not updated immediately after a change was detected. In both cases, we still got faster download times, but the subflows counting was more challenging. In fact, our dissecting tool reported several times an incorrect number of circuits. Nonetheless, we believe further experiments and evaluations are required to verify these findings and formalize an appropriate technique.

Second, we think that the transport design of Tor still needs to address the cross-circuit interference problem and its subsequent head-of-line blocking issue. With MPTCP, ORs still have a common shared socket buffer where arriving packets are queued before passing to the application. Thus, the risk that a packet of a certain user blocks others from concurrent connections remains. While it is true that MPTCP may in certain situations mitigate this effect, Tor still needs to handle congestion smarter and tailored for its kind of traffic. We see here the opportunity to integrate modern transport protocols such as QUIC [117], which also allows for multipath communication as MPTCP. Thus, it would possible to extend the performance benefits of multiple transport connections with specific mechanisms against cross-circuit interference.

6.2.2 Road-map for Adoption of MPTCP in Tor

We have identified the following challenges to allow for a smooth adoption of MPTCP as a Tor transport protocol. First, communication between the MPTCP kernel and Tor via an API is required. To the best of our knowledge, the API used in our evaluation (v0.94) is not updated with the latest version of the MPTCP kernel (v0.95). It seems that the continuous deployment of the transport protocol has not been enough well accompanied by a robust deployment of the API. This may be explained by the fact that MPTCP developers encourage to use the protocol in an agnostic manner without modifying the applications. Thus, complementary deployments have been sidelined.

Another important factor to highlight is that MPTCP was primarily designed for multihomed clients and to operate with a coupled congestion control algorithm. While we left the door open to ORs with multiple network interfaces, we strictly require an uncoupled congestion control algorithm. The latter requirement is contrary to one of the design goals of MPTCP.

Hence, we consider necessary that MPTCP developers redraw this design goal with more flexibility in order to avoid that future releases may not support the utilization of uncoupled congestion at all.

Although the number of non-Linux-based ORs is marginal within the Tor network, we believe it is still important to offer an MPTCP alternative for other operative systems. The main reason for this is that, if in the future, only Linux-based systems operate with MTPCP, while others only with TCP, adversaries could use this meta-information to craft attacks for deanonymizing clients. For instance, OS fingerprinting could become easier to perform.

Last but not least, integrating MPTCP in Tor faces similar challenging as adopting it for any other application [118]. Among others, this includes a coordinated action between OS developers, vendors, and end users, as well as the guarantee that the protocol will always work out-of-the-box.

To sum up, several challenges need to be addressed for a proper adoption of MTPCP in Tor. It is important to coordinate a continuous deployment of this protocol within the Tor community, and that MPTCP developers include Tor as a use case that deserves particular attention.

6.3 Limitations and Perspectives of our Traffic-Splitting Defense

We demonstrated that our defense is highly effective to counter today's WFP attacks. We evaluated, to the best possible extent, all adversarial strategies against defended traffic. We showed that our defense still performed notably in adverse situations (e.g., for multiple malicious entry ORs). We next detail some open issues for our defense, and how they can be addressed.

6.3.1 Protecting against ISP-level Adversaries

In contrast to former WFP defenses that aim at protecting against a malicious ISP *and* a malicious entry OR, our defense provides protection against malicious entry ORs only. To defend also against eavesdroppers on the link between a Tor user and an entry OR, i.e., the case of a malicious ISP, the user can utilize different access links (e.g., using distinct ISPs via DSL, Wi-Fi or cellular networks) to connect to the entry ORs, as proposed by Henri et al. [92]. However, it is worth noting that while it is hard to become a user's ISP, it is easier (and, thus, more dangerous for the user) to become an entry OR by launching multiple ORs in Tor.

6.3.2 Multiple Entry ORs and Path Compromise

From another perspective, the use of multiple entry ORs by our defense may increase the chances for an attacker to become one of these entry ORs. Thus, despite that our defense can deal even with several malicious entry ORs (see Section 5.5.5), we propose to apply the already existing guard concept of Tor [119] for the selection of entry ORs. This idea was already discussed and

suggested for adoption in the scope of multipath extensions of Tor for performance improvements [119]. According to the current guard specification, a Tor user selects a set of entry ORs for its guard list and uses it for a certain time period. If the user selects a malicious entry OR as its guard in Tor, it is completely exposed to the WFP attack at the entry OR. Contrary, in our defense, selecting one or even several malicious guards, as our evaluation shows, does not lead to successful mounting of WFP. Therefore, we believe that the adoption of our defense will not worsen the user's situation with respect to path compromises, though further research is needed to ultimately clarify this question.

6.3.3 Multiple Entry ORs vs. Congestion and DoS Attacks.

These attacks seek to flood with unnecessary traffic [32, 33, 69, 120] the targeted ORs, to reduce, and in some cases entirely disable, their capability to serve legitimate clients. The objective is to reduce the performance of the Tor network to such an extend that clients either refrain to use the network or are forced to repeatedly select new nodes until a malicious OR becomes part of the circuit. In cases where the target OR is at the entry position, utilizing multiple entries may mitigate the effects of a congestion or DoS attack, because the client still has alternative available paths to complete the communication and shift away the load injected on the target entry OR. Evidently, to achieve such mitigation, the traffic-splitting strategy of our defense should be modified to distribute cells according to the path's perceived performance. This highlights the necessity of designing a new strategy that considers the path's congestion while efficiently limiting WFP attacks. Fortunately, as we explained in Chapter 5, our multipath design offers the flexibility to easily introduce such new traffic splitting schemes.

6.3.4 Multiple Entry ORs vs. Guard Fingerprinting

We have analyzed this specific threat in Section 4.5.2. We concluded, that for a multipath client, the set of utilized entry ORs tends to be unique, and therefore can serve as a pseudonymous. This may open the door for WFP attacks from the middle OR because it could infer both to the destination and the user identifiable by the almost-unique set of entry ORs. As possible mitigation techniques, we could insert an extra node between the middle OR and the set of entry ORs, or translating the merging point to the exit OR. We believe more research is required to determine a proper solution and to analyze the real impact of this attack.

6.3.5 Selective Padding over Multiple Entry ORs

Another potential improvement of our defense to extend the protection against adversaries with a global view of the traffic on the entry side comprises the combination of traffic splitting and padding [121]. As we have demonstrated, when traffic of one or two malicious entry ORs is observed, WFP attacks do not succeed. Then, it would be possible to insert cover traffic on the remaining connections, and thus, achieve integral protection. Evidently, partial padding incurs

in significantly lower overheads than inserting dummy packets over the full traffic trace. Furthermore, we believe that since distinguishable patterns are already deficient, the padding technique could be reduced or simplified only to obfuscate those features that traffic splitting did not cover (i.e., timing-related features). We believe that, in the best case, this could be achieved even with trivial padding techniques that do not require a-priori knowledge about the websites to be protected. In a more conservative scenario, a global adversary (e.g., an ISP) would have less chances to mount a successful attack because he is aware that, on one hand, single connections are protected via splitting, and on the other hand, merged traffic includes padding packets.

6.4 Multipath Routing in other Aspects of Tor's Design Space

We believe have addressed the most advantageous design aspects of Tor for multipath routing — at circuit and transport level. However, we cannot discard that this paradigm can be beneficial in other aspects of Tor. We have identified there is opportunity to employ multipath for bridges. Particularly, some recent bridge protocols (e.g., Snowflake) rely on ephemeral nodes, thus, having alternative paths can help to enhance the reliability of the connection. The idea is to be able for fast recovering when one or more circuits fail. This principle can be also applied to increase the reliability of regular Tor circuits. It would be possible to replicate, fully or partially, some of the traffic along several paths, and so, provide more stable connections to clients. Furthermore, following a similar idea, blocking and censorship over certain ORs could also be evaded.

Just when we thought we had all the answers, suddenly, the questions changed.

Mario Benedetti

7 Conclusions

We have provided a set of novel multipath routing techniques to improve both performance and privacy of Tor users. We have identified the most advantageous design elements to incorporate the multipath routing paradigm, and then, conducted extensive evaluations to demonstrate the achieved benefits. In Chapter 4 we first introduced a comprehensive taxonomy for multipath onion routing-based anonymous communication systems. We can affirm from this survey, that onion routing-based approaches (e.g., Tor) can leverage multipath 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 multipath 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 multipath onion routing-based designs, but also for other types of anonymization systems.

For future multipath designs, greater performance improvements are expected if the current congestion estimation mechanisms can be refined to reflect the actual transport layer congestion into the multipath layer. Furthermore, other aspects such as anonymity and load balancing should be taken into consideration when designing the multipath circuit structure and scheduling mechanisms. We noticed 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.

From the resulting recommendations of our taxonomy, we designed MPTCP-Tor, an out-of-the-box and easy-to-deploy transport protocol for Tor. We showed that using MPTCP leads to noticeable performance improvements for Tor clients. In real-world settings, MPTCP can speed up web and bulk downloads on average 15% compared to TCP. We also analyzed the privacy

implications of integrating MPTCP as a Tor transport protocol. We showed that connection profiling might become a serious concern when adopting our design in Tor. In response to this, we have proposed several mitigation strategies, which we plan to address in detail in future work.

Later in Chapter 5, we presented a novel, lightweight, and high effective defense to counter WFP attacks. We drastically decreased the accuracy of the most powerful existing attacks. Our defense is based on user-controlled traffic splitting via multiple Tor paths. We also analyzed the effectiveness of different splitting schemes that can be integrated in our defense. We showed that our defense is able to reduce the accuracy from more than 98% to less than 16% for all state-of-the-art WFP attacks without adding any artificial delays or dummy traffic. Through an extensive evaluation, we identified system parameters and traffic-splitting strategies that effectively hamper WFP attacks. Besides the compatibility with the current Tor network, our defense does not insert noticeable bandwidth overhead and incurs only minimal latency overhead. Thus, it serves as suitable candidate for deployment in Tor.

Then, we proposed further refinements to our splitting strategy to better utilize the selected entry ORs. By also considering the bandwidth of the entry ORs, our traffic-splitting strategy offers similar performance to clients as if they were using the regular WFP-unprotected Tor.

In the last part of Chapter 5, we evaluated our traffic-splitting defense against end-to-end correlation performed by one malicious entry OR and the corresponding exit OR. The promising obtained results suggest that only by using two entry ORs, this attack becomes substantially less effective, and using five entry ORs, the attacker only gets slightly higher performance than the random guess.

Lastly, Chapter 6 revised the challenges of our contributions and analyzed how our WFP defense would behave against other threats. Additionally, further open issues that require more research were also detailed.

7.1 Answer to the Research Question

We started this dissertation with the following research question.

"How could multipath routing techniques improve the privacy and performance of Tor users?"

In order to provide multipath techniques that improve privacy an performance of Tor users, we first needed to conduct and extensive research about the design space, and the implications of incorporating multipath on each component. This resulted in several design recommendations that future systems should follow. Thus, we obtained a first partial response. *Yes*, indeed it is possible to provide improvements, but the specific techniques need to be developed. In other words, the *how* was is still unanswered. We continued our investigations to detail specific procedures that employ multipath to protect against WFP. Here, we found an answer for the *how*. To protect against WFP, a Tor client should use five entry ORs, and distribute traffic following the batched weighted random strategy. The last aspect to be answered — *how* multipath routing techniques

improve the privacy of Tor users? — is addressed with our MPTCP-Tor proposal. Tor users can experience faster download times in realistic conditions, if the ORs participating in the employed circuits communicate with each other using MPTCP and maintain as many open subflows as active circuits coexist. Thus, we believe have provided a comprehensive and compelling answer to the research question initially raised.

7.2 Future Research Perspectives

There are still several elements in Tor that can be improved for better privacy and performance of their users. We consider important to address cross-circuit interference, which is a significant problem in Tor, with mitigation techniques that often affect anonymity. It is relevant 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 believe that future research should point 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.

We also consider relevant to fill the gaps for adoption of our proposals in Tor. For our defense, an integration with multi-homing for extending the protection against malicious ISPs is necessary. Moreover, it is important to further investigate the impact of our defense against other threats to make sure that no new attack surfaces emerge.

For our transport design, we suggest that it must be incrementally integrated and evaluated in the real Tor network. Since our proposal is easy to deploy and works out-of-the-box, we believe it is ready for an *alpha* integration in Tor.

In general, we believe that the future of Tor is multipathed. The single-circuit overlay structure seems insufficient for an optimal usage of the existing resources. We argue that multiple paths offer more advantages than drawbacks. Overall, we think that attacks to the system will require higher efforts when the communication is moved via several paths. With this, we reinforce the idea that multipath requires further investigation. For instance, what is the upper limit of paths to use? Should Tor clients use a fixed or variable number of circuits for other purposes (e.g., censorship evasion)? Nonetheless, we believe have made notable scientific advances, which, we hope, provide significant insights for the research community in the light of further improvements for privacy enhancing technologies.

Abbreviations

k-NN k-nearest neighbors 23

AS autonomous system 26
AUC area under the curve 72

BWR batched weighted random 62

CBT circuit build time 11
CNN convolutional neural network 15
CR compromise rate 42
CV coefficient of variation 70

DC Dining cryptographers 8
DL deep learning 15
DoS denial of service 12
DT download time 37

FP false positive 85
FPR false positive rate 14

HoL head-of-line 17

ISP Internet service provider 9

KIST kernel-informed socket transport for Tor 11

LSTM long short-term memory 24

ML machine learning 14
MPTCP multipath TCP 46

OP onion proxy 9
OR onion relay 9

ROC receiver operating characteristic 72
RR round robin 29
RTT round trip time 35

SDAE stacked denoising autoencode 23
SDoS selective denial of service 43
SVM support vector machine 22

Tor the onion router 2
TP true positive 85
TPR true positive rate 14
TTFB time to first byte 37

TTLB time to last byte 51

WFP website fingerprinting 4
WR weighted random 62

Bibliography

[1] S. Varrette, P. Bouvry, H. Cartiaux, and F. Georgatos. Management of an academic hpc cluster: The ul experience. In *Proc. of the 2014 Intl. Conf. on High Performance Computing & Simulation (HPCS 2014)*, pages 959–967, Bologna, Italy, July 2014. IEEE.

[2] List of government mass surveillance projects. `https://en.wikipedia.org/wiki/List_of_government_mass_surveillance_projects`, 2020.

[3] Michael Carl Tschantz, Sadia Afroz, Anonymous, and Vern Paxson. SoK: Towards Grounding Censorship Circumvention in Empiricism. In *Symposium on Security and Privacy*, S&P, pages 914–933, San Jose, CA, USA, May 2016. IEEE.

[4] M. Li, A. Lukyanenko, Z. Ou, A. Ylä-Jääski, S. Tarkoma, M. Coudron, and S. Secci. Multipath transmission for the internet: A survey. volume 18, pages 2887–2925, 2016.

[5] Marjan Radi, Behnam Dezfouli, Kamalrulnizam Abu Bakar, and Malrey Lee. Multipath routing in wireless sensor networks: Survey and research challenges. volume 12, pages 650–85, 12 2012.

[6] J. Qadir, A. Ali, K. A. Yau, A. Sathiaseelan, and J. Crowcroft. Exploiting the power of multiplicity: A holistic survey of network-layer multipath. volume 17, pages 2176–2213, 2015.

[7] Karim Habak, Khaled A Harras, and Moustafa Youssef. Bandwidth aggregation techniques in heterogeneous multi-homed devices: A survey. volume 92, pages 168–188. Elsevier, 2015.

[8] Adi Shamir. How to share a secret. volume 22, pages 612–613. ACm New York, NY, USA, 1979.

[9] D. Das, S. Meiser, E. Mohammadi, and A. Kate. Anonymity trilemma: Strong anonymity, low bandwidth overhead, low latency - choose two. In *2018 IEEE Symposium on Security and Privacy (SP)*, pages 108–126, 2018.

[10] M. AlSabah, K. Bauer, T. Elahi, and I. Goldberg. The Path Less Travelled: Overcoming Tor's Bottlenecks with Traffic Splitting. volume 7981, pages 143–163. Springer, July 2013.

[11] L. Yang and F. Li. mTor: A Multipath Tor Routing Beyond Bandwidth Throttling. In *IEEE Conference on Communications and Network Security (CNS)*, Florence, Italy, September 2015. IEEE.

[12] M. AlSabah and I. Goldberg. Performance and Security Improvements for Tor: A Survey. *ACM Computing Surveys*, November 2016.

[13] Wladimir De la Cadena, A. Mitseva, J. Hiller, Jan Pennekamp, S. Reuter, T. Engel, K. Wehrle, and A. Panchenko. Trafficsliver: Fighting website fingerprinting attacks with traffic splitting. In *ACM CCS Conference*, November 2020.

[14] David L Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. volume 24, pages 84–90. ACM New York, NY, USA, 1981.

[15] George Danezis, Roger Dingledine, and Nick Mathewson. Mixminion: Design of a type iii anonymous remailer protocol. In *Proceedings of the 2003 IEEE Symposium on Security and Privacy*, SP '03, page 2, USA, 2003. IEEE Computer Society.

[16] David L Chaum. The dining cryptographers problem: Unconditional sender and recipient untraceability. Springer, 1988.

[17] R. Dingledine, N. Mathewson, and P. Syverson. Tor: The Second-generation Onion Router. In *13th conference on USENIX Security Symposium*, San Diego, CA, USA, August 2004. USENIX Association.

[18] M. AlSabah and Ian I. Goldberg. Performance and Security Improvements for Tor: A Survey. pages 32:1–32:36, New York, NY, USA, November 2016. ACM.

[19] A. Panchenko, F. Lanze, J. Pennekamp, T. Engel, A. Zinnen, M. Henze, and K. Wehrle. Website Fingerprinting at Internet Scale. In *23nd Annual Network and Distributed System Security Symposium (NDSS)*, San Diego, CA, USA, February 2016. Internet Society.

[20] Payap Sirinam, Mohsen Imani, Marc Juarez, and Matthew Wright. Deep Fingerprinting: Undermining Website Fingerprinting Defenses with Deep Learning. In *Proceedings of the 25th ACM SIGSAC Conference on Computer and Communications Security (CCS)*, Toronto, ON, Canada, 2018. ACM.

[21] Jamie Hayes and George Danezis. k-fingerprinting: A Robust Scalable Website Fingerprinting Technique. In *Proceedings of the 25th USENIX conference on Security Symposium*, Austin, TX, USA, 2016. USENIX Association.

[22] Tao Wang and Ian Goldberg. Improved Website Fingerprinting on Tor. In *Proceedings of the 12th ACM Workshop on Workshop on Privacy in the Electronic Society (WPES)*, Berlin, Germany, 2013. ACM.

[23] R. Dingledine and B. Mathewson. Tor Protocol Specification. `https://gitweb.torproject.org/torspec.git?a=blob_plain;hb=HEAD;f=tor-spec.txt`, 2018.

[24] Rob Jansen, Tavish Vaidya, and Micah Sherr. Point break: A study of bandwidth denial-of-service attacks against tor. In *Proceedings of the 28th USENIX conference on Security Symposium*, Santa Clara, CA, 2019. USENIX Association.

[25] R. Jansen, K. Bauer, N. Hopper, and R. Dingledine. Methodically Modeling the Tor Network. In *5th Workshop on Cyber Security Experimentation and Test (CSET)*, Bellevue, WA, USA, August 2012. USENIX.

[26] M. Imani, A. Barton, and M. Wright. Guard Sets in Tor using AS Relationships. In *Proceedings on Privacy Enhancing Technologies (PETS)*, Barcelona, Spain, July 2018. Springer.

[27] R. Jansen, J. Geddes, C. Wacek, M. Sherr, and P. F. Syverson. Never Been KIST: Tor's Congestion Management Blossoms with Kernel-Informed Socket Transport. In *Proceedings of the 23rd USENIX Security Symposium*, San Diego, CA, USA, August 2014. USENIX Association.

[28] Rob Jansen, Matthew Traudt, John Geddes, Chris Wacek, Micah Sherr, and Paul Syverson. Kist: Kernel-informed socket transport for tor. *Transactions on Privacy and Security*, 2018.

[29] Steven J. Murdoch and George Danezis. Low-cost traffic analysis of tor. In *Proceedings of the 2005 IEEE Symposium on Security and Privacy*, 2005.

[30] Aaron Johnson, Chris Wacek, Rob Jansen, Micah Sherr, and Paul Syverson. Users get routed: Traffic correlation on tor by realistic adversaries. In *Proceedings of the 2013 ACM SIGSAC Conference on Computer and Communications Security*, CCS '13, page 337–348, New York, NY, USA, 2013. Association for Computing Machinery. URL `https://doi.org/10.1145/2508859.2516651`.

[31] Milad Nasr, Alireza Bahramali, and Amir Houmansadr. Deepcorr: Strong flow correlation attacks on tor using deep learning. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, CCS '18, page 1962–1976, New York, NY, USA, 2018. Association for Computing Machinery. URL `https://doi.org/10.1145/3243734.3243824`.

[32] Nathan S. Evans, Roger Dingledine, and Christian Grothoff. A Practical Congestion Attack on Tor Using Long Paths. In *Proceedings of the 18th Conference on USENIX Security Symposium*, Montreal, Canada, 2009. USENIX Association.

[33] Rob Jansen, Florian Tschorsch, Aaron Johnson, and Björn Scheuermann. The Sniper Attack: Anonymously Deanonymizing and Disabling the Tor Network. In *Proceedings of the 21st Annual Network and Distributed System Security Symposium (NDSS)*, San Diego, CA, USA, 2014. Internet Society.

[34] Tobias Pulls and Rasmus Dahlberg. Website Fingerprinting with Website Oracles. In *Proceedings on Privacy Enhancing Technologies (PoPETS)*, Montreal, Canada, 2020. Sciendo.

[35] Sanjit Bhat, David Lu, Albert Kwon, and Srinivas Devadas. Var-CNN: A Data-Efficient Website Fingerprinting Attack Based on Deep Learning. In *Proceedings on Privacy Enhancing Technologies (PoPETS)*, Stockolm, Sweden, 2019. Sciendo.

[36] Marc Juarez, Mohsen Imani, Mike Perry, Claudia Diaz, and Matthew Wright. Toward an Efficient Website Fingerprinting Defense. In *Proceedings of the 21st European Symposium on Research in Computer Security (ESORICS)*, Heraklion, Greece, 2016. Springer.

[37] Ye Zhu, Xinwen Fu, Bryan Graham, Riccardo Bettati, and Wei Zhao. On flow correlation attacks and countermeasures in mix networks. In *Proceedings of the 4th International Conference on Privacy Enhancing Technologies*, PET'04, page 207–225, Berlin, Heidelberg, 2004. Springer-Verlag. URL `https://doi.org/10.1007/11423409_13`.

[38] Brian N. Levine, Michael K. Reiter, Chenxi Wang, and Matthew Wright. Timing attacks in low-latency mix systems. In Ari Juels, editor, *Financial Cryptography*, pages 251–265, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.

[39] Milad Nasr, Amir Houmansadr, and Arya Mazumdar. Compressive traffic analysis: A new paradigm for scalable traffic analysis. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, CCS '17, page 2053–2069, New York, NY, USA, 2017. Association for Computing Machinery. URL `https://doi.org/10.1145/3133956.3134074`.

[40] Yixin Sun, Anne Edmundson, Laurent Vanbever, Oscar Li, Jennifer Rexford, Mung Chiang, and Prateek Mittal. RAPTOR: Routing Attacks on Privacy in Tor. In *Proceedings of the 24th USENIX conference on Security Symposium*, 2015.

[41] T. Wang, K. Bauer, C. Forero, and I. Goldberg. Congestion-Aware Path Selection for Tor. In *Financial Cryptography and Data Security (FC)*, pages 98–113, Kralendijk, Bonaire, March 2012. Springer.

[42] M. Akhoondi, C. Yu, and H.V. Madhyastha. LASTor: A Low-Latency AS-Aware Tor Client. In *Symposium on Security and Privacy (S&P)*, San Francisco, CA, USA, May 2012. IEEE.

[43] R. Snader and N. Borisov. Improving Security and Performance in the Tor Network through Tunable Path Selection. *IEEE Transactions on Dependable and Secure Computing*, September 2011.

[44] R. Snader and N. Borisov. 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.

[45] R. Dingledine and S. Murdoch. 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.

[46] S. Murdoch. Comparison of Tor Datagram Designs. Technical report, 2011.

[47] J. Reardon and I. Goldberg. Improving Tor Using a TCP-over-DTLS Tunnel. In *18th Conference on USENIX Security Symposium*, pages 119–134, Montreal, Canada, August 2009. USENIX Association.

[48] C. Tang and I. Goldberg. An Improved Algorithm for Tor Circuit Scheduling. In *17th ACM Conference on Computer and Communications Security*, pages 329–339, Chicago, Illinois, USA, October 2010. ACM.

[49] F. Tschorsch and B. Scheuermann. Tor is Unfair – And What to Do about It. In *36th Conference on Local Computer Networks*, Bonn, Germany, October 2011. IEEE.

[50] M. AlSabah, K. Bauer, I. Goldberg, D. Grunwald, D. McCoy, S. Savage, and G.M. Voelker. DefenestraTor: Throwing Out Windows in Tor. *Privacy Enhancing Technologies (PETS)*, 6794:134–154, July 2011.

[51] K. Loesing, S. Murdoch, and R. Jansen. Evaluation of a libutp-based Tor datagram implementation. Technical report, Tech. Rep. 2013-10-001, The Tor Project, 2013.

[52] C. Viecco. UDP-OR: A Fair Onion Transport Design. In *1st Workshop on Hot Topics in Privacy Enhancing Technologies (HotPETS)*, Leuven, Belgium, July 2008.

[53] John Geddes, Rob Jansen, and Nicholas Hopper. IMUX: Managing Tor Connections from Two to Infinity, and Beyond. In *Proceedings of the 13th Workshop on Privacy in the Electronic Society (WPES)*. ACM, 2014.

[54] M. AlSabah and I. Goldberg. PCTCP: Per-circuit TCP-over-IPsec Transport for Anonymous Communication Overlay Networks. In *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security*, Berlin, Germany, November 2013. ACM.

[55] Deepika Gopal and Nadia Heninger. Torchestra: Reducing Interactive Traffic Delays over Tor. In *Proceedings of the 2012 ACM Workshop on Privacy in the Electronic Society WPES*. ACM, 2012.

[56] Michael F. Nowlan, David Isaac Wolinsky, and Bryan Ford. Reducing latency in tor circuits with unordered delivery. In *Presented as part of the 3rd USENIX Workshop on Free and Open Communications on the Internet*, 2013.

[57] F. Tschorsch and B. Scheuermann. Mind the Gap: Towards a Backpressure-based Transport Protocol for the Tor Network. In *13th Usenix Conference on Networked Systems Design and Implementation (NSDI)*, pages 597–610, Santa Clara, CA, USA, March 2016. USENIX Association.

[58] A. Serjantov and S. Murdoch. Message Splitting Against the Partial Adversary. In *Privacy Enhancing Technologies (PETS)*, pages 26–39, Cavtav, Croatia, June 2005. Springer.

[59] J. Feigenbaum, A. Johnson, and P. Syverson. Preventing Active Timing Attacks in Low-latency Anonymous Communication. In *Privacy Enhancing Technologies (PETS)*, pages 166–183, Berlin, Germany, July 2010. Springer.

[60] A. Engelmann and A. Jukan. Defying censorship with multi-circuit tor and linear network coding. In *2019 12th CMI Conference on Cybersecurity and Privacy (CMI)*, pages 1–6, 2019.

[61] S. Katti, J. Cohen, and D. Katabi. Information Slicing: Anonymity Using Unreliable Overlays. In *4th Usenix Conference on Networked Systems Design and Implementation (NSDI)*, Cambridge, MA, USA, April 2007. USENIX Association.

[62] R. Snader. *Path Selection for Performance- and Security-Improved Onion Routing*. PhD thesis, University of Illinois at Urbana-Champaign, 2010.

[63] H. Karaoglu, M. Akgun, M. Gunes, and M. Yuksel. Multi Path Considerations for Anonymized Routing: Challenges and Opportunities. In *5th International Conference on New Technologies, Mobility and Security (NTMS)*, pages 1–5. IEEE, May 2012.

[64] M. Handley, O. Bonaventure, C. Raiciu, and A. Ford. TCP Extensions for Multipath Operation with Multiple Addresses. RFC 1654, 1995.

[65] O. Landsiedel, A. Pimenidis, K. Wehrle, H. Niedermayer, and G. Carle. Dynamic Multi-path Onion Routing in Anonymous Peer-To-Peer Overlay Networks. In *50th Annual IEEE Global Telecommunications Conference (GLOBECOM)*, pages 64–69, Washington, DC, USA, November 2007. IEEE.

[66] Kevin Bauer, Damon McCoy, Dirk Grunwald, Tadayoshi Kohno, and Douglas Sicker. Low-resource routing attacks against tor. In *Proceedings of the 2007 ACM Workshop on Privacy in Electronic Society WPES*, 2007.

[67] Nikita Borisov, George Danezis, Prateek Mittal, and Parisa Tabriz. Denial of service or denial of security? In *Proceedings of the 14th ACM conference on Computer and communications security*, 2007.

[68] Henry Tan, Micah Sherr, and Wenchao Zhou. Data-plane defenses against routing attacks on tor. volume 2016, pages 276 – 293, Berlin, 2016. Sciendo. URL `https://content.sciendo.com/view/journals/popets/2016/4/article-p276.xml`.

[69] Marco Valerio Barbera, Vasileios P. Kemerlis, Vasilis Pappas, and Angelos D. Keromytis. CellFlood: Attacking Tor Onion Routers on the Cheap. In *Proceedings of the 18th European Symposium on Research in Computer Security (ESORICS)*, Egham, UK, 2013. Springer.

[70] Andriy Panchenko, Lukas Niessen, Andreas Zinnen, and Thomas Engel. Website Fingerprinting in Onion Routing Based Anonymization Networks. In *Proceedings of the 10th Annual ACM Workshop on Privacy in the Electronic Society (WPES)*. ACM, 2011.

[71] Tao Wang and Ian Goldberg. On Realistically Attacking Tor with Website Fingerprinting. In *Proceedings on Privacy Enhancing Technologies (PoPETs)*, Philadelphia, PA, USA, 2015.

[72] Tao Wang, Xiang Cai, Rishab Nithyanand, Rob Johnson, and Ian Goldberg. Effective Attacks and Provable Defenses for Website Fingerprinting. In *Proceedings of the 24th USENIX conference on Security Symposium*, San Diego, CA, USA, 2014. USENIX Association.

[73] Payap Sirinam, Nate Mathews, Mohammad Saidur Rahman, and Matthew Wright. Triplet Fingerprinting: More Practical and Portable Website Fingerprinting with N-Shot Learning. In *Proceedings of the 26th ACM SIGSAC Conference on Computer and Communications Security (CCS)*, London, United Kingdom, 2019. ACM.

[74] Dominik Herrmann, Rolf Wendolsky, and Hannes Federrath. Website Fingerprinting: Attacking Popular Privacy Enhancing Technologies with the Multinomial Naïve-Bayes Classifier. In *Proceedings of the 2009 ACM Workshop on Cloud Computing Security*, Chicago, IL, USA, 2009. ACM.

[75] Andriy Panchenko, Fabian Lanze, Andreas Zinnen, Martin Henze, Jan Pennekamp, Klaus Wehrle, and Thomas Engel. Website Fingerprinting at Internet Scale. In *23rd Annual Network and Distributed System Security Symposium*, NDSS, San Diego, CA, USA, Februar 2016. Internet Society.

[76] Marc Juarez, Sadia Afroz, Gunes Acar, Claudia Diaz, and Rachel Greenstadt. A Critical Evaluation of Website Fingerprinting Attacks. In *Proceedings of the 21st ACM SIGSAC Conference on Computer and Communications Security (CCS)*, Scottsdale, AZ, USA, 2014. ACM.

[77] Xiang Cai, Rishab Nithyanand, Tao Wang, Rob Johnson, and Ian Goldberg. A Systematic Approach to Developing and Evaluating Website Fingerprinting Defenses. In *Proceedings of the 21st ACM SIGSAC Conference on Computer and Communications Security (CCS)*, Scottsdale, AZ, USA, 2014. ACM.

[78] Kevin Dyer, Scott Coull, Thomas Ristenpart, and Thomas Shrimpton. Peek-a-Boo, I Still See You: Why Efficient Traffic Analysis Countermeasures Fail. In *Proceedings of the 33rd IEEE Symposium on Security and Privacy (S&P)*, San Francisco, CA, USA, 2012. IEEE.

[79] Kota Abe and Shigeki Goto. Fingerprinting Attack on Tor Anonymity using Deep Learning. In *Proceedings of the Asia Pacific Advanced Network Workshop*, APAN, 2016.

[80] Vera Rimmer, Davy Preuveneers, Marc Juárez, Tom van Goethem, and Wouter Joosen. Automated Website Fingerprinting through Deep Learning. In *Proceedings of the 25th Network and Distributed System Security Symposium (NDSS)*, San Diego, CA, USA, 2018. Internet Society.

[81] Charles Wright, Scott Coull, and Fabian Monrose. Traffic Morphing: An Efficient Defense Against Statistical Traffic Analysis. In *Proceedings of the 16th Anual Network and Distributed System Security Symposium (NDSS)*, San Diego, CA, USA, 2009. Internet Society.

[82] Marc Liberatore and Brian Levine. Inferring the Source of Encrypted HTTP Connections. In *Proceedings of the 13th ACM Conference on Computer and Communications Security (CCS)*, Alexandria, VA, USA, 2006. ACM.

[83] Xiang Cai, Rishab Nithyanand, and Rob Johnson. CS-BuFLO: A Congestion Sensitive Website Fingerprinting Defense. In *Proceedings of the 13th Workshop on Privacy in the Electronic Society (WPES)*, Scottsdale, AZ, USA, 2014. ACM.

[84] Xiang Cai, Rishab Nithyanand, Tao Wang, Rob Johnson, and Ian Goldberg. A Systematic Approach to Developing and Evaluating Website Fingerprinting Defenses. In *Proceedings of the 21st ACM SIGSAC Conference on Computer and Communications Security (CCS)*, Scottsdale, Arizona, USA, 2014. ACM.

[85] Rishab Nithyanand, Xiang Cai, and Rob Johnson. Glove: A Bespoke Website Fingerprinting Defense. In *Proceedings of the 13th Workshop on Privacy in the Electronic Society (WPES)*, Scottsdale, Arizona, USA, 2014. ACM.

[86] Mike Perry. Experimental Defense for Website Traffic Fingerprinting. `https://blog.torproject.org/experimental-defense-website-traffic-fingerprinting`, 2011. (Accessed: January 2020).

[87] Xiapu Luo, Peng Zhou, Edmond W. W. Chan, Wenke Lee, Rocky K. C. Chang, and Roberto Perdisci. HTTPOS: Sealing information leaks with browser-side obfuscation of encrypted flows. In *Proceedings of the 18th Anual Network and Distributed System Security Symposium (NDSS)*, San Diego, CA, USA, 2011. Internet Society.

[88] Giovanni Cherubin, Jamie Hayes, and Marc Juarez. Website Fingerprinting Defenses at the Application Layer. In *17th Privacy Enhancing Technologies Symposium*, PETS, pages 186–203, Minneapolis, USA, July 2017. DE GRUYTER.

[89] Ian Goldberg. Network-Based Website Fingerprinting. `https://tools.ietf.org/html/draft-wood-privsec-wfattacks-00`, 2019. (Accessed: August 2019).

[90] Nick Mathewson. New Release: Tor 0.4.0.5. `https://blog.torproject.org/new-release-tor-0405`, 2019. (Accessed: January 2020).

[91] Tao Wang and Ian Goldberg. Walkie-Talkie: An Efficient Defense Against Passive Website Fingerprinting Attacks. In *Proceedings of the 26th USENIX conference on Security Symposium*, Vancouver, BC, Canada, 2017. USENIX Association.

[92] Sébastien Henri, Ginés García-Avilés, Pablo Serrano, Albert Banchs, and Patrick Thiran. Protecting against Website Fingerprinting with Multihoming. In *Proceedings on Privacy Enhancing Technologies (PoPETS)*, Montreal, Canada, 2020. Sciendo.

[93] F. Tschorsch and B. Scheurmann. 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*, New York, NY, USA, June 2012. ACM.

[94] O. Titz. Why TCP Over TCP Is A Bad Idea. URL `http://sites.inka.de/bigred/devel/tcp-tcp.html`. `http://sites.inka.de/bigred/devel/tcp-tcp.html`, accessed 2018-11-16.

[95] The Tor Relay Guide. `https://trac.torproject.org/projects/tor/wiki/TorRelayGuide`, 2018.

[96] F. Shirazi, M. Goehring, and C. Diaz. Tor Experimentation Tools. In *Security and Privacy Workshops (SPW)*, pages 206–213, San Jose, CA, USA, July 2015. IEEE.

[97] R. Jansen and N. Hopper. 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.

[98] A. Das and N. Borisov. Securing Anonymous Communication Channels under the Selective DoS Attack. In *Proceedings of Financial Cryptography and Data Security*, pages 362–370, Okinawa, Japan, April 2013. Springer.

[99] J. Hayes and G. Danezis. Guard Sets for Onion Routing. In *Proceedings on Privacy Enhancing Technologies (PETS)*, Philadephia, PA, USA, June 2015. Springer.

[100] J. Hayes. Traffic Confirmation Attacks Despite Noise. In *Understanding and Enhancing Online Privacy Satellite Workshop of NDSS*, San Diego, CA, USA, April 2016.

[101] J. Hayes and G. Danezis. k-fingerprinting: A Robust Scalable Website Fingerprinting Technique. In *25th USENIX Conference on Security Symposium*, Austin, TX, USA, August 2016. USENIX Association.

[102] T. Wang, X. Cai, R. Nithyanand, R. Johnson, and I. Goldberg. Effective Attacks and Provable Defenses for Website Fingerprinting. In *Proceedings of the 23rd USENIX Security Symposium*, San Diego, CA, USA, August 2014. USENIX Association.

[103] V. Rimmer, D. Preuveneers, M. Juarez, T. Van Goethem, and W. Joosen. Automated Website Fingerprinting through Deep Learning. In *25nd Annual Network and Distributed System Security Symposium (NDSS)*, San Diego, CA, USA, February 2018. Internet Society.

[104] Linux 5.6 is released. `https://linuxreviews.org/Linux_5.6_Is_Released`, 2020.

[105] Thomas Dreibholz, Martin Becke, Erwin P Rathgeb, and Michael Tuxen. On the use of concurrent multipath transfer over asymmetric paths. In *IEEE Global Telecommunications Conference GLOBECOM 2010*. IEEE, 2010.

[106] C. Paasch and S. Barre et al. Multipath TCP in the Linux Kernel. `https://www.multipath-tcp.org`, 2020.

[107] B. Hesmans. MPTCP API for Linux v0.94. `https://github.com/bhesmans/mptcp`, 2018.

[108] Tor Metrics. `https://metrics.torproject.org/`, 2020.

[109] Payap Sirinam, Mohsen Imani, Marc Juarez, and Matthew Wright. Deep fingerprinting: Undermining website fingerprinting defenses with deep learning. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, CCS '18, pages 1928–1943, New York, NY, USA, 2018. ACM.

[110] Wladimir De la Cadena, Daniel Kaiser, Asya Mitseva, Andriy Panchenko, and Thomas Engel. Analysis of Multi-path Onion Routing-Based Anonymization Networks. In *Proceedings of the 33rd Anual IFIP Conference on Data and Applications Security and Privacy*, DBSec, Charleston, SC, USA, 2019. Springer.

[111] Andriy Panchenko and Johannes Renner. Path Selection Metrics for Performance-Improved Onion Routing. In *Proceedings of the 9th IEEE/IPSJ Symposium on Applications and the Internet*, SAINT, Seattle, Washington, USA, July 2009. IEEE.

[112] `https://github.com/TrafficSliver`, 2020.

[113] Alexa. Alexa Tor 100 most popular websites. `https://www.alexa.com/`, 2020. (Accessed: September 2018).

[114] Roger Dingledine and Nick Mathewson. Tor Protocol Specification. `https://gitweb.torproject.org/torspec.git/tree/tor-spec.txt`, 2019. (Accessed: January 2020).

[115] Junhua Yan and Jasleen Kaur. Feature Selection for Website Fingerprinting. In *18th Privacy Enhancing Technologies Symposium*, PETS, pages 200–219, Barcelona, Spain, July 2018. DE GRUYTER.

[116] Aaron Johnson, Chris Wacek, Rob Jansen, Micah Sherr, and Paul Syverson. Get Users Routed: Traffic Correlation on Tor by Realistic Adversaries. In *Proceedings of the 20th ACM SIGSAC Conference on Computer and Communications Security (CCS)*, Berlin, Germany, 2013. ACM.

[117] Quic, a multiplexed stream transport over udp. `https://www.chromium.org/quic`, 2020.

[118] Alexandros Kostopoulos, Henna Warma, Tapio Levä, Bernd Heinrich, Alan Ford, and Lars Eggert. Towards multipath tcp adoption: challenges and opportunities. In *6th EURO-NGI Conference on Next Generation Internet*, pages 1–8. IEEE, 2010.

[119] Isis Lovecruft, George Kadianakis, Ola Bini, and Nick Mathewson. Tor Guard Specification. `https://gitweb.torproject.org/torspec.git/tree/guard-spec.txt`, 2019. (Accessed: January 2020).

[120] Vasilis Pappas, Elias Athanasopoulos, Sotiris Ioannidis, and Evangelos P. Markatos. Compromising Anonymity Using Packet Spinning. In *Proceedings of the 11th International Conference Information Security (ISC)*, Taipei, Taiwan, 2008. Springer.

[121] Marc Juarez, Mohsen Imani, Mike Perry, Claudia Diaz, and Matthew Wright. Toward an efficient website fingerprinting defense. In *ESORICS*, pages 27–46, 09 2016.