NOVA SCHOOL OF SCIENCE & TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE

GONÇALO GOMES RODRIGUES

BSc Degree in Informatics Engineering

THE UNOBSERVABILITY SCHEDULER

USE OF DIFFERENTIAL PRIVACY FOR UNOBSERVABLE PRIVACY-PRESERVED COMMUNICATION

Dissertation Plan
MASTER IN COMPUTER SCIENCE AND ENGINEERING

NOVA University Lisbon
February, 2024

NOVA SCHOOL OF
SCIENCE & TECHNOLOGY

DEPARTMENT OF
COMPUTER SCIENCE

THE UNOBSERVABILITY SCHEDULER

USE OF DIFFERENTIAL PRIVACY FOR UNOBSERVABLE PRIVACY-PRESERVED COMMUNICATION

GONÇALO GOMES RODRIGUES

BSc Degree in Informatics Engineering

Adviser: Henrique João
Associate Professor, NOVA University Lisbon

Dissertation Plan
MASTER IN COMPUTER SCIENCE AND ENGINEERING

NOVA University Lisbon
February, 2024

Abstract

The internet has become an indispensable tool for accessing information and exercising freedom of speech, a fundamental right that empowers individuals worldwide. However, authoritarian regimes and government agencies tend to suppress this right through surveillance and censorship activities. These efforts have become increasingly sophisticated, employing techniques such as traffic analysis attacks and advanced traffic correlation methods to monitor and undermine user anonymity.

In response to the growing demand for secure and private communication, Anonymity Networks like Tor have emerged as essential tools for safeguarding online end-to-end communication and fighting censorship, which millions of people rely on daily. However, recent research demonstrated that potential de-anonymization attacks can be performed by powerful state-level adversaries by traffic correlation and fingerprinting using advanced deep machine learning techniques.

This dissertation addresses these challenges by strengthening Tor Network's defenses against the above-mentioned attacks. The main goal is to develop an innovative solution targeted to improve Tor relay nodes leveraging the existing software architecture by introducing formal privacy guarantees based on differentially private circuits. We expect that our proposed solution will maintain practical levels of throughput and latency for end-to-end communication, ensuring compatibility with real-world usage scenarios while providing users with adaptable and more robust anonymity protection.

As stated above, the core of the proposed solution is the integration of Differential Privacy principles to dynamically introduce carefully bounded random noise into traffic flows, mitigating correlation attacks and empowering a better solution resilient against fingerprinting. To the best of our knowledge, the expected dissertation contributions can represent a pioneering effort in incorporating Differential Privacy into the software architecture of Tor relay nodes. Moreover, the use of Differential Privacy will allow for a formally proven, scalable, and effective mechanism that bolsters online anonymity, resistant against state-of-the-art internet censorship techniques and upholding the fundamental right to free expression in an increasingly monitored digital world.

Keywords: Anonymity Networks, Differential Privacy, End-to-End Privacy Enhanced Guarantees, Privacy-Preserving Communication, Traffic Analysis Attacks, Traffic Correlation

Resumo

A Internet tornou-se uma ferramenta indispensável para aceder à informação e exercer a liberdade de expressão, um direito fundamental que confere poder aos indivíduos em todo o mundo. No entanto, os regimes autoritários e as entidades governamentais tendem a restringir este direito através de práticas de vigilância e censura. Estes esforços têm-se tornado cada vez mais sofisticados, recorrendo a técnicas como a análise de tráfego e métodos avançados de correlação de dados para monitorizar e comprometer o anonimato dos utilizadores.

Em resposta à crescente necessidade de comunicações seguras e privadas, surgiram redes de anonimato como o Tor, que desempenham um papel essencial na proteção das comunicações online e na resistência à censura, das quais milhões de pessoas dependem diariamente. No entanto, investigações recentes demonstraram que ataques de desanonimização podem ser conduzidos por adversários poderosos a nível estatal, utilizando correlação de tráfego e técnicas avançadas de aprendizagem automática para identificar padrões e comprometer a privacidade dos utilizadores.

Esta dissertação aborda estes desafios, reforçando as defesas da rede Tor contra os ataques supramencionados. O principal objetivo é desenvolver uma solução inovadora para melhorar os nós de retransmissão Tor (Relay Nodes), aproveitando a arquitetura de software existente e introduzindo garantias formais de privacidade baseadas em circuitos privados diferenciados. Pretende-se que a solução proposta preserve níveis práticos de throughput e latência para comunicações de ponta a ponta, garantindo a compatibilidade com cenários de utilização real e proporcionando aos utilizadores uma proteção do anonimato mais robusta e adaptável.

Como referido anteriormente, o cerne da solução proposta assenta na integração dos princípios da Privacidade Diferencial, introduzindo dinamicamente ruído aleatório cuidadosamente delimitado nos fluxos de tráfego, mitigando os ataques de correlação e reforçando a resistência à recolha de impressões digitais. Tanto quanto é do nosso conhecimento, os contributos esperados desta dissertação poderão representar um avanço pioneiro na incorporação da Privacidade Diferencial na arquitetura de software dos nós de retransmissão Tor (Relay Nodes). Além disso, a aplicação da Privacidade Diferencial

permitirá o desenvolvimento de um mecanismo formalmente comprovado, escalável e eficaz, que reforça o anonimato online e se mantém resiliente face às técnicas mais avançadas de censura na Internet, protegendo, assim, o direito fundamental à liberdade de expressão num mundo digital cada vez mais sujeito a monitorização.

Palavras-chave: Redes de Anonimato, Privacidade Diferencial, Comunicação Preservadora de Privacidade, Ataques de Análise de Tráfego, Garantias Reforçadas de Privacidade Ponto a Ponto, Correlação de Tráfego

Contents

List of Figures vii

1 Introduction 1
- 1.1 Motivation . . . 2
- 1.2 Problem . . . 3
- 1.3 Goals . . . 3
  - 1.3.1 Expected Contributions . . . 4
- 1.4 Report Organization . . . 4

2 Related Work 5
- 2.1 Anonymity Networks & Mixnets . . . 5
  - 2.1.1 Groove . . . 6
  - 2.1.2 Loopix . . . 7
  - 2.1.3 Stadium . . . 8
  - 2.1.4 MIRACE . . . 8
  - 2.1.5 Problems & Analysis . . . 9
- 2.2 Tor Network . . . 9
  - 2.2.1 Tor Bridges & Pluggable Transport . . . 10
  - 2.2.2 Tor Hidden Services . . . 10
  - 2.2.3 Tor Circuit Scheduling . . . 11
  - 2.2.4 Tor Vulnerabilities . . . 12
- 2.3 k-Anonymity . . . 14
- 2.4 Differential Privacy . . . 15
  - 2.4.1 Properties of Differential Privacy . . . 16
  - 2.4.2 Local Differential Privacy . . . 17
  - 2.4.3 Differential Private Communication . . . 18
- 2.5 Summary . . . 18

3 System Model & Software Architecture 20
- 3.1 TTT Proposal . . . 20

3.2 System Model . . . 21
3.3 Threat Model . . . 23
3.4 Software Architecture . . . 25
3.4.1 Tor Relay Architecture . . . 25
3.4.2 TTT Architecture . . . 26
3.5 Parameterization . . . 27
3.5.1 Used Techniques . . . 27
3.5.2 Jitter Parameterization . . . 27
3.5.3 Packet Padding Cells Parameterization . . . 27
3.6 Jitter Development Strategy . . . 27
3.7 Packet Padding Cells Development Strategy . . . 27
3.8 Discussion . . . 27

4 TTT Prototype and Implementation **28**
4.1 Implementation Guidelines and Techniques . . . 28
4.2 Implementation Model . . . 28
4.3 Prototyping Metrics . . . 28
4.4 Complexity Analysis . . . 28
4.5 Prototype Availability . . . 28
4.6 Summary . . . 28

5 Validation and Experimental Evaluation **29**
5.1 Evaluation Criteria . . . 29
5.1.1 Testbenchs . . . 29
5.1.2 Experimental Observations . . . 29
5.2 Performance Evaluation . . . 29
5.2.1 Packet Padding Cells . . . 29
5.2.2 Jitter Injector Tor Schedulers . . . 30
5.3 Unobservability Evaluation . . . 34
5.3.1 Methods and Tools . . . 34
5.3.2 Experimental Observations . . . 34
5.4 Formal Validation . . . 34
5.5 Discussion . . . 34
5.6 Summary . . . 34

6 Conclusions **35**
6.1 Main Contributions . . . 35
6.2 Future Work . . . 35

Bibliography **36**

List of Figures

3.1 Differential Private Network System Model 22
3.2 Threat Model over System Model 24
3.3 Original Cell Processing Pipeline 25
3.4 TTT Cell Processing Pipeline 26

5.1 Packet Padding Cells Generation and Impact 29
5.2 Download Latency with Packet Padding Cells Generation 30
5.3 Download Throughput with Packet Padding Cells Generation 30
5.4 Download Total Time with Packet Padding Cells Generation 31
5.5 Tor Schedulers and Jitter Impact on Latency 31
5.6 Tor Schedulers and Different Distributions Impact on Throughput 32
5.7 Tor Schedulers and Total Time For Different Distributions 33

1

Introduction

Freedom of speech and access to information are fundamental rights that must people take for guaranteed. However, in some regions of the world, these rights are denied due to powerful and oppressive regimes and state level adversaries with access to vast resources that control the flow of information and restrict the access to the internet, in order to control the population [3, 70, 2].

To protect these populations and ensure their access to freedom of speech and information, Anonymity Networks are anonymity-preserving network that aim to anonymize their users and protect their privacy and security. Authoritarian regimes intensify their efforts to restrict and control their population access by employing advanced techniques. With these efforts, Anonymity Networks are becoming increasingly important to protect the privacy and security of users, as well as the privacy and anonymity guarantees offered by these networks.

These regimes resort to the use of traffic analysis techniques to passively monitor the traffic and infer information about the users, in order to enforce and control the network and their users. Traffic analysis attacks are a significant threat to Anonymity Networks, as they can be used to infer users' activities and deny their privacy, as well as de-anonymize them [8, 66, 30]. Some have demonstrated that advances in machine learning lead to more sophisticated and effective traffic correlation attacks [39, 24, 7], such as DeepCoFFEA [42] and FINN [49], which leverage deep learning techniques to correlate traffic patterns and infer users' activities with high accuracy, in Anonymity Networks like Tor [33]. Fingerprinting attacks also represent a major threat to Anonymity Networks, as they allow attackers to learn information about user's online activities, even on encrypted traffic, by eavesdropping and collecting side-channel information, between the user and the entry node. With this information, the attacker can use these machine learning techniques to infer the user's online activities, such as the websites visited, by analyzing the traffic patterns, with high accuracy [54, 48, 10].

As attacks grow more sophisticated, journalists, whistleblowers, and anyone striving to exercise their freedom of speech in oppressive regions face increasing risks of identification and persecution. This not only diminishes the utility of anonymity networks but also fuels

mistrust and fear. To combat these threats, the scientific community must develop robust solutions that strengthen anonymity guarantees, backed by proven privacy and security measures.

1.1 Motivation

Tor [14], one of the most popular Anonymity Networks, is an anonymous communication service based on the Onion Routing Protocol. This network is designed to protect users' privacy and security by routing their traffic through a network of volunteer-operated servers, encrypting it at each step, and ensuring that no single server knows both the source and destination of the traffic.

As stated previously, Anonymity Networks remain vulnerable to attacks such as traffic analysis, which threatens the anonymity and security of the users these networks are designed to protect [8, 66, 30, 13, 36, 54, 48, 10, 53, 69]. To address the challenge of circumventing censorship, researchers have proposed a range of solutions.

Some of these proposed sophisticated methods include Traffic Encapsulation, Pluggable Transports, k-Anonymity, Multipath Strategies, and Traffic Splitting, which we briefly present. Traffic encapsulation is a method that conceals data by embedding packets within other data formats or protocols, thereby obfuscating their true content. This technique leverages protocols, such as those for web browsing and video streaming, to disguise network traffic, making it indistinguishable [61, 44, 65, 5, 19, 25]. Pluggable Transports serve as a mechanism for obfuscating network traffic by employing diverse protocols to connect to the Tor network, thereby complicating an attacker's ability to identify and analyze the traffic [47, 46, 61]. k-Anonymity is a formal privacy definition designed to ensure that an individual's data remains indistinguishable within a larger set [40, 57, 23]. Meanwhile, Multipath Strategies and Traffic Splitting enhance security by distributing data across multiple concurrent paths, reducing the risk of interception and traffic analysis [61, 44, 45, 63]. Similarly, traffic shuffling is a technique that focus on reordering packets, with statistical or cryptographic techniques, to make it difficult for an attacker to correlate the traffic [51, 35, 11, 45, 67].

Some of these solutions have presented promising results in terms of privacy and security. However, we argue that the privacy guarantees provided by these solutions are not quantifiable, potentially making it difficult to evaluate their effectiveness and to compare them with other solutions. To address this problem, we apply Differential Privacy (DP) [15, 17, 40] on the Tor Network enhance its resistance against fingerprinting and correlation attacks, and provide quantifiable formal privacy guarantees to users, while maintaining practical levels of performance. DP was originally designed for database analysis and privacy preserving analysis but some works on communications systems have shown that DP can also be used to enhance privacy [45, 60, 62, 68, 50].

1.2 Problem

Although Anonymity Networks are designed to provide robust privacy and security, advancements in traffic analysis and correlation techniques have increasingly exposed vulnerabilities, reducing their reliability and trustworthiness for users — particularly in regions where pervasive surveillance by authoritarian regimes is a critical concern. These regimes actively invest in sophisticated surveillance methods and traffic manipulation strategies to maintain control over digital communication [3, 70, 2]. Tor [14], despite being one of the most widely used Anonymity Networks, is not immune to these challenges. It remains vulnerable to several attacks, including traffic correlation, congestion-based attacks, timing-based correlations, and fingerprinting techniques [8, 33, 66, 30]. Such vulnerabilities highlight the ongoing arms race between those striving to safeguard online anonymity and those seeking to undermine it through technical and infrastructural advancements, such as deep learning techniques.

The absence of standardized metrics for evaluating privacy guarantees in a formal and rigorous manner raises concerns about the effectiveness of anonymizing networks like Tor. Research has demonstrated vulnerabilities in the Tor network to various attacks, including traffic analysis. For instance, Chakravarty et al. showed that statistical correlation techniques can be used to perform successful traffic analysis attacks [8]. Additionally, Shi and Matsuura demonstrated that user privacy can be compromised through fingerprinting attacks [52].

1.3 Goals

As stated previously in Section 1.1, it is important to ensure that Anonymity Networks are secure against threats like traffic fingerprinting and correlation attacks, empowered by deep learning techniques, and to address the lack of standardized privacy metrics to quantify the anonymity guarantees provided by these networks.

To address this challenge, we add Differential Privacy (DP) into the Tor open-source project, by incorporating Differential Privacy-based mechanisms in the existing software architecture of Tor relay node, in order to dynamically inject carefully bounded random noise into traffic flows to provide formal privacy guarantees to users. Our work builds upon the foundation laid by Jansen et al. [31], who introduced KIST, a congestion control scheduler for Tor designed to enhance network performance by reducing latency and increasing throughput. While KIST achieves significant efficiency gains, its focus remains solely on performance, without introducing additional privacy guarantees or defenses against traffic analysis attacks beyond those inherent to Tor. Inspired by this limitation, we extend KIST's approach, not only preserving its performance benefits but also reinforcing privacy protections and bolstering resistance to traffic analysis attacks.

With this solution, we reinforce the Tor end-to-end communication anonymity guarantees, even under traffic fingerprinting and correlation attacks. To ensure real-world

applicability of this solution, it is important to maintain practical levels of throughput and latency for end-to-end communication.

1.3.1 Expected Contributions

To address the stated goal, we produce the following contributions[1]:

1. Definition and formalization of the system design model to incorporate differentially private-based circuits as a possible interesting solution for Tor relay nodes to improve Tor anonymity conditions against traffic fingerprinting and correlation attacks;

2. Specification of the software architecture and modular components for the engineering effort using Differential Privacy-based mechanisms and related components in extended Tor nodes;

3. Prototyping of the proposed solution where the implemented nodes will be used in a validation test bench and publishing the prototype in open-source for the possible use of interested researchers and practitioners;

4. Validation of the designed solution and prototype considering three main pillars: (*i*) Development of formal security proofs behind the designed solution; (*ii*) conduct an extensive experimental on performance indicators including communication latency, throughput, as well as resources' usage for running the proposed nodes, and (*iii*) experimental observation of the unobservability properties of the designed solution against traffic correlation and fingerprinting techniques.

In the final validation effort we obtained comparative metrics considering a test bench involving our designed solution and the use of the Tor network and current Tor relay nodes

1.4 Report Organization

The remainder of this report is organized as follows: Chapter 2 introduces Anonymity Networks and Differential Privacy, as well as some other related topics. The proposal of work is presented on Chapter 3, which provides some system model and guidelines for the development and discusses the proof method to verify the work. Finally, workplan is explained and addressed in Chapter **??**.

[1]The dissertation effort is planned as a research project task within NOVA LINCS (Nova Laboratory of Informatics and Computer Science), Computer Systems Research Group. The elaboration phase will involve integrating the expected contributions with collaborative engagement and follow-up of two PhD students: João Afonso Vilalonga and Hugo Gamaliel Pereira.

2

Related Work

In this chapter, we provide a more detailed overview of the state-of-the-art concepts introduced in the previous chapter. First, we discuss the concept of Anonymity Networks and Mixnets, followed by a detailed review of notable works in this field. Next, we examine the Tor Network, including key technologies such as Pluggable Transports and the Tor relay node scheduler. We then introduce *k*-Anonymity, and finally, we provide an in-depth overview of Differential Privacy.

2.1 Anonymity Networks & Mixnets

Mix Networks (Mixnets) were introduced by Chaum [9] as a public key cryptography based protocol designed to obscure the relationship between senders and receivers in a communication network, as well as its content, without the need for a universal trusted authority. Mixnets consist of a series of intermediary servers, called 'mixes', through which messages are decrypted, encrypted, randomly permuted and sent forward. To transmit messages, participants encapsulate their data within successive layers of encryption, with each layer corresponding to a specific mix in the network — a method conceptually akin to onion routing [20]. The length of the encrypted messages is proportional to the number of mixes. Upon receiving a message, each mix removes the outermost layer of encryption, processes the decrypted instructions, and forwards the partially decrypted message to the subsequent mix. This iterative procedure continues until the message arrives at the final mix, which removes the last layer of encryption and transmits the plaintext message to the intended recipient. By design, this system ensures that no single mix possesses complete knowledge of both the sender's identity and the recipient's address, thereby achieving a robust level of anonymity.

Various designs of mixnets have been proposed, based on Chaum work, addressing multiple security and performance issues. Flash Mixing, introduced by Jakobsson [27], focuses on achieving strong anonymity guarantees with reduced latency and computational overhead by shuffling messages, broadcasting the encrypted list to all mixes, who then together compute the output. Hybrid Mixnets [28, 43] efficiently combine public-key

and symmetric-key cryptography. Real-time Mixnets [32] aim voice and data communication where continuous data streams have to be transmitted and provide low-latency communication, as long as a certain delay at the start of a connection is tolerable. Work on this field has focused both on security and privacy guarantees, as well as on performance and scalability, with the goal of providing a practical and efficient solution for anonymous communication. Mixmaster [38] allows the sender of a message to remain anonymous to the recipient. Mixminion [12] uses fixed sized messages and supports anonymous replies and ensures forward anonymity using link encryption between nodes. Onion Routing [20] followed a similar approach, but focused on low latency communication, where messages are encrypted in layers and decrypted by a chain of authorized nodes.

Tor [14] is one of the most popular low latency anonymity system and protects against sender-receiver message linking against a partially global adversary and ensures perfect forward secrecy, messages' integrity, and congestion control. However, Tor is vulnerable to traffic analysis attacks, in case the adversary is able to observe the entry and exit points of the network. Mix-In-Place (MIP) [41] is a mixnet that uses a cascade of functions in a single proxy, instead of multiple intermediary nodes, to provide anonymity, and proving more resistant to traffic analysis attacks. Vuvuzela [22] operates in rounds, leading to offline users' inability to receive messages and all messages must transverse a single chain of relay servers. This design protects against both active and passive adversaries unless there is no honest mix node in the network.

2.1.1 Groove

Groove [4] is a scalable, metadata-private messaging system designed to support users with multiple devices, enabling them to send messages at any time, even when recipients are offline, while conserving bandwidth and energy. Built on mixnets, Groove ensures unlinkability between senders and their messages by shuffling batches of messages across servers. Traditional mixnets, however, require all users to submit messages during every round and receive messages at a synchronized rate to prevent correlated traffic patterns, which limits their scalability and usability. To overcome these limitations, Groove introduces a novel approach called oblivious delegation. In this model, users interact with an untrusted service provider that participates in the mixnet on their behalf and synchronizes their clients across all devices. This ensures that even if the service provider is compromised, an adversary cannot infer communication metadata. Unlike prior systems such as Karaoke [35], Stadium [60], and Vuvuzela [22], which require users to be online simultaneously, communicate in synchronized rounds, and support only single-device usage, Groove addresses these restrictions, offering asynchronous messaging and multi-device support. A significant advantage of Groove lies in its ability to minimize the resource demands of each message channel on the mixnet, with a particularly notable reduction in memory usage compared to earlier systems. Groove ensures Differential Privacy even against attackers with full network control and the ability to compromise

multiple servers. Users communicate through persistent message channels, similar to Tor's circuits, while their service provider handles message submission to the mixnet, stores received messages, and synchronizes clients across devices. To maintain anonymity, Groove leverages Parallel Mixnets, which efficiently scale with the number of servers by offering multiple parallel routes for processing messages. Differential privacy is a core goal of Groove, ensuring that traffic patterns between a user's client and their service provider do not disclose information about the user's communication partners. This is achieved through a scheduler that operates independently of the sender-recipient relationship. In contrast to Loopix [45], Groove maintains user privacy even if their service provider is compromised, eliminating the need for users to operate their own servers.

2.1.2 Loopix

Loopix [45] is a low-latency anonymous communication system offering bidirectional 'third-party' sender and receiver anonymity as well as unobservability. It employs cover traffic and Poisson mixing to provide strong anonymity guarantees and resist traffic analysis by a Global Network Adversary (GNA). The system ensures robust sender-receiver unlinkability against a Global Passive Attacker (GPA) capable of observing all network traffic between users, providers, and mix servers, even in cases where some mix nodes are compromised. Loopix also guarantees sender online unobservability under a corrupt provider and receiver unobservability under the condition of an honest provider. Consequently, the GPA cannot infer the type of transmitted messages, while intermediate nodes remain unable to distinguish between real messages, dropped cover messages, or loops of traffic generated by clients and other nodes. To protect against active attacks, mixes and clients employ self-monitoring mechanisms through self-injected traffic loops. These loops not only act as cover traffic to strengthen anonymity but also enhance sender and receiver unobservability. Similar to Groove, Loopix utilizes service providers to mediate access to the network, manage accounting, and enable offline message reception. These semi-trusted providers enforce rate limits, store messages for offline users, and facilitate message retrieval at a later time. Despite this reliance, Loopix is resilient against adversaries capable of observing all communications and conducting active attacks. A distinguishing feature of Loopix is its continuous operation, unlike Groove's deterministic round-based approach. Messages in Loopix can be retrieved at any time, ensuring users do not lose messages while offline. Furthermore, Loopix employs Poisson mixing, a simplified version of the stop-and-go mixing strategy [34], which introduces independent delays for each message, making packet timings unlinkable. Unlike circuit-based onion routing, where a fixed path is established, Loopix determines the communication path for each individual message independently, even when sent between the same pair of users. This approach enhances privacy and unpredictability. The Poisson mix operates as follows: mix servers listen for incoming packets, check for duplicates, and decode received messages using their private keys. Duplicates are discarded, and the next mix

packet is extracted. Decoded packets are not forwarded immediately; instead, each packet is delayed based on a pre-determined delay specified by the source.

2.1.3 Stadium

Stadium [60] is a point-to-point messaging system that ensures metadata and data privacy while efficiently scaling its workload across hundreds of low-cost providers operated by different organizations. Stadium achieves its provable privacy guarantees through the use of noisy cover traffic and Differential Privacy, which bounds metadata leakage over time. The system employs a mixnet architecture, where user messages and noise messages are distributed among providers and mixed in parallel. Each provider processes only a fraction of the total messages, ensuring scalability and efficiency. As long as at least one provider remains honest, Stadium achieves global verification and secure mixing. Communication in Stadium occurs in fixed rounds, with each round processing a batch of messages accumulated from users during the preceding interval. To enforce Differential Privacy, servers generate cover traffic at the start of each round. Noise messages are injected into the system in a single, large step before mixing begins. However, this design leaves room for malicious servers to discard noise messages before mixing. To address this vulnerability and ensure that noise remains in the system, Stadium employs cryptographic techniques for verifiable message processing. These techniques enable honest servers to verify the actions of others, preserving the system's Differential Privacy guarantees. To optimize the computational workload of this verification process, Stadium introduces hybrid verifiable shuffling. Stadium's mixing process relies on a parallel mixing scheme. Each server initially processes a small fraction of input messages, mixes them, and redistributes the mixed messages among other servers. This cycle of mixing and redistribution is repeated multiple times. Although messages are initially partitioned based on their originating server, the repeated cycles of splitting and mixing result in global mixing across all servers.

2.1.4 MIRACE

MIRACE [44] is a censorship circumvention system developed by Pereira that leverages dynamically constructed circuits to provide secure communications. This work allows traffic to be split across multiple circuits, each composes of several nodes. These solutions combine traffic splitting with encapsulations, through the combination of TLS, QUIC and WebRTC tunnels for diverse covert communication strategies, and traffic shaping, by the addition of jitter and padding. Pereira demonstrated that MIRACE is able to ensure secure communications against correlation attacks, even with machine learning-based website fingerprinting attacks. The author also showed that MIRACE is able to maintain practical levels of latency and throughput, even with the addition of nodes. Under fingerprinting attacks, MIRACE showed strong levels of resistance with high accuracy and the ability to effectively obscure traffic patterns.

2.1.5 Problems & Analysis

Mixnets have been analyzed in terms of security and privacy guarantees, as well as performance and scalability. Zhu et al. revealed that flow-correlation attacks can be used to de-anonymize users in mixnets, in relation to the system parameterization, such as sample size, noise level, payload flow rate, and detection rate [69]. Shmatikov and Wang demonstrated that Mixnets are vulnerable to timing analysis, even with defenses in place, by the use of advanced statistical techniques, such as cross-correlation and machine learning, successfully de-anonymizing users [53].

2.2 Tor Network

Tor [14] is a circuit-based low-latency anonymous communication service based on *onion routing* that aims to anonymize TCP-based applications, such as web browsing, SSH and instant messaging. The network relies on *Onion Routers* (ORs) that are responsible for maintaining TLS connections to every other OR and for forwarding traffic along the circuits, and are also maintained by volunteers.

An *Onion Proxy* (OP) is the local software run on the client and that handles connections with the users' applications, fetches the current network information and the lists of *Onion Routers* (ORs) and builds circuits through the network. Each OP maintains TLS connections to nodes they have been in contact recently and a pair of keys: long-term identity key and short-term onion key. Long-term keys are used to sign TLS certificates and its router descriptor. Short-term keys are used to decrypt requests, negotiate ephemeral keys and set up circuits.

Traffic travels the network through 512 bytes fixed-size *cells* with a header and a payload. The header contains the circuit identifier and the command to describe what to do with the payload. Based on the header's command, cells are either control cells or relay cells. Control cells are used to manage circuits and connections, while relay cells are used to carry data and can only be sent after a circuit is established. Relay cells have a digest assigned by the sender and its header and payload are iteratively encrypted with the symmetric key of each hop up to the target OR. This way, only the last OR in the circuit can read the payload, given the fact that the digest is encrypted to a different value at each step.

The sequences of ORs which traffic is routed through are called *circuits*, normally composed by 3 relays (entry or guard, middle and exit). To create a new circuit, a user's OP must execute a Diffie-Hellman key exchange with each OR in the circuit, one hop at a time. This way, each OR knows the previous and the next node in the circuit, but not the source and destination of the data. Once the circuit is established, relay cells are sent through the circuit,

2.2.1 Tor Bridges & Pluggable Transport

One vulnerability about Tor is the traffic blocking by ISPs of censored regimes. One way to perform this blocking is by blacklisting Tor relays, since the list of their IP addresses is public, which prevents clients from establishing circuits. Bridges come as a solution to this problem, where unpublished proxies forward client's traffic to a Tor entry relay, making the client connect to some unlisted bridge, rather than some potentially blacklisted relay [37]. Nevertheless, bridges are still vulnerable to traffic fingerprinting attacks. Matic, Troncoso, and Caballero demonstrated that 55% of public bridges are vulnerable to the traffic blocking referred before. Even with encrypted traffic, Tor's traffic is still identifiable by its packet size, currently fixed-sized 512 byte cells, and by the TLS byte patterns, more precisely the TLS extension called Server Name Indication (SNI), which adds the domain name to the TLS header without any encryption.

To overcome this issue, Tor bridges support *pluggable transports* (PT) [47] which transform the Tor traffic flow between the client and the bridge, making traffic monitoring between these look innocent, instead of actual Tor traffic. Some examples of PTs are *obfs4*, *meek*, *Snowflake* and *WebTunnel* [46]. *obfs4* obfuscates Tor traffic to make it appear random, thwarting efforts by censors to detect it through Internet scanning; *meek* disguises Tor traffic to resemble ordinary browsing activity on prominent websites; *Snowflake* routes your connection through volunteer-operated proxies to make it look like you're placing a video call instead of using Tor; and *WebTunnel* camouflages Tor traffic by making it indistinguishable from standard HTTPS requests, thereby creating the illusion of accessing a secure website.

2.2.2 Tor Hidden Services

Tor circuits, by their inherent design, do not ensure the anonymity of the receiver. This limitation arises from the fact that, for a client to reach a destination via an exit node, the recipient's IP address must be publicly accessible. To address this vulnerability, Tor introduces hidden services and the concept of a rendezvous point (RP) — a Tor relay node that facilitates recipient anonymity through the construction of dual circuits.

Hidden services enable any entity to host a TCP-based service without disclosing its IP address. Clients wishing to access such services utilize a special onion service address, which is derived from the service's public key. This cryptographically generated address provides a secure means of routing without exposing network-layer identifiers.

The fundamental mechanism underpinning hidden services involve the establishment of two distinct Tor circuits: one from the client to the rendezvous point and another from the hidden service to the same point. These circuits are constructed independently, and neither endpoint is aware of the other's IP address. The rendezvous point serves solely as an intermediary to facilitate communication between the two parties.

This architecture inherits the sender anonymity traditionally offered by Tor circuits and extends it to receivers. Only the respective entry nodes of the circuits are aware of the

origin of the traffic, and no single entity can link the sender and receiver. Notably, the exit node of the client circuit terminates at the rendezvous point, perceiving it as the ultimate recipient. Consequently, the client does not require knowledge of the hidden service's IP address, thereby ensuring the recipient's anonymity.

In summary, the use of rendezvous points and independently constructed circuits within Tor's hidden service framework effectively achieves mutual anonymity, safeguarding both client and service identities during communication.

2.2.3 Tor Circuit Scheduling

Traffic handling in Tor involves several buffers and schedulers. Each circuit is used by only one client but if multiple circuits use the same two ORs, they share the same connection. That is, OR will have multiple simultaneously connections with other ORs but only one to any given OR, and each connection will transport data for various circuits. When a OR receives a packet from an TCP stream, the packet gets demultiplexed, together with other possible incoming packets from different TCP streams, and they get placed into the kernel socket input buffers. Here packets are processed by the OS, usually in FIFO order, and sent to the Tor input buffers. Upon receiving the packets, Tor will remove the TLS layer and onion-encrypted (or onion-decripted, depending on the circuit's direction) and finally enqueued in the Tor Circuit Queue. Each relay maintains a queue for each circuit that it serves. Cells from the same Tor input buffer might not be enqueued in the same circuit queue, as they might belong to different circuits. The cells are then selected, dequeued, onion-encrypted and stored in a Tor output buffer. The data is then written to a kernel socket output buffer when the Tor output buffer contains sufficient data to form a TLS packet.

Tor has a congestion control mechanism (Circuit-Level Throttling), designed to regulate communications across circuits, which ensures resource distribution, prevents network congestion and provides safety against attacks such as Denial-of-Service. As Tor is a low latency network and its security guarantees rely on how many users are using the network, it is important to have a good congestion control mechanism and practical levels of latency and throughput. However, it has been shown that Tor's Circuit Scheduling Algorithm allows busy circuits to crowd out bursty circuits, leading to congestion and latency issues [59]. Additionally, Jansen et al. have demonstrated that Tor's congestion occurs in the kernel socket buffers. Both these works have proposed solutions to improve Tor's congestion control and latency issues [31].

Even with the improvements in congestion control, as merging Tang and Goldberg and Jansen et al. works, Tor is still vulnerable to traffic analysis attacks such as website fingerprinting and correlation-based attacks [31]. Even though none of these works presented additional security risks or vulnerabilities to the Tor Network, they have shown little or no improvement in the network against these attacks.

2.2.3.1 Exponential Weighted Moving Average (EWMA)

Tang and Goldberg proposed a circuit scheduling algorithm that handles circuits by their recent activity, where bursty circuits have higher priority over busy ones [59]. Generally, bursty circuits are those used for web browsing and instant messaging, and also referred as interactive streams. On the other hand, busy circuits are those used for file transfers, and also referred as non-interactive streams. Interactive streams are more sensitive to latency while non-interactive streams tolerate higher delays. The approach is based on the Exponential Weighted Moving Average (EWMA) algorithm, which assigns a weight to each circuit based on the number of cells sent on each circuit. When selecting the circuit to process, the algorithm chooses the one with lowest EWMA value. Usually, newly created circuits have a lower EWMA value, leading to a higher priority. Even though this algorithm has merged into Tor, it has been demonstrated that it actually reduces performance for clients under some network conditions [29]. Additionally, this work does not address the adversaries' ability to correlate and fingerprint users' traffic, which is a major issue in Tor Network.

2.2.3.2 Kernel Informed Socket Transport (KIST)

Motivated by the still congestion problem with Tor and the lack of understanding of where this problem occurs, Jansen et al. found that the congestions occurs inside the kernel socket buffers and on Tor's sockets management. They proposed a new scheduling algorithm, Kernel Informed Socket Transport (KIST), that solves these problems by choosing from all circuits with writable data rather than just those belonging to a single TCP socket and by dynamically managing the amount of data written to each socket based on real-time kernel and TCP state information that can be queried from user space. This way, KIST reduces Tor's circuit congestion by more than 30%, reduces network latency by 18%, and increases network throughput by nearly 10% [31]. KIST was merged and configured as default socket scheduling algorithm in Tor since January 2018, replacing the EWMA-based algorithm, referred in Section 2.2.3.1. Although, KIST authors acknowledge that it does not affect the adversaries abilities to collect accurate measurements required for the throughput correlation attack, compared to the 'vanilla' Tor scheduling algorithm.

2.2.4 Tor Vulnerabilities

Studies have pointed that Tor is susceptible to traffic analysis attacks, where an adversary can infer information about a user, like who is communicating with whom, by observing the traffic patterns. Here we cover some of the most common traffic analysis attacks against Tor, which have been demonstrated to be effective in de-anonymizing users.

2.2.4.1 Website Fingerprinting Attack

Chakravarty et al. also proposed a traffic analysis attack model against Tor where the adversary uses statistical correlation over the server to exit and entry to client traffic to find similar traffic patterns [8]. This attack consists on clients generating sustained traffic for a long period of time, by downloading a file for example, and the adversary being therefore able to inject traffic patterns by perturbing the TCP connection. This way, the adversary can obtain traffic analysis from the server to exit node and correlate it with the traffic from the entry node to the client, by finding the flow which carries the injected fingerprint. Testing showed that the attack was able to correctly identify the source of anonymous traffic 81.4% of the time. Deep Fingerprinting [54] is a website fingerprinting designed using deep learning techniques, that uses a simple input format and does not require handcrafting feature for classification. The authors demonstrated that this attack is 98.3% accurate in identifying the website visited by the user. Sirinam et al. proposed a website fingerprinting attack that uses triplet networks and a machine learning technique, that requires few training samples, called N-shot learning, which reduces the effort of gathering and training with large datasets [55]. Finally, this work achieved up to 95% accuracy in identifying the website visited by the user, only by using 20 examples per website.

2.2.4.2 Correlation-Based Attacks

Correlation attacks were acknowledged by Dingledine, Mathewson, and Syverson [14] as a potential threat to users' anonymity guarantees in the Tor Network. These attacks exploit the adversary's ability to observe both the entry and exit points of the network, allowing them to correlate the traffic patterns and de-anonymize users. Sun et al. [56] proposed asymmetric traffic analysis as an end-to-end timing analysis that allows AS-level adversaries to compromise Tor users' anonymity. Internet path from exit relays to the web server may differ from the path from the web server to the exit relay, allowing the adversary to observe the TCP acknowledgement traffic on the path from the server to the exit relay, even if the adversary is enabled to observe the data traffic on path from the exit relay to the server. This attacks might be applicable where the adversary observes: data traffic from the client to the entry relay and from the exit relay and the server; data traffic from the client to the entry relay and TCP acknowledgement traffic from the server to the exit relay; TCP acknowledgement traffic from the entry relay to the client and data traffic from exit relay to the server; or TCP acknowledgement traffic from the entry relay to the client and TCP acknowledgement traffic from the server to the exit relay. The authors proposed a suite of new attacks against Tor called 'RAPTOR', where they demonstrated an accuracy of 95% in correlating client/server pair.

DeepCorr [39] and its extension DeepCoFFEA [42] are both systems that leverage advanced deep learning techniques to correlate traffic. DeepCorr is able to correlate Tor flows by first using deep learning to learn a correlation function and then using this

function to cross-correlate the traffic. In contrast to website fingerprinting, this work has no need to learn target destinations, instead the function can be used to link flows on arbitrary destinations. Additionally, it does not require the traffic to be sent in the same circuits used to learn the function. DeepCoFFEA extends DeepCorr by using a modified triplet network approach and amplification techniques, demonstrating that it achieved lower computational costs and higher efficiency compared to DeepCorr, improving the accuracy of the correlation attack.

2.3 k-Anonymity

To address the exponential growth in the number and variety of data collections containing person-specific information, and the pressing need to prevent privacy compromise, k-Anonymity [57] emerged as a foundational privacy model. This model ensures data anonymity not only by removing explicit identifiers such as names, addresses, or phone numbers but also by protecting against re-identification through linking or matching data with external sources. It achieves this by mitigating the risk posed by unique characteristics within the dataset.

Earlier work by Sweeney demonstrated the vulnerability of anonymized data to linkage attacks using the 1990 U.S. Census. They highlighted that 87% of the U.S. population reported characteristics that rendered them unique, and over half of the population could be uniquely identified using just three attributes: place, gender, and date of birth [58]. To counter these risks, Sweeney defined k-Anonymity as a property satisfied by a dataset `D` if, and only if, each sequence of values for quasi-identifiers in `D` is indistinguishable from at least $k-1$ other records in `D`. In practical terms, any tuple in `D` must be identical to at least $k-1$ other tuples concerning its quasi-identifiers. This property ensures that the data cannot be easily linked to other sources and protects users from re-identification, with the privacy guarantees strengthening as k increases.

Ahn, Bortz, and Hopper extended the concept of k-Anonymity to communication systems, defining a protocol as sender k-anonymous if it ensures that an adversary attempting to identify the sender of a message can narrow their search to a set of k potential senders. Similarly, receiver k-Anonymity guarantees that an adversary can only narrow down the possible recipients to a group of k [1].

Despite its foundational role, k-Anonymity remains vulnerable to several types of attacks, including unsorted matching, complementary release, and temporal attacks, as noted by Sweeney [57]. Unsorted matching attacks exploit the order of tuples in a released dataset, which can be mitigated by randomizing the tuple order. Complementary release attacks arise when subsequent data releases are combined, allowing adversaries to re-identify individuals by correlating datasets. Temporal attacks exploit data changes over time, such as additions, deletions, or modifications, leading to violations of k-Anonymity guarantees as datasets evolve.

Further research has explored enhancing k-Anonymity to address its limitations. For instance, Hopper and Vasserman observed that receiver k-Anonymity improves effectiveness against long-term intersection and statistical disclosure attacks. Additionally, incorporating periodic sender k-Anonymity under low churn conditions enhances resistance to mass surveillance, albeit without achieving an optimal cost function [23].

Recognizing that k-Anonymity alone may not suffice to protect privacy in the era of big data, Gosain and Chugh proposed combining k-Anonymity with Differential Privacy. This hybrid approach addresses the challenges posed by the massive scale and interconnected nature of modern data systems [21]. Similarly, in response to the proliferation of social networks and the accompanying surge in data collection, Campan and Truta introduced an anonymization technique for social network data, masking it according to the k-Anonymity model [6].

While k-Anonymity offers practical privacy guarantees, it is primarily suited for systems prioritizing efficiency over robust anonymity protections. It remains less applicable in scenarios where strong, comprehensive guarantees are essential.

2.4 Differential Privacy

Differential Privacy [15, 17, 40], introduced by Dwork et al., is a formal notion of privacy and is a property of algorithms, rather than data like k-Anonymity. An algorithm or function satisfies Differential Privacy if for all neighboring datasets x and x', and all possible sets of outputs S:

$$\frac{\Pr[F(x) \in S]}{\Pr[F(x') \in S]} \leq e^{\varepsilon} \tag{2.1}$$

In this privacy mechanism definition, F is a randomized function, and its output will be similar, with or without the data of any specific individual. Moreover, F's randomness should be enough so that outputs do not reveal the input data. This way, we can ensure that the privacy of any individual is preserved, by ensuring plausible deniability. This happens because the output of the function cannot be correlated to any specific input, whether the input is present or not. Furthermore, the mechanism can maintain certain level of accuracy, due to having a precise understanding of the noise generation process. Differential Privacy intends to reject no malicious adversary, instead it ensures that the adversary is not capable of inferring any specific information, due to the randomness of the output.

Another important requirement of differential private mechanisms is the ϵ parameter [40, 17, 16, 15]. This parameter is called *privacy parameter*, and it controls the 'amount of privacy' that the correspondent function provides. Small values of ϵ require F to provide very similar output when given similar inputs, originating in higher levels of privacy. On the other hand, higher values of ϵ allow outputs to diverge more, leading to lower levels

of privacy. In practice, ϵ should be less or equal to 1, and no greater than 10, to provide meaningful privacy guarantees.

Differential Privacy originated from the challenge of enabling the extraction of meaningful insights about an underlying population while rigorously safeguarding individual privacy. Therefore, the earlier definitions of this concept were focused on datasets and queries over them.

The Laplace Mechanism was proposed together with the definition of Differential Privacy by Dwork et al. [17]. The simplest way to achieve Differential Privacy is by adding noise to the output of a function. The challenge is to balance the injected noise in order to have enough to satisfy the definition of DP but not too much to make the output useless. The Laplace Mechanism is a simple mechanism which, for a function $f(x)$ that returns a real number, $F(x)$ satisfies ϵ-Differential Privacy:

$$F(x) = f(x) + \text{Lap}(\frac{S}{\epsilon}) \tag{2.2}$$

where s is the sensitivity of the function f, and $Lap(S)$ denotes sampling from the Laplace distribution with scale S and center 0.

The sensitivity of a function is a measure of how much the output of the function can change when the input changes. This property is important to determine how much noise should be added to the output of the function, in order to guarantee Differential Privacy.

2.4.1 Properties of Differential Privacy

In this section, we cover the three main properties of Differential Privacy: Sequential Composition, Parallel Composition and Post-Processing.

2.4.1.1 Sequential Composition

Sequential Compositions [40, 16] is a major property of Differential Privacy, which determines the total privacy cost of releasing multiple results of differential private mechanisms on the same input. Formally, this property states that if $F_1(x)$ satisfies ϵ-Differential Privacy and $F_2(x)$ satisfies ϵ'-Differential Privacy, then the composition of these two functions, $G(x) = (F_1(x), F_2(x))$, satisfies $(\epsilon + \epsilon')$-Differential Privacy. This property is important to algorithms that access data more than once. The bound on privacy cost given by this property is an upper bound, and the actual privacy cost can be lower than the sum of the individual privacy cost.

2.4.1.2 Parallel Composition

Parallel Composition [40, 16] can be considered as an alternative to Sequential Composition, as it determines the total privacy cost of releasing multiple results of differential private mechanisms. The idea consists on splitting a dataset into disjoint chunks and applying a differentially private mechanism to each chunk separately. Formally, if $F(x)$ satisfies

ϵ-Differential Privacy, and we split the dataset X into k disjoint chunks such that $X = X_1 \cup X_2 \cup \ldots \cup X_k$, then the mechanism that releases all the results $F(x_1), \ldots F(x_k)$ satisfies ϵ-Differential Privacy. Comparatively, to Sequential Composition, Parallel Composition gives a better upper bound. Since F is run k times, Sequential Composition states that this procedure satisfies $k\epsilon$-Differential Privacy.

An issue with Differential Privacy arises with small datasets. A large dataset is able to achieve a strong privacy guarantee with relatively weak noise, and allows the results to be useful. However, small datasets require stronger noise to achieve the same privacy guarantee. As we split the dataset into chunks, we must care for the utility of the results and the impact of the number of chunks that we split the dataset into. The more chunks we split the dataset into, the stronger the noise we must add to the results, and the less useful the results will be.

2.4.1.3 Post-Processing

The last property is Post-Processing [40, 16], and it states that is impossible to reverse the privacy protection provided by a Differential Privacy by post-processing the data in some way. Formally, if $F(x)$ satisfies ϵ-Differential Privacy, then for any function h, $h(F(X))$ satisfies ϵ-Differential Privacy. This means that no danger will arise from performing additional computations on an output of a differential private mechanism, as the privacy guarantee will be preserved (and not reversed). This property is essential to ensure that Differential Privacy is resistant against privacy attacks based on auxiliary information and that attacks effectiveness is only limited by the privacy parameter ϵ.

2.4.2 Local Differential Privacy

Differential Privacy was initially proposed in a central model, where sensitive data was collected in a single dataset and the data curator must be a trusted entity in order to correctly execute differential private mechanisms, even in scenarios where the analyst is malicious [40]. However, this assumption might not be very realistic. In practice, the data curator and the analyst might be the same entity, and the data curator might not be trusted, leading to no differential private mechanisms being in place. This way, *Local Differential Privacy* (LDP) raises as a solution in which data is made Differential Privacy before being collected by the data curator, being already used by big companies such as Google [18] and Apple [26].

Randomized Response is a mechanism for LDP, proposed by Warner [64] in 1965, intended to improve survey responses about sensitive issues, but was not originally design as a mechanism for Differential Privacy. Dwork and Roth presented a variant of this mechanism, in which the subject flips a coin to answer a 'yes' or 'no' question. If the coin is heads, the subject answers truthfully, otherwise, the subject flips the coin again and answers 'yes' if heads and 'no' if tails. The randomization in this algorithm comes from the two coin flips, creating uncertainty about the true answer, and therefore providing privacy.

2.4.3 Differential Private Communication

Some studies have been conducted on applying Differential Privacy to communication systems [62, 68, 50]. These studies focus on providing privacy guarantees to users and protect them against traffic analysis attacks, powered by machine learning techniques.

Zhang et al. [68] proposed a differential private mechanism for streaming data, by the use of a Chrome extension with a DP mechanism that proxies streams between the browser and the server. The extension intercepts the requests from the client, which sends requests on behalf of the client based on a differentially private mechanism, instead of instantly relaying them immediately. The authors found that differentially privacy lowered the machine learning techniques, used by the adversary, efficiency, even though the accuracy of the attack was less affected in case of the adversary training model with the same DP mechanism.

NetShaper [50] is a network side-channel mitigation system, proposed by Sabzi et al., based on traffic shaping, which provides quantifiable and tunable privacy guarantees, through the use of Differential Privacy. This work shapes traffic based on a DP-mechanism that adds noise to the traffic, that can be split into 3 steps: *queue, query, and post-process*. In the first step, the system queues the input traffic. After a fixed parameterizable periodic interval, the system queries the queue and adds noise to the traffic, performed by the differential private traffic shaping algorithm. Finally, the packets are post-processed and transmitted to the network.

Vilalonga et al. [62] suggested Randomized Response could be used to strengthen the privacy of the Tor network, by adding noise to the traffic, making it harder for adversaries to correlate the traffic and de-anonymize users. The authors proposed a mechanism that uses pluggable transports to connect the client to the Tor Network and a noise-adding algorithm. This algorithm sends packets based on the Randomized Response mechanism, in this case, where the algorithm chooses to send a packet or a noise packet based on a probability.

2.5 Summary

Anonymity Networks have been around for almost 40 years but advances in machine learning and traffic analysis techniques have compromised the privacy guarantees that these networks aim to provide. Tor is a widely used low-latency Anonymity Network that has been shown to be vulnerable to such attacks. We take a particular interest in the developments on Tor's scheduling algorithms that proposed (and later been merged into the source code) solutions to improve Tor's congestion control. The works referred in this chapter have both demonstrated effects in performance and congestion control but lack on providing privacy guarantees that Tor aims to provide. Some work on Anonymity Network, including Tor, have tried to mitigate these attacks, such as website fingerprinting and correlation-based attacks, but they lack on giving formal privacy

guarantees. In most of the cases, privacy and security are only measured by practical testing, which may not be enough to ensure that the network is secure and private, if some conditions are not rigorously tested and proven. To address this problem, this dissertation introduces formal privacy guarantees to Anonymity Networks, more precisely to Tor, by implementing a Differential Private Tor Socket Scheduler that, which also offers adaptable and parameterizable privacy protections. To this day, there is no work that focus on applying Differential Privacy techniques into Tor Network.

3

System Model & Software Architecture

This chapter provides an in-depth exploration of the system model and software architecture of TTT. We begin by introducing the system's overarching objectives and foundational principles. Next, we define the design goals that guided its development, followed by a detailed exposition of the system model and software architecture. Subsequently, we present the threat model, analyzing the potential threats and adversaries accounted for in TTT's design. Finally, we conclude with a comprehensive summary that synthesizes the key insights discussed throughout the chapter.

3.1 TTT Proposal

As shown previously in section 1.1, Tor is a widely used anonymity network that provides users privacy and anonymity while browsing the internet. However, advances in machine learning and data analysis have made it possible to de-anonymize Tor users by analyzing their traffic patterns, therefore compromising the privacy and anonymity that Tor aims to provide [REFS AQUI].

This can be achieved through some traffic analysis attacks, such as fingerprinting and traffic correlation. As discussed in section 1.2, these traffic analysis attacks represent a significant threat to Tor users' privacy and anonymity, as sophisticated adversaries can use machine learning and artificial intelligence techniques to analyze monitored traffic patterns and de-anonymize users and their browsing activities.

This way, we propose TTT, an extension of Tor source code that includes a differential private scheduler to protect the users' data and prevent de-anonymization attacks. By incorporating Differential Privacy into the Tor network, we have increased the resistance against fingerprinting and correlation attacks. In addition, TTT also introduces a new differential private 'Packet Padding Cells' generation mechanisms, which allows hosts to generate additional traffic that can be used to obfuscate the traffic patterns and make it more difficult for adversaries to analyze the traffic.

The main goal is to reinforce Tor's resistance against traffic analysis attacks, specially fingerprinting and correlation, therefore fortifying users' privacy and anonymity, and ensuring that the solution maintains reasonable performance and usability. To achieve this goal, we designed TTT with the following design goals in mind:

Privacy and Anonymity: The solution must reinforce Tor's resistance against traffic analysis attacks, specifically fingerprinting and correlation attacks.

Formally Proven: The solution must be formally proven to provide a certain level of privacy, by using Differential Privacy. This way, we add a new and important layer of security and trust to the Tor network and users.

Tor's Extension: The solution must be an extension of the Tor project, respecting Tor's design principles and rules. As a very popular open source project, Tor has a large community of users and developers, and it is important to ensure that the solution can be easily integrated into the existing Tor source code. The project contains a set of principles and rules that we must respect.

Compactibility: The solution must be easily integrated into the existing Tor network, allowing users and hosts to configure the trade-offs between privacy and performance. The Tor network is composed of several voluntary relays, where hosts decide how to configure. Therefore, we must ensure that the solution can be easily integrated into the existing Tor network and compactible with relays that may be unaware of our solution.

Unobservability: The solution must enhance Tor's resistance against traffic analysis attacks, specifically fingerprinting and correlation attacks, in comparison to the existing Tor network.

Configurability: As an extension of the Tor project, the solution must use the existing Tor configuration files and allow users and hosts to configure the trade-offs between privacy and performance. This way, users can choose the level of privacy they want to achieve.

Performance: We consider that performance is a key aspect of the solution. However, the solution does not aim to improve the performance of the Tor network, but rather to ensure that the solution does not significantly impact the performance of the Tor network.

3.2 System Model

As stated previously, the solution is an extension of the Tor project, so the system model is also similar to the Tor system model, as briefly explained in Section 2.2. In order to preserve the Tor network's design principles and rules, our system model does not

introduce any new components or mechanisms in the network layer of the system, as illustrated in Figure 3.1.

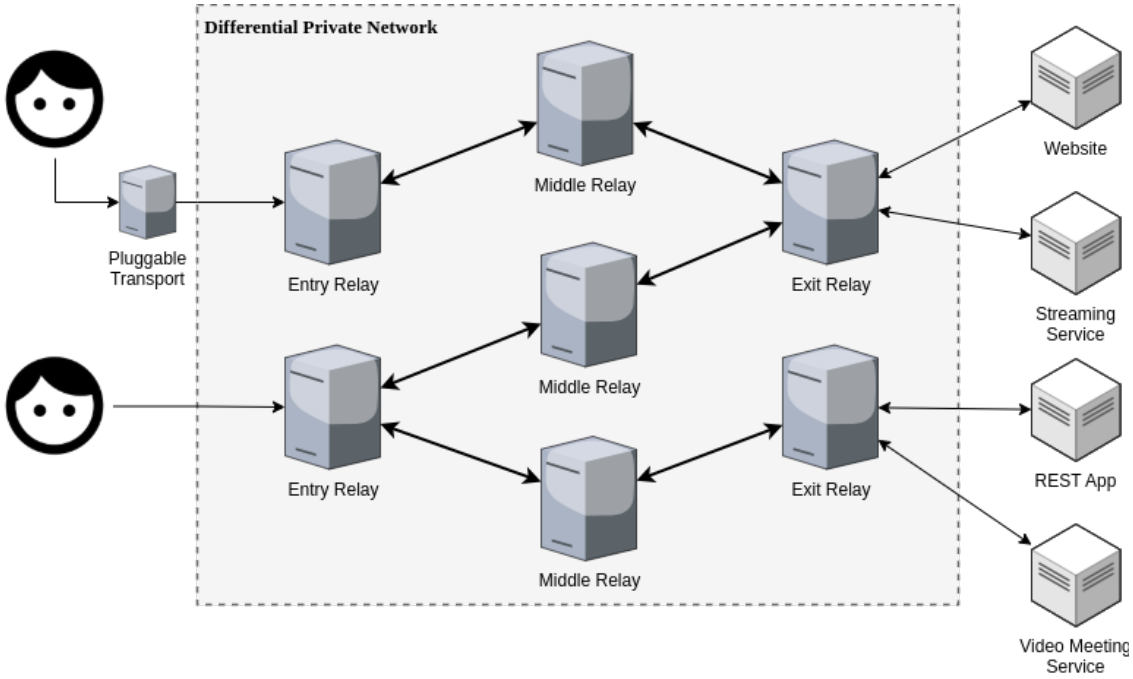


Figure 3.1: Differential Private Network System Model

Instead, it focuses on enhancing the scheduling on existing Tor relays, with the proposed Differential Privacy Tor Cell Scheduler, and on introducing a new mechanism for generating additional traffic, called 'Packet Padding Cells' mechanism, which are both presented further below.

Differential Private Tor Cell Scheduler: A new scheduler for Tor cells that incorporates Differential Privacy techniques to apply jitter to the Tor network traffic, by applying, or not, a delay on the decision of when the scheduler will run again. Tor's schedulers run on a fix and predetermined time. By setting a random delay on the decision of when the scheduler will run again, we can therefore apply jitter to the Tor network traffic, making it more difficult for adversaries to analyze the traffic patterns and de-anonymize users. As each scheduler can or not apply a delay, the jitter is applied by each individual relay and/or client, making the differential privacy properties accumulate across the network, therefore providing a stronger privacy guarantee. Additionally, this property of the scheduler allows the network to be composed by nodes that are unaware of the solution.

Packet Padding Cells (PPC): A new mechanism for generating additional traffic. The Tor cells are generated based on a decision-making differential private mechanism triggered when the relay receives a new Tor 'relay' cell. The newly generated cells are added to the circuit cell queue of the original cell, which leads to the generated traffic to only be added to active and established circuits. In comparison to the above scheduler, the differential

private properties do not accumulate across the network, as the cells are generated only when a new Tor cell is received, and discarded on reception. This way, considering that 2 segments of the network have senders enhanced with our proposed PPC mechanism, these 2 segments will have a different number of TLS packets and/or different sized TLS packets, making more difficult for adversaries to correlate both segments. Additionally, the client does not produce any additional traffic, unlike the scheduler.

These two new features are designed to work independently, meaning that the user can choose to use one or both of them, but with the same goal of enhancing the Tor network's resistance against traffic analysis attacks.

3.3 Threat Model

As mentioned earlier, Tor network is widely used anonymity network that may be vulnerable to traffic analysis attacks, such as fingerprinting and traffic correlation. With these types of attacks, adversaries analyze and monitor traffic in certain segments of the network, aiming to gather information to train and strengthen their machine and deep learning models, which can then be used to de-anonymize users and expose their browsing activities. This way, users that browse the internet through the Tor network, to protect their anonymity, privacy and security, may be vulnerable to these attacks, which can lead to the exposure of their browsing activities and identities, and making Tor inefficient.

TTT is designed to strengthen the Tor network's privacy protections and bolster its defenses against traffic analysis attacks, with a particular focus on the before mentioned threats. In shaping our threat model, we closely follow the foundational Tor threat model outlined by [14]. Accordingly, we consider adversaries who can monitor portions of network traffic, manipulate data flows, operate their own onion routers, and compromise a fraction of existing routers. The adversary's ultimate objective is the de-anonymization of users and the exposure of their browsing activities.

Before defining our adversary model, we first clarify the types of adversaries considered and which are not included in our threat model. State-Level adversaries (SLAd) are capable of observing Tor's traffic patterns in one or few Autonomous Systems (AS). Large-Scope adversaries (LSAd) are considered a group of SLAd that can monitor a significant fraction of the network, by collaborating and working together and therefore controlling the collection of regions controlled by each collaborator. Finally, Omnipresent adversaries (OPAd) are a powerful attacker, or a group of attackers, that can monitor the entire Tor network, including all segments and regions, and therefore can observe all traffic patterns.

Figure 3.2 illustrates the threat model over the system model, showing the adversaries' capabilities and the Tor network segments that they can monitor. The red magnifying glasses indicate the more commonly observed segments of the network, whilst the orange magnifying glasses represent the segments that are less commonly observed. Nonetheless, it is important to note that the adversaries considered by our threat model are not limited

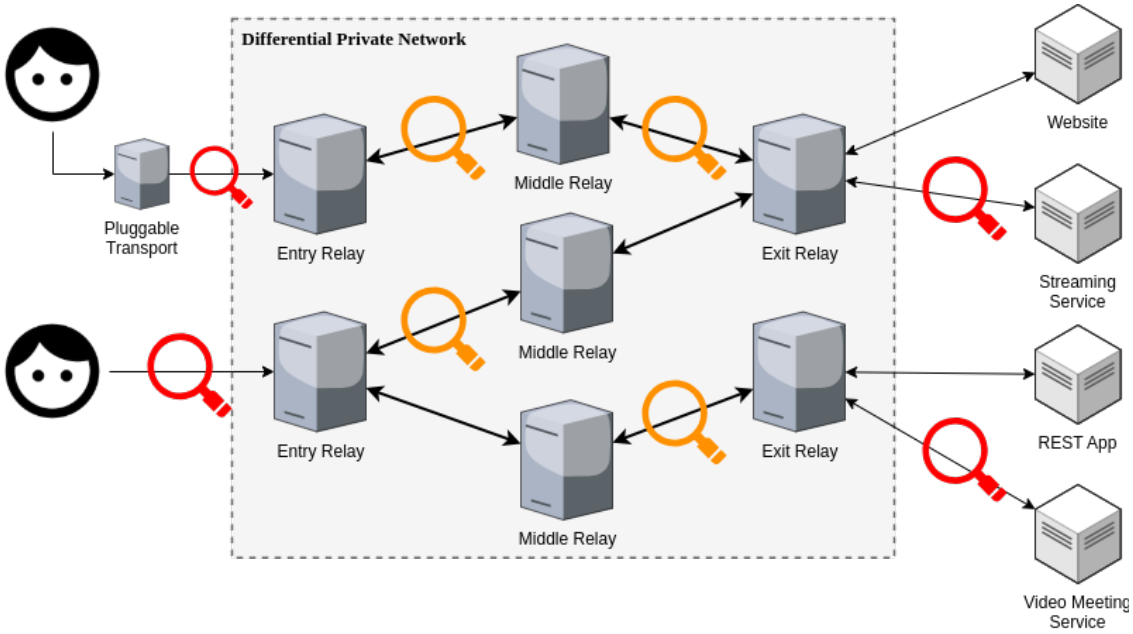


Figure 3.2: Threat Model over System Model

to the firstly described segments and our adversaries can monitor any pair of segments in the network.

For the scope of our solution, we considered SLAd to be the main adversary, as they are more common and realistic in the context of Tor. LSAd are also considered to be a threat and also included in the scope of our threat model, even though we recognize that they are less common. On the other hand, OPAd are not considered in our threat model, as we consider they are not realistic and do not represent a common threat to Tor users.

Building on the original assumptions put forth by Dingledine, Mathewson, and Syverson [14], we consider both passive and active adversaries. A passive adversary merely observes traffic patterns, applying machine learning and data analysis techniques to infer user identities and behaviors. Conversely, an active adversary compromises onion routers to inject traffic into the network — not merely to observe, but to facilitate more sophisticated traffic analysis attacks. Traffic injection, in this context, becomes a strategic tool for the adversary to train machine learning models capable of uncovering users' identities and tracking their navigation through the Tor network. However, it is important to note that we do not take relay compromise into account. In some context, adversaries may deploy a malicious voluntary relay and our solution is not intended to defend users, as the attacker would be able to monitor the traffic patterns and defeat the Packet Padding Cells mechanism.

In summary, our threat model considers powerful State-Level and Large-Scope adversaries capable of monitoring significant portions of the Tor network traffic, but not on its entirety. These adversaries are equipped with advanced and state-of-the-art machine and deep learning and data analysis techniques. These adversaries may be passive, observing traffic patterns, or active by injecting traffic into the network to train their machine learning

models.

3.4 Software Architecture

To ensure a clear understanding of the software architecture of TTT, we will first present the Tor relay architecture, which is the foundation of TTT, and then we will present the TTT architecture, as an extension of the Tor relay architecture.

3.4.1 Tor Relay Architecture

Tor relays are the backbone of the Tor network, responsible for routing traffic between clients and servers while maintaining user anonymity. As explained in more detailed in section 2.2, Tor relays are connected through TCP over TLS connections, forming a network of relays that route TLS packets composed by a variable number of cells between clients, relays and servers, encrypting and decrypting the traffic at each hop, layer by layer.

Our solution extends the Tor relay architecture by introducing a new scheduler and a new mechanism for generating additional traffic, presented in section 3.2. At the time we proposed, Tor has 2 types of schedulers: the 'KIST' scheduler, briefly presented in subsubsection 2.2.3.2, which also has a 'Lite-KIST' variant for low-end devices, and the Vanilla scheduler, which is a simpler scheduler. To develop our solution, we chose to extend the Vanilla scheduler, as it is simpler and easier to understand, and therefore easier to extend.

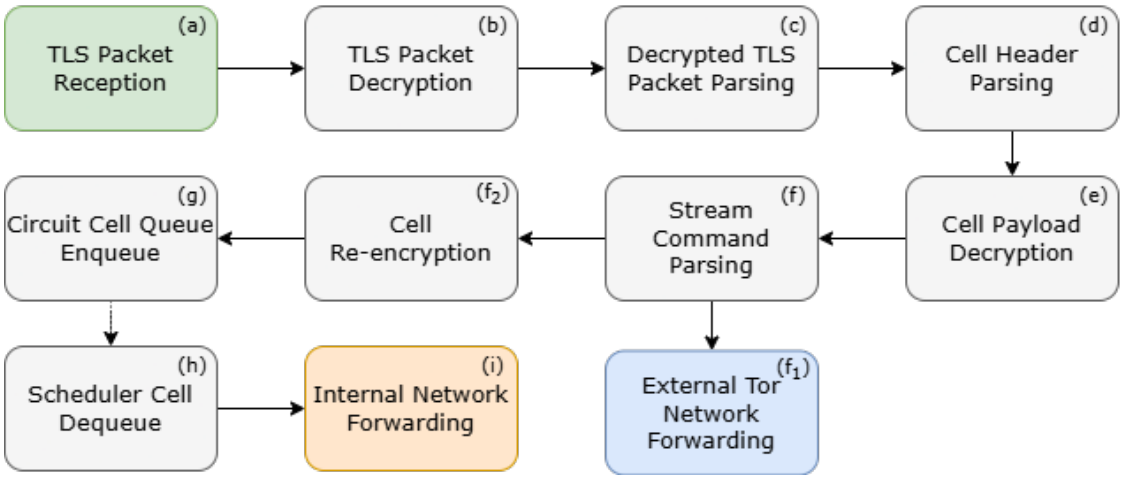


Figure 3.3: Original Cell Processing Pipeline

To ease the understanding of the Tor relay architecture, we present an illustration of the lifetime of a Tor cell in a Tor relay in Figure 3.3. As shown in the figure, after receiving a TLS packet (a), it gets decrypted (b) and all Tor Cells within the packet are extracted (c). For each Tor Cell within the packet, its header is checked (d) and its payload gets decrypted (e). After decrypting the payload, the cell's 'stream command' is used to determine if the relay is the exit and the cell must be forwards to the destination (f_1), or if the relay

corresponds to an entry or middle relay, in which case the cell is forwarded to the next hop. Finally, the cells that must be forwarded are re-encrypted (f_2) and enqueued in the respective circuit cell queue (g). The scheduler then dequeues the cells from the circuit cell queue (h), and forwards them to the next hop (i). This process is repeated until the cell reaches its destination, which can be a client or a server.

3.4.2 TTT Architecture

After presenting the Tor relay architecture, together with a better understanding of the Tor cell lifetime in a relay, we can now present the TTT architecture. As stated in section 3.2, our solution brings 2 main features to the Tor project: a new scheduler that applies jitter to the Tor network traffic, and a new mechanism for generation additional traffic.

To allow a better comprehension and comparison to the differences of Tor cells lifetime in the original Tor project and in TTT, we present a similar illustration to Figure 3.3 in Figure 3.4. The figure illustrates the lifetime of a Tor cell in a TTT relay, showing the differences introduced by our solution.

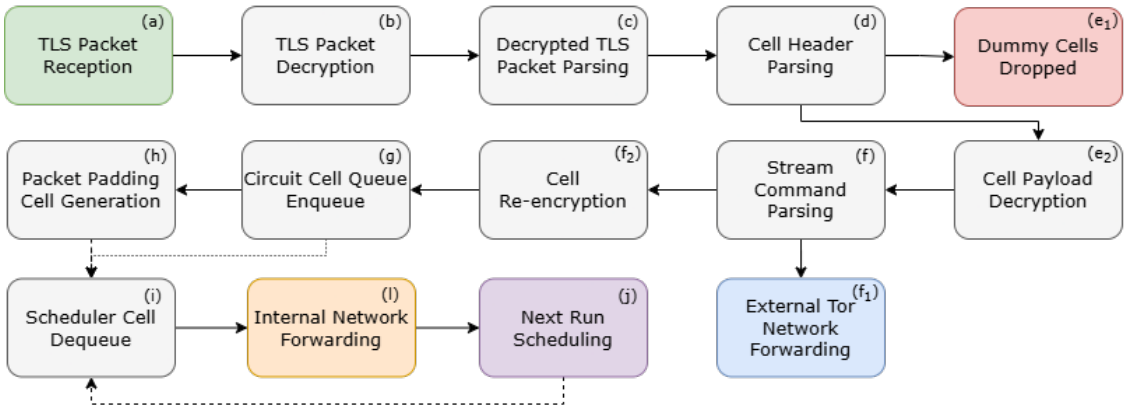


Figure 3.4: TTT Cell Processing Pipeline

Upon receiving a TLS packet (a), the relay decrypts it (b) and extracts all cells within the packet (c), same as the original Tor. For each cell within the packet, the relay checks its header (d). If the cell is a 'dummy' cell, it is discarded (e_1), otherwise the cell is decrypted (e_2). After decrypting the cell, the 'stream command' is checked to determine if the relay is the exit and the cell must be forwarded to the destination (f_1), or if the relay corresponds to an entry or middle relay, in which case the cell is forwarded to the next hop. For the cells that must be forwarded, they are re-encrypted (f_2) and enqueued in the respective circuit cell queue (g). After queuing a cell, the relay decides, based on a differential private algorithm, if it generates a new 'Packet Padding Cell' (h). Newly generated cells are added to the circuit cell queue of the original cell. Once any cell is added to the circuit cell queue, the circuit is marked as 'writable', allowing the scheduler to dequeue the cells from the circuit cell queue (i). After being dequeued cells, the cells are places in the output buffer for retransmission, and forwards the cells to the next hop (l). Meanwhile, the scheduler

calculates if jitter must be applied to the circuit, therefore setting a random delay on the decision of when it will run again or not (j), impacting the following cells time of retransmission.

In comparison to the original Tor cell lifetime, the main differences are the step (e_1), where the dummy cells are discarded, the step (i), where the process of decision and generation of Packet Padding Cells take place, and the step (m), where the scheduler decides if it will apply jitter to the circuit or not. The scheduler implementation of jitter is originated by scheduling the next time the scheduler will run again based on a random delay, which is calculated based on the differential private algorithm. This way, the scheduler can apply jitter to the Tor network traffic and be non-blocking to other relay processes essential for the Tor software to work properly.

3.5 Parameterization

3.5.1 Used Techniques

3.5.2 Jitter Parameterization

3.5.3 Packet Padding Cells Parameterization

3.6 Jitter Development Strategy

3.7 Packet Padding Cells Development Strategy

3.8 Discussion

4

TTT Prototype and Implementation

4.1 Implementation Guidelines and Techniques

4.2 Implementation Model

4.3 Prototyping Metrics

4.4 Complexity Analysis

4.5 Prototype Availability

4.6 Summary

5

Validation and Experimental Evaluation

5.1 Evaluation Criteria

5.1.1 Testbenchs

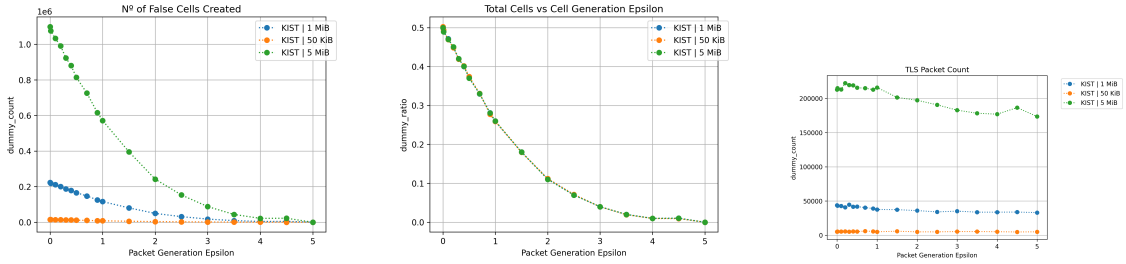
5.1.2 Experimental Observations

5.2 Performance Evaluation

5.2.1 Packet Padding Cells

To test the performance of the Packet Padding Cells (PPCs) generation, we downloaded a file from a web server using curl. The tests were performed 100 times, with different ϵ values, ranging from 0 to 10.

To show the impact of the PPCs generation, we also measured the number of PPCs generated, shown in figure **??**, the ration of PPCs to total cells, shown in figure 5.1b, and the total number of TLS packets, shown in figure 5.1c.



(a) Total Packet Padding Cells (b) Packet Padding Cells Ratio (c) Total TLS Packets

Figure 5.1: Packet Padding Cells Generation and Impact

Thee lower the ϵ, more PPCs are generated and the higher the ratio of PPCs to total cells. However, it is important to note that the number of total TLS packets does not correspond to the cells graphic. This is because each TLS packet can contain multiple cells,

and the number of cells is not directly proportional to the number of packets.

5.2.1.1 Latency

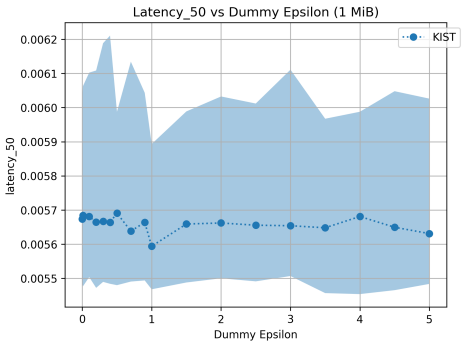


Figure 5.2: Download Latency with Packet Padding Cells Generation

The latency keeps in the same range across all tested ϵ with higher percentile 90 values near lower values. The median latency of all tests concentrates between 0.0056 and 0.0057 seconds.

5.2.1.2 Throughput

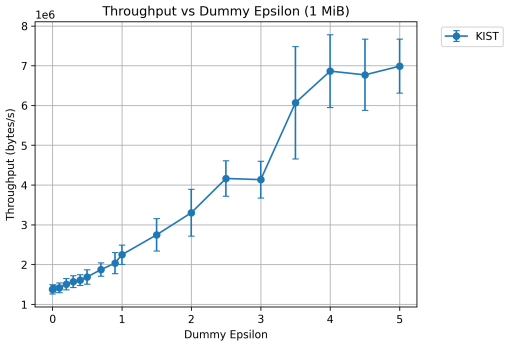


Figure 5.3: Download Throughput with Packet Padding Cells Generation

The throughput gets higher as the ϵ increases, as expected, and the standard deviation also increases.

5.2.1.3 Total Time

The total time needed to download the file decreases as the ϵ gets higher, as expected. The standard deviation keeps similar values across all tests.

5.2.2 Jitter Injector Tor Schedulers

To evaluate the performance of the new implemented Priv_KIST and Priv_Vanilla Tor Schedulers, we downloaded a file from a web server using curl. The tests were performed

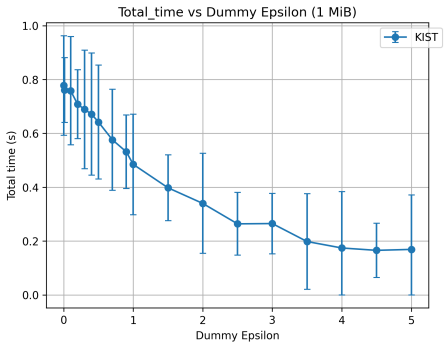
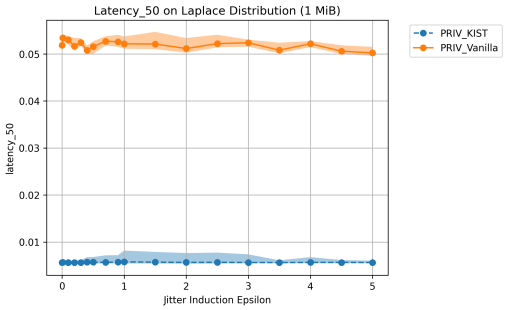


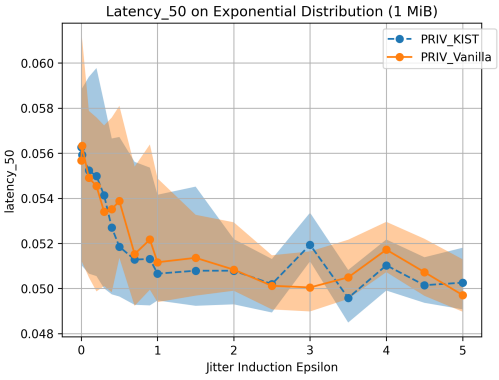
Figure 5.4: Download Total Time with Packet Padding Cells Generation

100 times, with different schedulers, with variable ϵ values, ranging from 0 to 10 and using 1 of the following mathematical distributions to generate jitter: Laplace, Poisson and Exponential. The file size was set to 1 MiB.

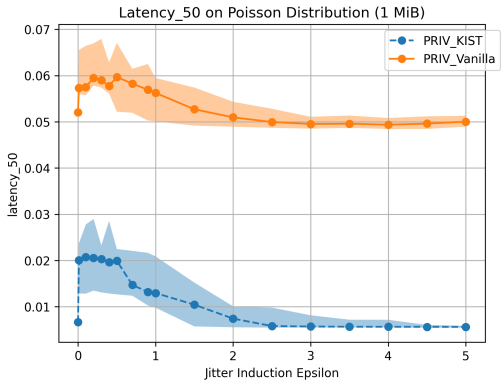
5.2.2.1 Latency



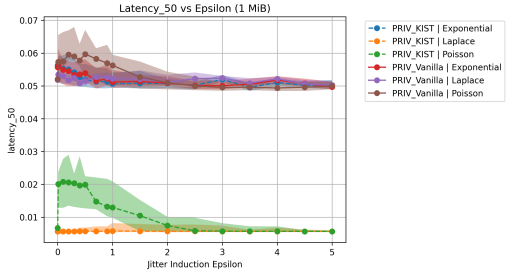
(a) Laplace Distribution



(b) Exponential Distribution



(c) Poisson Distribution



(d) All Distributions

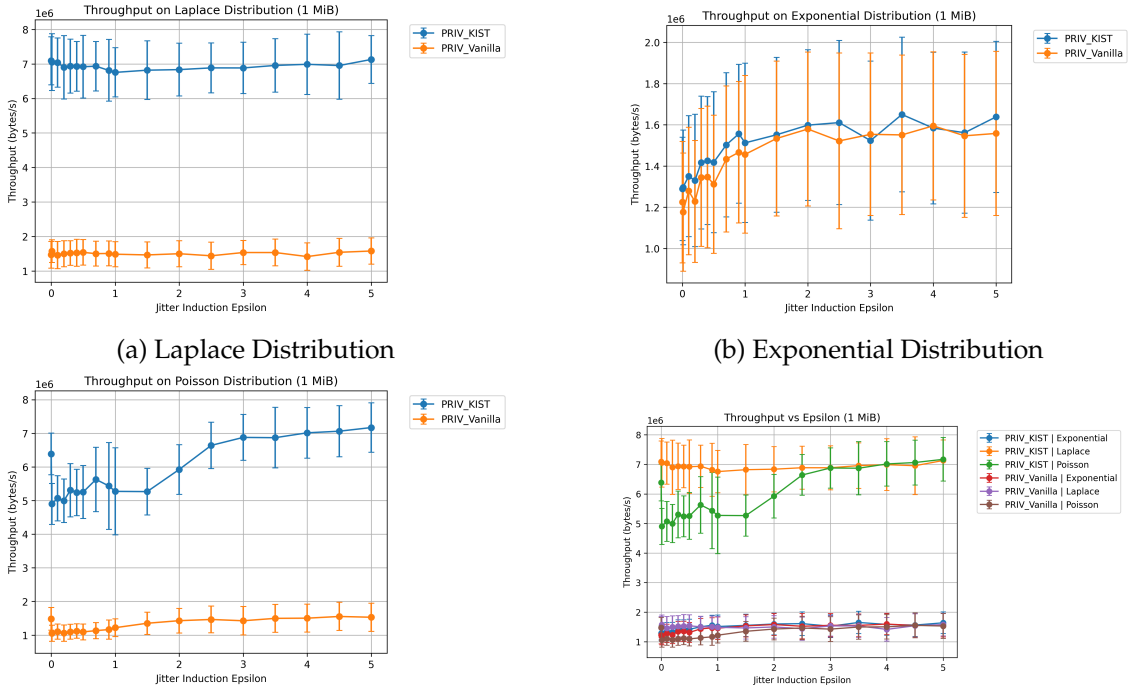
Figure 5.5: Tor Schedulers and Jitter Impact on Latency

The Laplace Distribution tests show that latency is not affected by the injected jitter, as the median latency remains constant across all tests. The Poisson Distribution gets

higher median latency values as the ϵ comes closer to 0 which then decreases and remains constant for ϵ values higher than 2,5. The Exponential Distribution shows a similar behavior to the Poisson Distribution, but shows a more pronounced increase in median latency as the ϵ approaches 0, and a more variable behavior throughout the tests.

In comparison, regarding latency, the Laplace Distribution is the most stable, while the Poisson and Exponential Distributions show more variability in the results. The PRIV_KIST scheduler performs better, with lower median latency values than the PRIV_Vanilla scheduler, even thought the Exponential Distribution shows similar median latency values for both schedulers.

5.2.2.2 Throughput



(a) Laplace Distribution

(b) Exponential Distribution

(c) Poisson Distribution

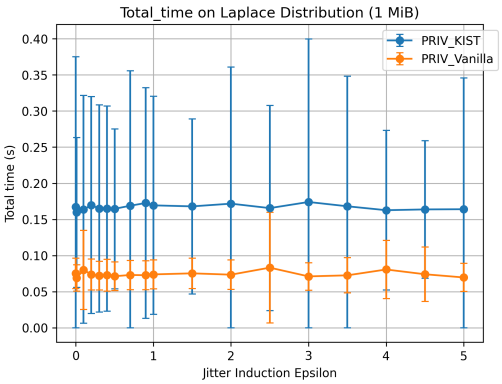
(d) All Distributions

Figure 5.6: Tor Schedulers and Different Distributions Impact on Throughput

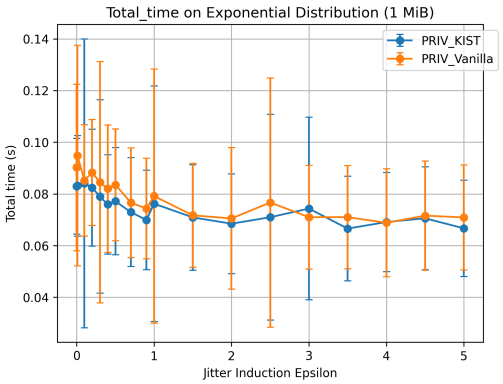
Similarly to the latency, the Laplace Distribution shows a stable throughput across all tests, PRIV_KIST having higher throughput values than PRIV_Vanilla. The Poisson Distribution shows a decrease in throughput as the ϵ approaches 0, with a more pronounced decrease for the PRIV_KIST scheduler, while having higher values than the PRIV_Vanilla scheduler. The Exponential Distribution also shows a decrease in throughput as the ϵ approaches 0 for both schedulers and a higher variability in the results. For this distribution, the throughput values are very similar for both schedulers.

In comparison, the Laplace Distribution is the most stable and the PRIV_KIST scheduler performs better than the PRIV_Vanilla scheduler.

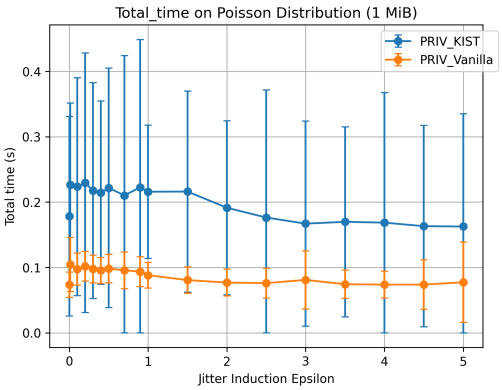
5.2.2.3 Total Time



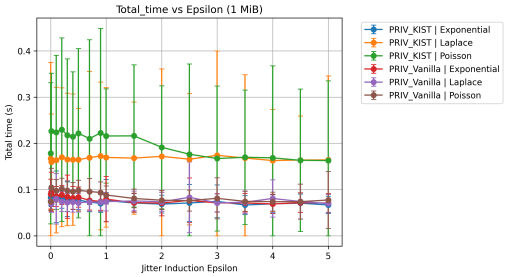
(a) Laplace Distribution



(b) Exponential Distribution



(c) Poisson Distribution



(d) All Distributions

Figure 5.7: Tor Schedulers and Total Time For Different Distributions

The Laplace and Poisson Distributions show a stable total time across all tests, with the PRIV_Vanilla scheduler having lower total time values than the PRIV_KIST scheduler. The Exponential Distribution also shows a decrease in total time as the ϵ increases for both schedulers and a higher variability in the results. For this distribution, the total time values are very similar for both schedulers.

In comparison, all distributions show similar results, in a range of 0.05 to 0.25 seconds, with the Laplace Distribution being the most stable. The PRIV_Vanilla scheduler performs better than the PRIV_KIST scheduler for the Laplace and Poisson Distributions, while both schedulers show similar results for the Exponential Distribution.

5.3 Unobservability Evaluation

5.3.1 Methods and Tools

5.3.2 Experimental Observations

5.4 Formal Validation

5.5 Discussion

5.6 Summary

6

Conclusions

6.1 Main Contributions

6.2 Future Work

Bibliography

[1] L. von Ahn, A. Bortz, and N. J. Hopper. "k-anonymous message transmission". In: *Proceedings of the 10th ACM Conference on Computer and Communications Security*. CCS '03. Washington D.C., USA: Association for Computing Machinery, 2003, 122–130. ISBN: 1581137389. DOI: 10.1145/948109.948128. URL: https://doi.org/10.1145/948109.948128 (cit. on p. 14).

[2] M. Alimardani and S. Milan. "The Internet as a Global/Local Site of Contestation: The Case of Iran". In: *Global Cultures of Contestion*. Ed. by D. T. Della and G. K. Thomas. Springer, 2018, pp. 171–192. URL: https://link.springer.com/chapter/10.1007/978-3-319-78889-7_9 (cit. on pp. 1, 3).

[3] S. Aryan, H. Aryan, and J. A. Halderman. "Internet Censorship in Iran: A First Look". In: *Proceedings of the 3rd USENIX Workshop on Free and Open Communications on the Internet (FOCI 13)*. USENIX Association, 2013. URL: https://www.usenix.org/conference/foci13/workshop-program/presentation/aryan (cit. on pp. 1, 3).

[4] L. Barman et al. "Groove: Flexible Metadata-Private Messaging". In: *16th USENIX Symposium on Operating Systems Design and Implementation (OSDI 22)*. Carlsbad, CA: USENIX Association, 2022-07, pp. 735–750. ISBN: 978-1-939133-28-1. URL: https://www.usenix.org/conference/osdi22/presentation/barman (cit. on p. 6).

[5] D. Barradas et al. "Poking a Hole in the Wall: Efficient Censorship-Resistant Internet Communications by Parasitizing on WebRTC". In: *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM, 2020, pp. 35–48. DOI: 10.1145/3372297.3423345. URL: https://doi.org/10.1145/3372297.3423345 (cit. on p. 2).

[6] A. Campan and T. Truta. "Data and Structural k-Anonymity in Social Networks". In: vol. 5456. 2008-01, pp. 33–54. ISBN: 978-3-642-01717-9. DOI: 10.1007/978-3-642-01718-6_4 (cit. on p. 15).

[7] B. Carvalho. "TIRDEANON. DEANONYMIZATION AND TRAFFIC-UNOBSERVABILITY ANALYSIS OF TIR AS A STRENGTHENED SOLUTION TO IMPROVE ANONYMITY GUARANTEES IN TOR". MA thesis. NOVA School of Science and Technology, 2023 (cit. on p. 1).

[8] S. Chakravarty et al. "On the Effectiveness of Traffic Analysis against Anonymity Networks Using Flow Records". In: *Passive and Active Measurement*. Ed. by M. Faloutsos and A. Kuzmanovic. Cham: Springer International Publishing, 2014, pp. 247–257 (cit. on pp. 1–3, 13).

[9] D. Chaum. "Untraceable Electronic Mail, Return Addresses and Digital Pseudonyms". In: *Secure Electronic Voting*. Ed. by D. A. Gritzalis. Boston, MA: Springer US, 2003, pp. 211–219. ISBN: 978-1-4615-0239-5. DOI: 10.1007/978-1-4615-0239-5_14. URL: https://doi.org/10.1007/978-1-4615-0239-5_14 (cit. on p. 5).

[10] G. Cherubin, R. Jansen, and C. Troncoso. "Online Website Fingerprinting: Evaluating Website Fingerprinting Attacks on Tor in the Real World". In: *31st USENIX Security Symposium (USENIX Security 22)*. Boston, MA: USENIX Association, 2022-08, pp. 753–770. ISBN: 978-1-939133-31-1. URL: https://www.usenix.org/conference/usenixsecurity22/presentation/cherubin (cit. on pp. 1, 2).

[11] H. Corrigan-Gibbs, D. Boneh, and D. Mazieres. "Riposte: An anonymous messaging system handling millions of users". In: *Proceedings of the IEEE Symposium on Security and Privacy (S&P)*. 2015, pp. 321–338 (cit. on p. 2).

[12] G. Danezis, R. Dingledine, and N. Mathewson. "Mixminion: design of a type III anonymous remailer protocol". In: *2003 Symposium on Security and Privacy, 2003.* 2003, pp. 2–15. DOI: 10.1109/SECPRI.2003.1199323 (cit. on p. 6).

[13] G. Danezis. "Statistical Disclosure Attacks". In: *Security and Privacy in the Age of Uncertainty*. Ed. by D. Gritzalis et al. Boston, MA: Springer US, 2003, pp. 421–426. ISBN: 978-0-387-35691-4 (cit. on p. 2).

[14] R. Dingledine, N. Mathewson, and P. Syverson. "Tor: The Second-Generation Onion Router". In: *Proceedings of the 13th USENIX Security Symposium*. USENIX Association, 2004, pp. 303–320. URL: https://www.usenix.org/conference/usenixsecurity04/tor-second-generation-onion-router (cit. on pp. 2, 3, 6, 9, 13, 23, 24).

[15] C. Dwork. "Differential privacy". In: *Proceedings of the 33rd International Conference on Automata, Languages and Programming - Volume Part II*. ICALP'06. Venice, Italy: Springer-Verlag, 2006, 1–12. ISBN: 3540359079. DOI: 10.1007/11787006_1. URL: https://doi.org/10.1007/11787006_1 (cit. on pp. 2, 15).

[16] C. Dwork and A. Roth. "The Algorithmic Foundations of Differential Privacy". In: *Found. Trends Theor. Comput. Sci.* 9.3–4 (2014-08), 211–407. ISSN: 1551-305X. DOI: 10.1561/0400000042. URL: https://doi.org/10.1561/0400000042 (cit. on pp. 15–17).

[17] C. Dwork et al. "Calibrating noise to sensitivity in private data analysis". In: *Proceedings of the Third Conference on Theory of Cryptography*. TCC'06. New York, NY: Springer-Verlag, 2006, 265–284. ISBN: 3540327312. DOI: 10.1007/11681878_14. URL: https://doi.org/10.1007/11681878_14 (cit. on pp. 2, 15, 16).

[18] Úlfar Erlingsson, V. Pihur, and A. Korolova. "RAPPOR: Randomized Aggregatable Privacy-Preserving Ordinal Response". In: *Proceedings of the 21st ACM Conference on Computer and Communications Security*. Scottsdale, Arizona, 2014. URL: https://arxiv.org/abs/1407.6981 (cit. on p. 17).

[19] G. Figueira, D. Barradas, and N. Santos. "Stegozoa: Enhancing WebRTC Covert Channels with Video Steganography for Internet Censorship Circumvention". In: *Proceedings of the 2022 ACM on Asia Conference on Computer and Communications Security*. ASIA CCS '22. Nagasaki, Japan: Association for Computing Machinery, 2022, 1154–1167. ISBN: 9781450391405. DOI: 10.1145/3488932.3517419. URL: https://doi.org/10.1145/3488932.3517419 (cit. on p. 2).

[20] D. Goldschlag, M. Reed, and P. Syverson. "Hiding Routing Information". In: 1996-08. ISBN: 978-3-540-61996-3. DOI: 10.1007/3-540-61996-8_37 (cit. on pp. 5, 6).

[21] A. Gosain and N. Chugh. "Privacy Preservation in Big Data". In: *International Journal of Computer Applications* 100 (2014-08), pp. 44–47. DOI: 10.5120/17619-8322 (cit. on p. 15).

[22] J. van den Hooff et al. "Vuvuzela: scalable private messaging resistant to traffic analysis". In: *Proceedings of the 25th Symposium on Operating Systems Principles*. SOSP '15. Monterey, California: Association for Computing Machinery, 2015, 137–152. ISBN: 9781450338349. DOI: 10.1145/2815400.2815417. URL: https://doi.org/10.1145/2815400.2815417 (cit. on p. 6).

[23] N. Hopper and E. Y. Vasserman. "On the effectiveness of k;-anonymity against traffic analysis and surveillance". In: *Proceedings of the 5th ACM Workshop on Privacy in Electronic Society*. WPES '06. Alexandria, Virginia, USA: Association for Computing Machinery, 2006, 9–18. ISBN: 1595935568. DOI: 10.1145/1179601.1179604. URL: https://doi.org/10.1145/1179601.1179604 (cit. on pp. 2, 15).

[24] M. Horta. "TOR K-ANONYMITY AGAINST DEEP LEARNING WATERMARKING ATTACKS: validating a Tor k-Anonymity input circuit enforcement against a deep learning watermarking attack". MA thesis. NOVA School of Science and Technology, 2022-10 (cit. on p. 1).

[25] A. Houmansadr et al. "IP over Voice-over-IP for censorship circumvention". In: *CoRR* abs/1207.2683 (2012). arXiv: 1207.2683. URL: http://arxiv.org/abs/1207.2683 (cit. on p. 2).

[26] A. Inc. *Learning with Privacy at Scale*. URL: https://machinelearning.apple.com/research/learning-with-privacy-at-scale (cit. on p. 17).

[27] M. Jakobsson. "Flash mixing". In: *Proceedings of the Eighteenth Annual ACM Symposium on Principles of Distributed Computing*. PODC '99. Atlanta, Georgia, USA: Association for Computing Machinery, 1999, 83–89. ISBN: 1581130996. DOI: 10.1145/301308.301333. URL: https://doi.org/10.1145/301308.301333 (cit. on p. 5).

[28] M. Jakobsson and A. Juels. "An optimally robust hybrid mix network". In: *Proceedings of the Twentieth Annual ACM Symposium on Principles of Distributed Computing*. PODC '01. Newport, Rhode Island, USA: Association for Computing Machinery, 2001, 284–292. ISBN: 1581133839. DOI: 10.1145/383962.384046. URL: https://doi.org/10.1145/383962.384046 (cit. on p. 5).

[29] R. Jansen and N. Hopper. "Shadow: Running Tor in a Box for Accurate and Efficient Experimentation". In: *Network and Distributed System Security Symposium*. 2012 (cit. on p. 12).

[30] R. Jansen, T. Vaidya, and M. Sherr. "Point Break: A Study of Bandwidth Denial-of-Service Attacks against Tor". In: *28th USENIX Security Symposium (USENIX Security 19)*. Santa Clara, CA: USENIX Association, 2019-08, pp. 1823–1840. ISBN: 978-1-939133-06-9. URL: https://www.usenix.org/conference/usenixsecurity19/presentation/jansen (cit. on pp. 1–3).

[31] R. Jansen et al. "KIST: Kernel-Informed Socket Transport for Tor". In: *ACM Trans. Priv. Secur.* 22.1 (2018-12). ISSN: 2471-2566. DOI: 10.1145/3278121. URL: https://doi.org/10.1145/3278121 (cit. on pp. 3, 11, 12).

[32] A. Jerichow et al. "Real-time mixes: a bandwidth-efficient anonymity protocol". In: *IEEE J.Sel. A. Commun.* 16.4 (2006-09), 495–509. ISSN: 0733-8716. DOI: 10.1109/49.668973. URL: https://doi.org/10.1109/49.668973 (cit. on p. 6).

[33] A. Johnson et al. "Users Get Routed: Traffic Correlation on Tor by Realistic Adversaries". In: *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security (CCS)*. ACM, 2013. DOI: 10.1145/2508859.2516651. URL: https://doi.org/10.1145/2508859.2516651 (cit. on pp. 1, 3).

[34] D. Kesdogan, J. Egner, and R. Büschkes. "Stop- and- Go-MIXes Providing Probabilistic Anonymity in an Open System". In: vol. 1525. 1998-04, pp. 83–98. ISBN: 978-3-540-65386-8. DOI: 10.1007/3-540-49380-8_7 (cit. on p. 7).

[35] D. Lazar, Y. Gilad, and N. Zeldovich. "Karaoke: Distributed private messaging immune to passive traffic analysis". In: *Proceedings of the 13th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*. 2018, pp. 711–725 (cit. on pp. 2, 6).

[36] N. Mathewson and R. Dingledine. "Practical Traffic Analysis: Extending and Resisting Statistical Disclosure". In: *Privacy Enhancing Technologies*. Ed. by D. Martin and A. Serjantov. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 17–34. ISBN: 978-3-540-31960-3 (cit. on p. 2).

[37] S. Matic, C. Troncoso, and J. Caballero. "Dissecting Tor Bridges: A Security Evaluation of their Private and Public Infrastructures". In: *Network and Distributed System Security Symposium*. 2017. URL: https://api.semanticscholar.org/CorpusID:28430784 (cit. on p. 10).

[38] U. Moeller. "Mixmaster Protocol Version 2". In: 2004. URL: https://api.semanticscholar.org/CorpusID:215824742 (cit. on p. 6).

[39] M. Nasr, A. Bahramali, and A. Houmansadr. "DeepCorr: Strong Flow Correlation Attacks on Tor Using Deep Learning". In: *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*. Toronto, ON, Canada: ACM, 2018, pp. 1962–1976. DOI: 10.1145/3243734.3243858 (cit. on pp. 1, 13).

[40] J. P. Near and C. Abuah. *Programming Differential Privacy*. Vol. 1. 2021. URL: https://programming-dp.com/ (cit. on pp. 2, 15–17).

[41] N. Nipane, I. Dacosta, and P. Traynor. ""Mix-in-Place" anonymous networking using secure function evaluation". In: *Proceedings of the 27th Annual Computer Security Applications Conference*. ACSAC '11. Orlando, Florida, USA: Association for Computing Machinery, 2011, 63–72. ISBN: 9781450306720. DOI: 10.1145/2076732.2076742. URL: https://doi.org/10.1145/2076732.2076742 (cit. on p. 6).

[42] S. E. Oh et al. "DeepCoFFEA: Improved Flow Correlation Attacks on Tor via Metric Learning and Amplification". In: *2022 IEEE Symposium on Security and Privacy (SP)*. 2022, pp. 1915–1932. DOI: 10.1109/SP46214.2022.9833801 (cit. on pp. 1, 13).

[43] M. Ohkubo and M. Abe. "A Length-Invariant Hybrid Mix". In: *Advances in Cryptology - ASIACRYPT 2000, 6th International Conference on the Theory and Application of Cryptology and Information Security, Kyoto, Japan, December 3-7, 2000, Proceedings*. Vol. 1976. Lecture Notes in Computer Science. Springer, 2000, pp. 178–191. DOI: 10.1007/3-540-44448-3_14 (cit. on p. 5).

[44] H. Pereira. "MIRACE: Multi-Path Integrated Routing Architecture for Censorship Evasion". MA thesis. NOVA School of Science and Technology, 2024 (cit. on pp. 2, 8).

[45] A. M. Piotrowska et al. "The Loopix Anonymity System". In: *26th USENIX Security Symposium (USENIX Security 17)*. Vancouver, BC: USENIX Association, 2017-08, pp. 1199–1216. ISBN: 978-1-931971-40-9. URL: https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/piotrowska (cit. on pp. 2, 7).

[46] T. Project. *Circumvention*. URL: https://tb-manual.torproject.org/circumvention/ (cit. on pp. 2, 10).

[47] T. Project. *Tor Plugable Transports*. URL: https://2019.www.torproject.org/docs/pluggable-transports (cit. on pp. 2, 10).

[48] M. S. Rahman et al. "Tik-Tok: The Utility of Packet Timing in Website Fingerprinting Attacks". In: *Proceedings on Privacy Enhancing Technologies* 2020.3 (2020), pp. 5–24. DOI: 10.2478/popets-2020-0032 (cit. on pp. 1, 2).

[49] F. Rezaei and A. Houmansadr. "FINN: Fingerprinting Network Flows using Neural Networks". In: *Proceedings of the 37th Annual Computer Security Applications Conference* (2021). URL: https://api.semanticscholar.org/CorpusID:243797821 (cit. on p. 1).

[50] A. Sabzi et al. "NetShaper: A Differentially Private Network Side-Channel Mitigation System". In: *33rd USENIX Security Symposium (USENIX Security 24)*. Philadelphia, PA: USENIX Association, 2024-08, pp. 3385–3402. ISBN: 978-1-939133-44-1. URL: https://www.usenix.org/conference/usenixsecurity24/presentation/sabzi (cit. on pp. 2, 18).

[51] T. Shen et al. " DAENet: Making Strong Anonymity Scale in a Fully Decentralized Network ". In: *IEEE Transactions on Dependable and Secure Computing* 19.04 (2022-07), pp. 2286–2303. ISSN: 1941-0018. DOI: 10.1109/TDSC.2021.3052831. URL: https://doi.ieeecomputersociety.org/10.1109/TDSC.2021.3052831 (cit. on p. 2).

[52] Y. Shi and K. Matsuura. "Fingerprinting Attack on the Tor Anonymity System". In: *Information and Communications Security*. Ed. by S. Qing, C. J. Mitchell, and G. Wang. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 425–438 (cit. on p. 3).

[53] V. Shmatikov and M.-H. Wang. "Timing Analysis in Low-Latency Mix Networks: Attacks and Defenses". In: vol. 4189. 2006-01, pp. 18–33. ISBN: 978-3-540-44601-9. DOI: 10.1007/11863908_2 (cit. on pp. 2, 9).

[54] P. Sirinam et al. "Deep Fingerprinting: Undermining Website Fingerprinting Defenses with Deep Learning". In: *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*. CCS '18. Toronto, Canada: Association for Computing Machinery, 2018, 1928–1943. ISBN: 9781450356930. DOI: 10.1145/3243734.3243768. URL: https://doi.org/10.1145/3243734.3243768 (cit. on pp. 1, 2, 13).

[55] P. Sirinam et al. "Triplet Fingerprinting: More Practical and Portable Website Fingerprinting with N-shot Learning". In: *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*. CCS '19. London, United Kingdom: Association for Computing Machinery, 2019, 1131–1148. ISBN: 9781450367479. DOI: 10.1145/3319535.3354217. URL: https://doi.org/10.1145/3319535.3354217 (cit. on p. 13).

[56] Y. Sun et al. "RAPTOR: routing attacks on privacy in tor". In: *Proceedings of the 24th USENIX Conference on Security Symposium*. SEC'15. Washington, D.C.: USENIX Association, 2015, 271–286. ISBN: 9781931971232 (cit. on p. 13).

[57] L. Sweeney. "k-anonymity: a model for protecting privacy". In: *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.* 10.5 (2002-10), 557–570. ISSN: 0218-4885. DOI: 10.1142/S0218488502001648. URL: https://doi.org/10.1142/S0218488502001648 (cit. on pp. 2, 14).

[58] L. Sweeney. "Simple Demographics Often Identify People Uniquely". In: *Carnegie Mellon University, Data Privacy* (2000). URL: http://dataprivacylab.org/projects/identifiability/ (cit. on p. 14).

[59] C. Tang and I. Goldberg. "An improved algorithm for tor circuit scheduling". In: *Proceedings of the 17th ACM Conference on Computer and Communications Security*. CCS '10. Chicago, Illinois, USA: Association for Computing Machinery, 2010, 329–339. ISBN: 9781450302456. DOI: 10.1145/1866307.1866345. URL: https://doi.org/10.1145/1866307.1866345 (cit. on pp. 11, 12).

[60] N. Tyagi et al. "Stadium: A Distributed Metadata-Private Messaging System". In: *Proceedings of the 26th Symposium on Operating Systems Principles*. SOSP '17. Shanghai, China: Association for Computing Machinery, 2017, 423–440. ISBN: 9781450350853. DOI: 10.1145/3132747.3132783. URL: https://doi.org/10.1145/3132747.3132783 (cit. on pp. 2, 6, 8).

[61] A. Vilalonga, J. S. Resende, and H. Domingos. *TorKameleon: Improving Tor's Censorship Resistance with K-anonymization and Media-based Covert Channels*. 2023. arXiv: 2303.17544 [cs.CR]. URL: https://arxiv.org/abs/2303.17544 (cit. on p. 2).

[62] A. Vilalonga et al. "Algoritmos de Resposta Aleatória como Mecanismo para a Desvinculação entre o Tráfego de Entrada e de Saída em Sistemas de Anonimato Baseadas em Circuitos". In: *INForum* (2024). URL: https://www.inforum.pt/static/files/papers/INForum_2024_paper_77.pdf (cit. on pp. 2, 18).

[63] M. Wang et al. "Leveraging strategic connection migration-powered traffic splitting for privacy". In: *Proceedings on Privacy Enhancing Technologies* 2022.3 (2022-07), 498–515. ISSN: 2299-0984. DOI: 10.56553/popets-2022-0083. URL: http://dx.doi.org/10.56553/popets-2022-0083 (cit. on p. 2).

[64] S. L. Warner. "Randomized Response: A Survey Technique for Eliminating Evasive Answer Bias". In: *Journal of the American Statistical Association* 60.309 (1965). PMID: 12261830, pp. 63–69. DOI: 10.1080/01621459.1965.10480775. URL: https://www.tandfonline.com/doi/abs/10.1080/01621459.1965.10480775 (cit. on p. 17).

[65] Z. Weinberg et al. "StegoTorus: A camouflage proxy for the Tor anonymity system". In: 2012-10, pp. 109–120. DOI: 10.1145/2382196.2382211 (cit. on p. 2).

[66] P. Winter and S. Lindskog. "How the Great Firewall of China is Blocking Tor". In: *Proceedings of the 2nd USENIX Workshop on Free and Open Communications on the Internet (FOCI)*. USENIX Association, 2012. URL: https://www.usenix.org/conference/foci12/workshop-program/presentation/winter (cit. on pp. 1–3).

[67] D. I. Wolinsky et al. "Dissent in Numbers: Making Strong Anonymity Scale". In: *10th USENIX Symposium on Operating Systems Design and Implementation (OSDI 12)*. Hollywood, CA: USENIX Association, 2012-10, pp. 179–182. ISBN: 978-1-931971-96-6. URL: https://www.usenix.org/conference/osdi12/technical-sessions/presentation/wolinsky (cit. on p. 2).

[68] X. Zhang et al. "Statistical Privacy for Streaming Traffic". In: *Proceedings 2019 Network and Distributed System Security Symposium* (2019). URL: https://api.semanticscholar.org/CorpusID:142503765 (cit. on pp. 2, 18).

[69] Y. Zhu et al. "Anonymity analysis of mix networks against flow-correlation attacks". In: *GLOBECOM '05. IEEE Global Telecommunications Conference, 2005.* Vol. 3. 2005, 5 pp.–. DOI: 10.1109/GLOCOM.2005.1577959 (cit. on pp. 2, 9).

[70] J. L. Zittrain et al. "The Shifting Landscape of Global Internet Censorship". In: Accessed: 2024-11-17. Berkman Klein Center for Internet & Society, 2017. URL: https://www.ssrn.com/abstract=2993485 (cit. on pp. 1, 3).