CLAM: Client-Aware Routing in Mix Networks

Mahdi Rahimi
mahdi.rahimi@esat.kuleuven.be
COSIC, KU Leuven
Leuven, Belgium

ABSTRACT

Mix networks (mixnets) enhance anonymity at the cost of increased end-to-end latency, deterring clients from adopting mixnets for web browsing or instant messaging. This often leads clients to seek alternative anonymous communication systems, potentially compromising on anonymity levels. Addressing this, LARMix (NDSS 2024) introduced a strategic message routing aimed at minimizing link latency within mixnets. However, LARMix's proposal does not cover reducing link latency from clients to the mixnet. Filling this gap, CLAM presents innovative methodologies for efficient message forwarding from clients to mixnets. Our analysis reveals that clients using CLAM can reduce link latency to the mixnet by up to 90% without significantly burdening the network. Moreover, our results indicate that optimizing client routing in mixnets does not substantially increase the risk of message deanonymization, even with adversaries compromising up to 20% of nodes in the mixnet.

CCS CONCEPTS

• **Security and privacy → Information flow control**; • **Networks → Traffic engineering algorithms**.

KEYWORDS

Anonymity, Latency, Mix Networks

ACM Reference Format:
Mahdi Rahimi. 2024. CLAM: Client-Aware Routing in Mix Networks. In *Proceedings of the 2024 ACM Workshop on Information Hiding and Multimedia Security (IH&MMSec '24), June 24–26, 2024, Baiona, Spain.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3658664.3659631

1 INTRODUCTION

Mix networks (mixnets) [6, 8, 16, 21] play a crucial role in preserving user anonymity. They are specifically designed to thwart the surveillance efforts of global passive adversaries, who are capable of observing all network message traffic. Considering such adversaries, mixnets employ a network of intermediary nodes, known as mixnodes, which shuffle messages to effectively mask the connection between a message's sender and its recipient. This obfuscation can rely on various types of mixnodes, including threshold mixes [6], which accumulate messages until a preset count is reached before forwarding them, and pool mixnodes [9], which consider both a threshold time and a number of messages before forwarding the messages to the next hop. Additionally, Stop-and-Go Mixes (SG-mix) [12] introduce a random, exponentially distributed delay to each message before flushing them out. These methods for intentionally delaying messages and employing multi-hop routing, despite enhancing message anonymity, lead to increased end-to-end latency [7], posing a challenge to the usability of mixnets for real-time or latency-sensitive applications.

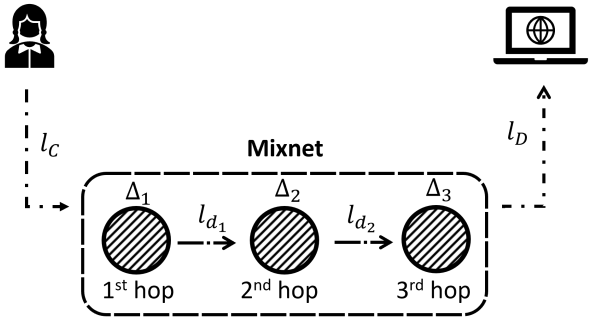


Figure 1: Components of End-to-End Latency: Consider an instance where a client routes messages through a mixnet comprising three hops. In this scenario, the total latency encountered is comprised of several elements, represented as $l_C + \Delta_1 + l_{d_1} + \Delta_2 + l_{d_2} + \Delta_3 + l_D$.

As a consequence, regardless of the topology employed [21], the mixnet introduces significant end-to-end latency for clients transmitting messages to their intended recipients. Fig. 1 illustrates a scenario in which a client aims to send a message through a mixnet comprising three hops. This example further delineates the components contributing to the overall end-to-end latency, primarily consisting of: the link delay between the client and the mixnet (l_C), the delays incurred through routing between mixnodes (l_{d_1} and l_{d_2}), the delay from the mixnet to the destination (l_D), and additional delays imposed by each mixnode for unlinkability purposes (Δ_1, Δ_2, and Δ_3). In contrast, a direct transmission would solely involve a latency component that, on average, is on par with the link delay l_D. This heightened latency introduced by mixnets can significantly impair the user experience, especially in time-sensitive applications such as web browsing, online messaging, and live streaming, compelling users to either opt for alternative anonymous communication methods with less deanonymization capability or forego anonymity altogether.

To mitigate the significant end-to-end latency prevalent in mix networks, LARMix [17] proposes a novel mixnode selection and message routing methodology. This approach capitalizes on the geographic diversity of mixnodes, aiming to minimize latency by

IH&MMSec '24, June 24–26, 2024, Baiona, Spain

ACM ISBN 979-8-4007-0637-0/24/06
https://doi.org/10.1145/3658664.3659631

favoring the selection of geographically proximate mixnodes for message routing. Initially, LARMix employs a diversification algorithm to strategically assign mixnodes from various jurisdictions to the mixnet in order to enhance the mixnet's geographic diversity.

Furthermore, LARMix introduces a routing mechanism that preferentially selects geographically closer mixnodes to reduce latency. This selection process is controlled by a parameter τ, where setting $\tau = 0$ specifically targets the nearest mixnodes for routing, significantly reducing the latency incurred by the link delays of intermediary mixnodes (e.g., l_{d_1} and l_{d_2} as illustrated in Fig. 1). Conversely, a $\tau = 1$ setting opts for a uniformly random selection of mixnodes, enhancing anonymity at the cost of increased latency. As a result, LARMix recommends adjusting the τ parameter to fine-tune the desired balance between minimizing latency and messages' anonymity within the mixnet.

While LARMix [17] effectively minimizes the latency associated with intermediary mixnodes, it presupposes that clients select the initial mixnode in a uniformly random manner. The underlying rationale for this design choice is to obfuscate the client's geographical location, ostensibly to enhance anonymity. Nonetheless, this approach presents a stark contrast to the scenario involving a global passive adversary, who is assumed capable of monitoring all communication links within the network. Given such an adversary's extensive surveillance capabilities, the manner in which the first mixnode is selected whether randomly or through a strategic approach does not inherently conceal the client's location from the adversary. This highlights a fundamental principle of mixnets: the essence of anonymity relies not on the initial selection of mixnodes but on the subsequent mixing process that occurs within each mixnode. However, in networks like Tor [10], client anonymity could be impacted by the choice of strategic routing. This suggests a potential oversight in LARMix's design, where the considerations applicable to mixnets may have been conflated with those pertinent to other anonymizing networks like Tor.

CLAM, in contrast to LARMix, focuses on reducing latency between clients and the mixnet, specifically targeting the initial link delay (l_C) as illustrated in Fig. 1. We introduce three innovative methods for selecting the initial mixnode, emphasizing the proximity of clients to the first hop in the mixnet. This approach is based on the premise that prioritizing proximity does not compromise client anonymity. Our strategies further aim to achieve two key objectives: ensuring low-latency connections and maintaining a relatively balanced load distribution across the entry hop of the mixnet.

Our empirical analysis demonstrates the effectiveness of these strategies in significantly reducing the latency between the client and the mixnet by up to 90%, with a minimal increase in message routing disproportionality (up to 4%) compared to uniformly selecting the first node. Additionally, we have managed to keep the computational complexity of these strategies at a reasonable level. This evaluation is supported by practical tests conducted using the NYM network[1] and the RIPE atlas dataset[2], as well as an analysis of a uniform distribution model for link delay across the network.

In addition to latency reduction, our analysis extends to the security implications, focusing on potential threats posed by mixnode adversaries together with global passive adversaries. A mixnode adversary's objective is to compromise mixnodes, including the entry hop of the mixnet, to correlate incoming messages with their outgoing counterparts. Our analytical findings indicate that CLAM's methods of selecting initial mixnodes does not significantly enhance the ability of such adversaries to breach anonymity, showing only a marginal increase (2.5%) in their received traffic rate when compromising 20% of the mixnet, compared to the uniform routing. This highlights CLAM's resilience against adversarial attempts to undermine the anonymity provided by the mixnet.

Eventually, we extend our analysis to consider an end-to-end network scenario, spanning from the client to the last layer of the mixnet. In this analysis, we examine both the end-to-end latency and the advantage of a global passive adversary as well as mixnode adversaries. The analysis suggests that using the CLAM approach together with LARMix can reduce latency by 51% in comparison with using LARMix alone. Furthermore, it is shown that under CLAM, anonymity is not significantly compromised while still providing minimal advantage to the mixnode adversary.

The remainder of this paper is organized as follows: Section 2 reviews the relevant literature. Section 3 introduces CLAM's innovative routing schemes, designed to reduce latency and achieve load balance. Section 4 presents the evaluation metrics for CLAM and analyzes its performance enhancements. Section 5 delves into the adversarial setting, focusing on the potential compromise of mixnodes. The paper concludes with Section 6, which extends the results to encompass end-to-end communications, and Section 7, which synthesizes our findings and outlines directions for future research.

2 RELATED WORK

A notable approach to reducing latency in mixnets through latency-aware routing mechanisms is presented by LARMix [17], which offers a methodological strategy for node selection and message routing aimed at minimizing link delays within the mixnet. While effective in reducing latency, this method does not tackle the challenge of reducing the link delay between clients and the initial hop of the mixnet, an issue which will be addressed in this paper.

In contrast, a broad range of proposals within the Tor network [1–3, 5, 11, 15, 18, 20, 25] enables clients to strategically select relays. These initiatives aim to enhance Tor's resilience against both passive and active adversaries, focusing on the anonymization of messages. Specifically, location-aware strategies [4, 14, 23, 24] in Tor facilitate relay selection based on the client's geographical location to potentially bypass surveillance efforts by Autonomous Systems (ASes) or Internet Service Providers (ISPs). However, such strategies are specifically tailored for Tor, where the threat model does not

[1]The NYM network (https://nymtech.net) is a deployed mixnet that adopts a Loopix-like architecture [16], featuring mixnets from various jurisdictions. The Verloc mechanism [13], implemented in this network, facilitates latency measurement and mixnode location determination.

[2]The RIPE dataset [22] provides a wide-reaching collection of endpoints, ensuring extensive coverage for our analysis.

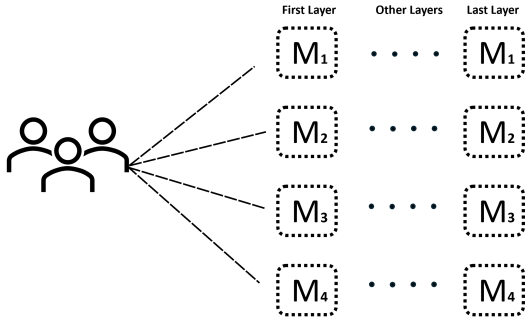


Figure 2: CLAM overview.

assume a global passive adversary. In mixnets, anonymity hinges on the mixing process executed by mixnodes, to avoid being under surveillance of a global passive adversary.

The most pertinent example of Tor's strategic routing to this paper is LASTor [1], which endeavors to decrease network latency by directing traffic through geographically proximate Tor relays. While effective in reducing latency, LASTor does not tackle the challenge of evenly distributing the load among Tor relays. This can result in performance bottlenecks, as certain nodes may become overburdened. Inspired by this observation, our work introduces new strategies that aim to achieve both low latency and equitable load distribution, thus overcoming the limitations identified in LASTor's approach.

Moreover, the study in [18] investigates the potential of clustering clients based on geographical or resilience metrics to further obscure their locations and strategically enhance the first relay selection process in Tor. However, we contend that such methodologies, while innovative, might not be directly transferable to or necessary for mixnet configurations. Mixnets inherently account for the possibility of global passive adversaries monitoring network traffic. As such, the specific challenges and solutions articulated for Tor, including those proposed in [18], may not align with the foundational principles and threat models of mixnets.

3 APPROACH

In this section, we clarify the specific problem our research on CLAM seeks to address, followed by an introduction to our novel methodologies devised to mitigate these issues.

3.1 Problem Statement

CLAM is designed to tackle the dual challenge of reducing latency in mixnet communications while ensuring that network load is distributed relatively evenly across the network.

Fig. 2 illustrates a scenario in which several clients aim to connect to a mixnet to anonymize their messages before sending them to specific destinations. The current standard method for routing clients' messages through the mixnet involves randomly selecting the first hop among the available mixnodes as the initial hop of the mixnet, a strategy that aids in equitable traffic distribution and network load balancing. However, this conventional approach does not account for the potential increase in latency that arises from selecting mixnodes geographically distant from the clients, thereby negatively affecting network performance and user experience.

To overcome this limitation, CLAM introduces an innovative strategy for the initial mixnode selection that prioritizes geographical proximity to the client, thus reducing latency by selecting nearer mixnodes for the first hop. Furthermore, CLAM integrates strategies to ensure that this proximity-based selection does not significantly deviate from achieving optimal load balancing across the network. Clients can choose subsequent hops within the mixnet using any preferred strategy, including LARMix [17], highlighting that CLAM's latency-reduction approach can be seamlessly integrated with existing methods to provide a comprehensive solution to latency challenges in mixnet environments.

3.2 Methodology

CLAM proposes three innovative strategies for selecting the initial hops in a mixnet, aiming to balance the trade-off between equitable traffic load distribution among the W mixnodes in the first mixnet hops and minimizing latency for M clients connecting to those mixnodes.

3.2.1 $K\alpha$-Closeness Routing. We define $V_K(C_i, S_{Mix})$ as a function that takes a client C_i and the set of available mixnodes in the initial mixing hops S_{Mix}. This function outputs a vector containing 0s and 1s, reflecting the latency distance of client C_i from the mixnodes. The vector assigns a 1 to entries representing mixnodes that are within the top K in terms of latency closeness to the client, and 0 to all others, assuming mixnodes are ranked from 0 to $W-1$ based on their latency distance from the client[3].

The probability distribution for a client C_i to select any mixnode as the first hops in the mixnet is then calculated using Eq. (1). For a constant K, a setting of $\alpha = 0$ ensures that the K closest mixnodes are chosen with a probability of $\frac{1}{K}$, excluding the rest. This method biases selection towards nearer mixnodes within that subset, maintaining fairness within those choices. As α increases, the selection bias towards the closest mixnodes decreases, and at $\alpha = 1$, all mixnodes are chosen with equal probability $\frac{1}{W}$, mirroring the LARMix approach for uniform selection. Thus, increasing α gradually shifts towards a more equitable distribution across all mixnodes.

Conversely, with a constant α and varying K, the selection dynamics change. For $K = 1$, the nearest mixnode is chosen with a higher probability of $1 - \alpha + \frac{\alpha}{W}$, favoring proximity. However, for $K = W$, the scenario evolves to a fully balanced uniform selection, with all nodes being selected with an equal probability of $\frac{1}{W}$.

$$f(C_i, M_j) = \begin{cases} \frac{\alpha}{W}, & \text{if } V_K[j] = 0, \\ \frac{1-\alpha}{K} + \frac{\alpha}{W}, & \text{otherwise.} \end{cases} \quad (1)$$

3.2.2 Linearly Programmed Routing (LPR). LPR method employs a linear programming model to optimize the routing from clients to the first hop of the mixnet. The formulation, presented in Eq. (2), aims to minimize the average latency experienced by clients when connecting to the initial hop of the mixnet. Let r_{ij} represent the probability that client i selects mixnode j, and L_{ij} denote the latency

[3]It is assumed that clients can measure their latency to all mixnodes, or such information is accessible through decentralized mechanisms like Verloc [13].

between client i and mixnode j. The objective is to minimize the overall average latency, expressed as $\frac{1}{MW}\sum_{i=1}^{M}\sum_{j=1}^{W} r_{ij}L_{ij}$ [4]. This optimization is subject to several constraints to ensure feasibility and fairness in routing decisions:

- The routing probabilities r_{ij} must be within the range $[0, 1]$ for all clients i and mixnodes j.
- Each client i must distribute its routing probabilities across the mixnodes in the first layer such that $\sum_{j=1}^{W} r_{ij} = 1$, ensuring that the routing probabilities for each client form a valid probability distribution.
- The load balance among mixnodes is controlled by the constraint $\frac{\alpha}{W} \leq \frac{\sum_{i=1}^{M} r_{ij}}{M} \leq \frac{W+\alpha(1-W)}{W}$ for each mixnode j, where $0 \leq \alpha \leq 1$. This ensures that traffic is distributed across mixnodes within specified bounds, achieving a balance between fair traffic distribution ($\alpha = 1$) and a low-latency bias ($\alpha \neq 1$).

The LPR method, therefore, provides a structured approach to minimizing latency while adhering to constraints that maintain a balanced load across the mixnet's first hop based on variable α which does not necessarily as same as the one in Eq. 1.

$$\begin{aligned} \text{Minimize} \quad & \frac{1}{MW}\sum_{i=1}^{M}\sum_{j=1}^{W} r_{ij}L_{ij}, \\ \text{subject to} \quad & \forall i, j, \quad 0 \leq r_{ij} \leq 1, \\ & \forall i, \quad \sum_{j=1}^{W} r_{ij} = 1, \\ & \forall j, \quad \frac{\alpha}{W} \leq \frac{\sum_{i=1}^{M} r_{ij}}{M} \leq \frac{W+\alpha(1-W)}{W}. \end{aligned} \tag{2}$$

3.2.3 Exponentially Prioritized Routing (EPR). EPR utilizes an exponential distribution to prioritize routing from clients to closer mixnodes. Let us consider a random variable $X \sim \exp(\lambda)$, characterized by the probability distribution $\mathbb{P}(X \leq x) = 1 - e^{-\lambda x}$. In addition, we define function $R(.)$ that ranks mixnodes for a client C_i by closeness, with $R(C_i, M_j)$ indicating the rank of mixnode M_j from 0 (closest) to $W-1$ (furthest).

In this routing approach, the probability that client C_i selects mixnode M_j as the first hop is determined by Eq. (3), tying the selection probability to the rank of mixnode closeness under the exponential distribution:

$$\begin{aligned} f(C_i, M_j) &= \frac{\mathbb{P}\left(X \leq R(C_i, M_j)\right) - \mathbb{P}\left(X \leq R(C_i, M_j) - 1\right)}{f_N}, \\ &= \frac{e^{-\lambda(R(C_i,M_j)-1)} - e^{-\lambda R(C_i,M_j)}}{f_N}. \end{aligned} \tag{3}$$

To normalize the probabilities, ensuring they sum to one, we compute the normalization factor f_N as:

$$\begin{aligned} f_N &= \sum_{j=1}^{W}\left[e^{-\lambda(R(C_i,M_j)-1)} - e^{-\lambda R(C_i,M_j)}\right], \\ &= \left(e^{\lambda}-1\right)\sum_{j=1}^{W} e^{-\lambda R(C_i,M_j)}, \\ &= \left(e^{\lambda}-1\right)\left[1 + e^{-\lambda} + e^{-2\lambda} + e^{-3\lambda} + \cdots\right], \\ &= \left(e^{\lambda}-1\right)\frac{1-e^{-\lambda W}}{1-e^{-\lambda}}. \end{aligned} \tag{4}$$

This yields the selection probability for M_j from C_i as Eq.(5).

$$f(C_i, M_j) = \frac{e^{-\lambda R(C_i,M_j)}\left(1-e^{-\lambda}\right)}{1-e^{-\lambda W}}. \tag{5}$$

The parameter λ adjusts the routing bias towards closer mixnodes: a small λ results in a uniform distribution, while a large λ biases the selection towards the closest mixnode. The limits of $f(C_i, M_j)$ as λ approaches 0 and ∞ are given by Eq. (6) and Eq. (7), respectively, illustrating the transition from uniform to fully low-latency selection:

$$\begin{aligned} \lim_{\lambda\to 0} f(C_i, M_j) &= \lim_{\lambda\to 0} \frac{e^{-\lambda R(C_i,M_j)}\left(1-e^{-\lambda}\right)}{1-e^{-\lambda W}}, \\ &= \lim_{\lambda\to 0} \frac{e^{-\lambda R(C_i,M_j)}\left(1-\left[1-\lambda+O\left(\lambda^2\right)\right]\right)}{1-\left[1-\lambda W+O\left(\lambda^2\right)\right]}, \\ &= \lim_{\lambda\to 0} \frac{e^{-\lambda R(C_i,M_j)}\left(1-[1-\lambda]\right)}{1-[1-\lambda W]}, \\ &= \frac{1}{W}. \end{aligned} \tag{6}$$

$$\begin{aligned} \lim_{\lambda\to\infty} f(C_i, M_j) &= \lim_{\lambda\to\infty} e^{-\lambda R(C_i,M_j)}, \\ &= \begin{cases} 1, & \text{if } R(C_i, M_j) = 0, \\ 0, & \text{otherwise.} \end{cases} \end{aligned} \tag{7}$$

4 EVALUATION

In this section, we outline the metrics and datasets employed to evaluate the CLAM framework, followed by an analysis of the latency reduction and load balancing status achieved through CLAM's routing strategies.

4.1 Datasets and Metrics

The evaluation of CLAM necessitates simulating the latency between network nodes to quantify the latency reduction. To this end, we employ three datasets [5]:

(1) The RIPE Atlas dataset, a global initiative for measuring internet latency, featuring over 10,000 nodes worldwide.
(2) NYM dataset derived using the Verloc mechanism [13] within the NYM network, a Loopix-based mixnet, to estimate node latencies through mixnodes collaboration [8].

[4] Note that our linear programming method is inspired by CLAPS [18], where a general linear programming optimization is provided to minimize an objective function within the Tor network.

[5] Note that the RIPE and NYM datasets used in this work were initially derived from LARMix: https://github.com/larmix/larmix

(3) Synthetic latencies assigned based on a uniform distribution, with an average of 60 ms, as derived based on average latencies in RIPE and NYM datasets.

These datasets collectively facilitate a comprehensive latency analysis for CLAM.

Latency measurement is subsequently performed by calculating the weighted average latency, expressed as $\frac{1}{MW}\sum_{i=1}^{M}\sum_{j=1}^{W} r_{ij}L_{ij}$, where L_{ij} represents the latency between nodes i and j, and r_{ij} denotes the routing probability.

For load balancing assessment, we examine the average traffic directed to mixnodes $\frac{1}{MW}\sum_{i=1}^{M} r_{ij}$, with the balancing factor for a mixnode j defined as $b_j = \frac{1}{MW}\sum_{i=1}^{M} r_{ij} - \frac{1}{W}$. This metric indicates the deviation from ideal load distribution, where a perfectly balanced mixnode receives an equal fraction of the total incoming traffic, ideally $\frac{1}{W}$. For instance when $W = 50$, a mixnode handling double the expected traffic would have $b_j = 0.02$, signifying an overload, while $b_j = -0.01$ indicates half the expected load, suggesting under loaded scenario. The balancing factors are illustrated using box plots to depict the distribution of load imbalances across mixnodes, highlighting the efficacy of our routing strategies in mitigating over or underloading scenarios.

4.2 Latency Reduction

To assess the latency reduction efficacy of CLAM, we considered an environment with $W = 50$ potential mixnodes serving as the first hop in the mixnet connected to $M = 200$ clients, each using CLAM to route their messages through the network. To ensure the robustness of our findings, the process of mixnodes slections was repeated 500 times. The resultant average latency between clients and the initial mixnet hop, under various routing strategies, is illustrated in Fig. 3. These strategies include $K\alpha$-Closeness, LPR and EPR.

Fig. 3a plots the relationship between the average latency and the randomness parameter α in the αK-Closeness approach, with K set to a constant value of $\frac{W}{10} = 5$. It is observed that an increase in α correlates with an increase in average latency. This phenomenon indicates that a higher α value reduces the priority of selecting the nearest mixnodes to the clients, thereby increasing the likelihood of choosing more distant mixnodes and, as a result, elevating the average latency.

In this experiment, results of different datasets (RIPE, NYM, and a uniform latency distribution), consistently demonstrate a similar trend. Notably, the NYM dataset typically results in lower latencies. A particularly interesting observation across all three datasets is their nearly linear response to changes in α. This linearity arises from the selection probability's direct correlation with α in Eq. (1). With a fixed K, the formula for calculating average latency exhibits a linear dependence on α. Through this analytical approach, we have determined that latency reductions of up to 74% are achievable for all the datasets when α is set to 0.

Fig. 3b presents how average latency changes with different values of K in the $K\alpha$-Closeness approach, with α set to 0.2. We find that as K increases, so does the average latency. This happens because a larger K includes more mixnodes as potential choices for clients, spreading out message delivery across more nodes. This wider spread means less focus on the very closest mixnodes, leading to higher average latency. When K is at its smallest, latency drops by as much as 58% in the NYM and RIPE dataset. This smaller decrease, compared to situations with a lower α, highlights α's role in adding randomness to mixnode selection. With α at 0.2, there's still a chance to pick mixnodes that have higher latency, even if K is small.

Fig. 3c illustrates the significant latency reduction achieved through the LPR approach, which optimizes routing by considering all potential paths from clients to the first-hop mixnodes in the mixnet. This comprehensive analysis leads to substantial improvements in latency. Interestingly, we observe that increasing the parameter α within this approach results in a minor increase in latency. This increment is attributed to the introduction of randomness by increasing α, which is intended to ensure a more balanced distribution among mixnodes.

It is important to note that in the context of linear programming, setting α to 1 does not equate to a completely random selection of mixnodes. Instead, it ensures that all first-hop mixnodes in the mixnet are chosen with equal probability, $\frac{1}{W}$. This approach can result in up to an 80% latency reduction compared to a uniform random routing [6] even if $\alpha = 1$. Moreover, latency can be further reduced by up to 90% when $\alpha = 0$. However, achieving optimal latency reduction through linear programming requires detailed information about client connections to the mixnet, a process that may pose implementation challenges.

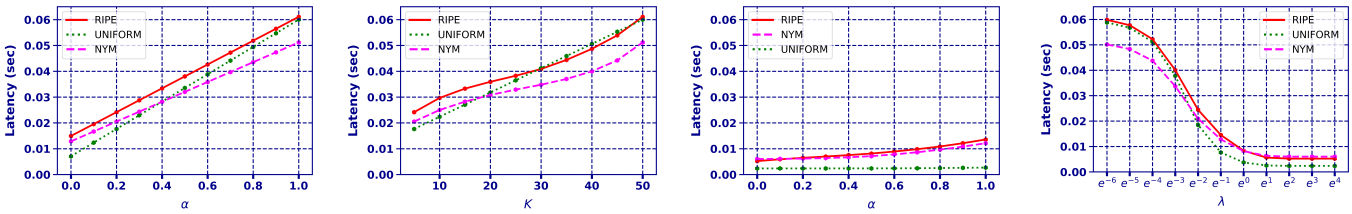
Fig. 3d presents the impact of varying the routing parameter λ on average latency in the EPR framework. As λ increases, latency decreases, aligning with expectations. This trend occurs because a larger λ heavily favors the nearest mixnodes to the clients, optimizing latency. Specifically, when λ is set to infinity, the latency reduction mirrors that achieved with $\alpha = 0$ in the LPR approach, showcasing significant efficiency. Conversely, reducing λ diversifies mixnode selection, increasing randomness. At $\lambda = 0$, selection among mixnodes becomes uniformly random. The experiments demonstrate that EPR achieves latency reductions comparable to LPR, with up to a 90% decrease across all datasets. In addition, EPR unlike LPR, is dependent from prior client information for routing decisions, enhancing its practicality for real-world application.

4.3 Imbalance Loads

Fig. 4 depicts the balancing factor b across different routing strategies, using the RIPE, NYM, and uniform datasets to represent latency distributions between clients and mixnodes. Fig. 4a focuses on the $K\alpha$-Closeness approach, setting $k = \frac{W}{10} = 5$, and employs box plots to illustrate the range of balancing factors among mixnodes. This visualization helps identify the extent of imbalance that may arise under different configurations.

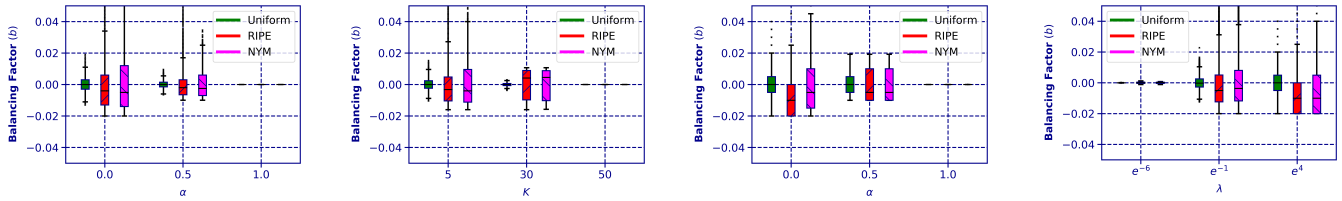
A key finding is that increasing α leads to a decrease in the variance of the balancing factor. Specifically, when $\alpha = 1$, the variance drops to zero, indicating perfect balance among mixnodes. Conversely, for $\alpha \neq 1$, mixnodes exhibit imbalance. Notably, the uniform dataset shows the least variance in imbalance, even when $\alpha = 0$. The maximum variance observed in the uniform dataset is under two percent, suggesting that a well-distributed network of mixnodes around the globe can prevent significant overloading,

[6] when $\alpha = 1$ or $K = W$ in the $K\alpha$-Closeness approach.



(a) $K\alpha$-Closeness, with $k = 5$ **(b) $K\alpha$-Closeness, with $\alpha = 0.2$** **(c) LPR** **(d) EPR**

Figure 3: Latency reduction provided by different routing approaches.



(a) $K\alpha$-Closeness, with $k = 5$ **(b) $K\alpha$-Closeness, with $\alpha = 0.2$** **(c) LPR** **(d) EPR**

Figure 4: Imbalance loads caused by strategic routing approaches.

even with biased routing choices. Essentially, if each country has mixnodes proportional to its number of clients aiming to choose the nearest mixnodes, ensures a balanced load despite inherent biases in selection.

Fig. 4a also reveals that with a uniform latency distribution, the median balancing factor is zero. This equilibrium indicates an equal number of overloaded and underloaded mixnodes. However, in the RIPE and NYM datasets scenarios, the maximum imbalance can reach up to 5%, equating to some mixnodes handling up to $\frac{0.05}{\frac{1}{50}} = 2.5$ times more traffic than in a balanced case. The variability in balancing is particularly pronounced in the NYM dataset, likely due to its smaller node count compared to the RIPE dataset. Moreover, the median balancing factors for RIPE and NYM are negative for $\alpha \neq 1$, signifying a predominance of underloaded nodes, with a few mixnodes experiencing significantly higher than average traffic loads.

Fig. 4b examines the balancing factor b in the $K\alpha$-Closeness approach with α set to 0.2, exploring the effects of different K values. As K increases, the amount of randomness for mixnode selection increases, promoting a more balanced distribution of loads across the nodes. Unlike the impact of increasing α, a higher K value tends to distribute loads across a broader set of mixnodes, mitigating the risk of any single node becoming fully overloaded. This approach results in a more even load distribution. The median value of balancing factor in this case is positive. This ensures that while some mixnodes may experience slightly higher loads, the overall network balance is maintained, avoiding the extreme overloads that could occur with a fixed K and varying α. The observed improvement in balance highlights the effectiveness of setting $\alpha = 0.2$, which achieves a judicious selection of mixnodes, thereby enhancing network performance and stability.

Fig. 4c showcases how the balancing factor b varies with adjustments to the α parameter in LPR strategies. It is evident that an increase in α leads to a decrease in load imbalances among the mixnodes. This effect is due to the inherent design of the linear programming approach, which progressively enhances the probability of achieving a balanced network as α increases. The trend in load distribution across different datasets mirrors that of the $K\alpha$-Closeness approach when K is fixed but with a slight difference: the median balancing factor is negative, indicating a propensity for mixnodes to be underloaded rather than overloaded. Notably, at $\alpha = 1$, the variance in balancing factors drops to zero, signifying a state of perfect balance across the network's mixnodes.

Fig. 4d explores the balancing factor b in EPR as the λ parameter is adjusted. It reveals that higher λ values lead to increased load imbalances among mixnodes. The most balanced scenario occurs at $\lambda = 0$, where mixnodes are selected with equal probability, resulting in a fully random and thus balanced distribution of loads.

A noteworthy observation is the similarity between the performance of EPR and the LPR approach, particularly in terms of load balancing and latency reduction. Both strategies demonstrate a comparable ability to significantly reduce latency, with EPR capable of achieving up to a 90% reduction. However, this comes at the cost of a 4% increase in network imbalance in the worst-case scenario, where some mixnodes may receive two times more traffic compared to a balanced case. This similarity and the effectiveness in latency reduction suggest that EPR could be a viable alternative to LPR, offering a promising balance between efficiency and network load distribution.

5 MIXNODE ADVERSARY

This section delves into the threat posed by adversaries in control of a subset of mixnodes within the mixnet. The primary aim of such adversaries is to maximize the fraction of message routes that pass through entirely compromised paths, enhancing their ability to de-anonymize messages in collaboration with global adversaries.

A particular focus is placed on the strategic selection of mixnodes in the first layer of the mixnet within the CLAM framework and its potential to augment the adversary's effectiveness in corrupting initial hops of the mixnet. This, in turn, could lead to an increased number of end-to-end paths being compromised, facilitating the full de-anonymization of messages.

We further assume that mixnodes in the first hop of the mixnet are assigned to this position uniformly at random, in a decentralized manner[7]. This assumption implies that the adversary lacks the ability to influence this initial distribution and can only corrupt up to β percent of the mixnodes within the mixnet.

5.1 Adversary Strategies

As previously outlined, adversaries aim to exploit the CLAM routing strategies to strategically compromise mixnodes, thereby enhancing their capability to de-anonymize messages. To achieve this, they might adopt various tactics:

(1) **Random Corruption:** The most straightforward strategy involves arbitrarily corrupting mixnodes within the initial hops of the mixnet. This method does not account for the routing strategy employed, potentially leading to a corruption of β percent of received traffic.
(2) **Cluster Corruption:** Leveraging the clustering algorithm detailed in Appendix B, adversaries can categorize available mixnodes based on the their latency table and geographical locations. By targeting the largest cluster for corruption, the adversary banks on the premise that mixnodes within this cluster, being closer and presumably more attractive to clients seeking lower latency paths, will intercept a higher volume of traffic. This tactic aims to corrupt traffic flow more effectively by focusing on strategically positioned mixnodes.
(3) **Greedy Corruption:** Another approach involves utilizing a greedy algorithm, as described in Appendix B, to corrupt mixnodes that would maximize the adversary's control over the network's traffic. While this method could potentially divert a significant portion of the traffic through compromised nodes, it faces practical challenges. Without precise knowledge of mixnode assignments within the mixnet, this strategy serves more as a theoretical upper bound on the adversary's potential impact when clients use CLAM routing strategies. This concept mirrors the scenario presented in LARMix [17], which explores the extreme case of maximizing corrupted paths.

5.2 CLAM Evaluation with Mixnode Adversary

In this section, we begin by examining the effects of various adversarial tactics on the proportion of traffic that corrupted mixnodes receive within the CLAM methodologies. Here, we solely assess the outcomes utilizing the NYM dataset, with an expansion to other datasets detailed in Appendix C.

Fig. 5 showcases the Fraction of Corruptions (FC), representing the portion of traffic intercepted by compromised mixnodes, under various adversarial tactics within the CLAM framework. Through an analysis of the NYM dataset, the effectiveness of these strategies across different CLAM routing protocols is evaluated.

[7]As suggested in [8].

Fig. 5a focuses on the performance of the $K\alpha$-Closeness strategy with k set to 5. The findings indicate that both Random and Cluster corruption strategies result in a corruption rate of approximately 20%, irrespective of the α value. This suggests that in the absence of detailed knowledge regarding mixnode assignment to the mixnet, adversaries are unable to significantly augment the fraction of traffic routed through their controlled nodes beyond the share of mixnodes they possess.

Furthermore, the efficacy of employing a greedy strategy for corruption, which represents the theoretical upper limit of adversarial advantage, decreases as α increases. Notably, at $\alpha = 1$, the FC for the greedy strategy aligns with those observed for both Random and Cluster approaches. This convergence is attributed to the introduction of increased randomness within the network, which effectively mitigates the adversary's ability to preferentially target mixnodes handling a larger share of the traffic.

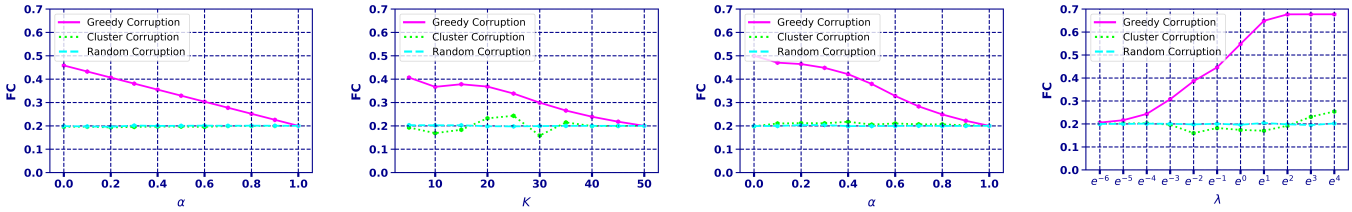
Fig. 5b elucidates the impact of varying K values on the Fraction of Corruptions (FC) within the $K\alpha$-Closeness strategy, with α fixed at 0.2. The analysis reveals that Random and Cluster strategies for compromising mixnodes in the initial hop of the mixnet yield comparable results across most K values. Notably, the Cluster strategy exhibits a marginal advantage in the range of $15 < K < 25$. This phenomenon suggests that increasing K leads to a larger subset of mixnodes within the corrupted cluster being classified as closer to clients, thereby attracting a greater share of traffic to these compromised mixnodes. However, this advantage wanes as K increases beyond this range, with the consideration of additional clusters' mixnodes as close, dispersing the targeted traffic more broadly.

The Greedy strategy, which aims for the most corruption, works best when K is small. But as K grows, its results look more like the Random and Cluster strategies. Although the Greedy strategy could lead to more corruption at smaller K values, its edge lessens because setting α to 0.2 adds some randomness to which mixnodes are chosen. This randomness reduces how much an adversary can benefit from carefully picking mixnodes to corrupt.

Figures 5c and 5d present the outcomes of routing strategies derived from LPR and EPR, respectively. In both cases, employing Random and Cluster strategies for corrupting mixnodes results in a similar fraction of traffic corruption, approximately around 20 percent. This indicates that leveraging the Cluster strategy does not significantly enhance the adversary's ability to corrupt more paths, except in the exponential distribution scenario where a high λ value slightly increases the fraction of corrupted paths.

Furthermore, the upper bound derived from the Greedy approach demonstrates that as α increases or λ decreases, its superiority diminishes, aligning the results with those obtained through Random or Cluster strategies. Notably, in the exponential prioritized routing, when λ approaches infinity, the upper bound of corruption can surge to 0.7, as opposed to 0.5 in linear programming with $\alpha = 0$. This suggests that, although the evaluation section proves that both LPR and EPR approaches yield comparable results in terms of latency reduction and load balancing variance, the EPR approach potentially offers a greater opportunity for adversaries to intercept or corrupt more traffic.

While both the LPR and EPR approaches achieve comparable efficiency in reducing latency and offer similar advantages in adversarial settings, it is crucial to highlight the distinctions between



(a) $K\alpha$**-Closeness, with** $k = 5$ **(b)** $K\alpha$**-Closeness, with** $\alpha = 0.2$ **(c) LPR** **(d) EPR**

Figure 5: Fraction of received traffic by corrupted mixnodes (Fraction of Corruption) when $\beta = 0.2$**.**

them. Specifically, LPR necessitates the utilization of a linear programming methodology, which is known for its substantial computational demands. This demand stems from the number of variables and constraints that must be managed to solve the linear programming, potentially leading to exponential complexity. Furthermore, LPR requires prior information about recently joined clients in the network, introducing practical challenges. In contrast, the EPR approach does not necessitate such assumptions, rendering it more adaptable to practical scenarios.

6 END-TO-END ANALYSIS

In this section, we broaden our analysis of latency and adversarial threats to encompass end-to-end communications extending from the client to the final layer of the mixnet, as illustrated in Fig.2. To minimize end-to-end latency, clients employ CLAM routing strategies, efficiently forwarding packets to the mixnet. Subsequently, LARMix routing can be utilized to further reduce latency within the mixnet, from the initial to the final layer. Additionally, we examine the potential advantages for a global passive adversary using an analytical method similar to that proposed by LARMix [17], while also considering the threats posed by mixnode adversaries in these scenarios.

Throughout the experiments detailed in this section, we exclusively utilized the NYM dataset to simulate link delay latency within a mixnet configured with $L = 3$ layers and $W = 50$ nodes per layer. The initial routing from clients to the first layer was evaluated using either CLAM routing strategies or a uniform routing approach. In the implementation of CLAM, routing strategies incorporated $K\alpha$-Closeness with parameter settings of $\alpha = 0.2$ and $K = 5$, along with LPR and EPR, set at $\alpha = 0.2$ and $\lambda = e^2$ respectively. Furthermore, within the mixnet—from the first layer onward—routing was executed using either uniform methods or through LARMix, which was configured with a tuning parameter $\tau = 0.6$. This parameter setting is recommended to achieve an optimal balance between latency and anonymity, as outlined in Eq. (1) on page 5 of the LARMix [17].

6.1 End-to-End Latency

Tab. 1 presents the results of the end-to-end latency analysis, where the first column lists the client-to-mix routing approaches, and the first row indicates mix-to-mix routing strategies. When both message forwarding to the mixnet and within mixnet routing are uniform, the end-to-end latency averages 153.4 ms. Using LARMix for within mixnet routing can reduce this latency to 97.7 ms, a 36% reduction in end-to-end latency. However, combining CLAM (EPR) with LARMix routing achieves a latency of 47.3 ms, demonstrating that considering CLAM routing strategies combined with LARMix settings can significantly reduce the end-to-end latency by up to 51% compared to LARMix solely deployment. Moreover, the combination of CLAM with uniform routing results in a latency of 105.5 ms compared to 153.4 ms for uniform routing across all connections, representing a 31% reduction in latency, thus showing the high efficacy of CLAM routing strategies.

Table 1: Comparative analysis of end-to-end latency.

Routing Approaches	LARMix	Uniform
Kα-Closeness	62.0 ms	117.2 ms
LPR	47.8 ms	105.9 ms
EPR	47.3 ms	105.5 ms
Uniform	97.7 ms	153.4 ms

Table 2: Comparative analysis of end-to-end anonymity.

Routing Approaches	LARMix	Uniform
Kα-Closeness	5.32 bits	5.64 bits
LPR	4.94 bits	5.64 bits
EPR	4.94 bits	5.64 bits
Uniform	5.64 bits	5.64 bits

6.2 End-to-End Anonymity

LARMix [17] developed a method to assess the advantage of a global passive adversary when deploying low-latency routing within mix networks. They posit that under uniform routing, upon any message being received by a specific mixnode in the first mixing layer, the probability of its exit from any mixnode in the last layer adheres to a uniform distribution. This distribution alters when routing favors low-latency paths, consequently enhancing a global passive adversary's ability to infer message routes by observing network links. LARMix describes this through a transformation matrix (T), where each row i corresponds to the probability distribution of messages entering mixnode i in the first layer across all mixnodes in the last layer. They utilize the Shannon entropy [19] of each row in this matrix to report their average value as a metric of analytical anonymity [8].

[8]To derive the transformation matrix, one first calculates the routing matrix between layers and then performs matrix multiplication.

In contrast, CLAM implements biased routing from clients to the mixnet, which necessitates adjusting our analysis to assess the probability distribution of messages from a specific client to all mixnodes in the final layer of the mixnet. This analysis involves deriving a transformation matrix from clients to the last mixing layer, where we compute the average entropy of the rows to measure the anonymity provided to clients.

To evaluate the entropy of the transformation matrix for CLAM in combination with uniform and LARMix routing, we conducted experiments, the results of which are presented in Tab. 2. When uniform routing is applied for both client-to-mix and mix-to-mix links, the entropy of the transformation matrix reaches 5.64 bits, equivalent to log(50). This indicates a uniform distribution of each client's messages to the last layer of the mixnet, thus maximizing anonymity. Interestingly, using uniform routing for the client-to-mix link combined with mix-to-mix routing via LARMix, or CLAM routing for client-to-mix and uniform routing for mix-to-mix, also achieves an entropy level of 5.64 bits. These outcomes may initially seem unexpected but can be rationalized as follows.

Consider the scenario where CLAM routing is applied to forward messages to the first layer of the mixnet, while the rest of the routing is kept uniform. Here, we focus on calculating the probability $\mathbb{P}(M_L|C_i)$ (probability of a message exiting the mixnode M_L in the last layer when initiated by client i), where C_i represents the ith client and M_L denotes the random variable indicating the mixnode in the last layer L, which varies from 1 to W. Given that client-to-mix routing is independent of mix-to-mix routing, this probability can be expressed as:

$$\mathbb{P}(M_L|C_i) = \sum_{j=1}^{W} \mathbb{P}(M_1 = m_j|C_i) \times \mathbb{P}(M_L|M_1 = m_j),$$

where M_1 represents the random variable indicating the chosen mixnode in the first layer. Because the routing within the mixnet ensures a uniform probability of exiting from any mixnode in layer L upon entering any mixnode in the first layer, $\mathbb{P}(M_L|M_1 = m_j) = \frac{1}{W}$. Thus, we have:

$$\mathbb{P}(M_L|C_i) = \sum_{j=1}^{W} \mathbb{P}(M_1 = m_j|C_i) \times \frac{1}{W} = \frac{1}{W} \sum_{j=1}^{W} \mathbb{P}(M_1 = m_j|C_i) = \frac{1}{W}.$$

This demonstrates that when CLAM is used with uniform routing within the mixnet, the probability of a client's message exiting from any mixnode in the last layer remains uniform, thereby validating the results shown in Tab. 2. This reasoning also applies when client-to-mix routing is uniform and mix-to-mix routing employs LARMix.

We can conclude that as long as the mix-to-mix routing remains uniform, CLAM does not reduce the entropy of the transformation matrix. However, combining CLAM with LARMix, we observe that the entropy of the transformation matrix, in the cases of EPR or LPR, is reduced to 4.94 bits, compared to 5.64 bits in scenarios where only LARMix is deployed. Considering the associated reduction in end-to-end latency, this decrease in entropy can be considered insignificant.

6.3 Fraction of Fully Corrupted Paths

In Section 5, we examined strategies to measure the advantage of an adversary compromising mixnodes in the first layer using CLAM routing. To address broader security concerns, we expand this analysis to encompass corruption across all mixnet layers. We model a scenario in which an adversary corrupts $\beta = 20\%$ of the mixnodes within the mixnet, allowing us to measure the fraction of fully corrupted paths (FCP). This metric quantifies the percentage of communication paths completely compromised by the adversary and highlights the real threat posed by adversaries capable of corrupting mixnodes.

Table 3: Comparative analysis of FCP.

Routing Approaches	LARMix	Uniform
Kα-Closeness	0.041	0.016
LPR	0.072	0.018
EPR	0.088	0.027
Uniform	0.062	0.008

The results, detailed in Tab. 3, demonstrate that employing CLAM (EPR) in conjunction with LARMix routing increases the FCP from 0.062 to 0.088 in the worst-case scenario. In contrast, combining CLAM with uniform routing elevates the FCP from 0.008 to 0.027 when compared to entirely uniform routing for both client-to-mix and mix-to-mix links. While the increase in FCP indicates a heightened risk of fully compromised paths when CLAM is used in conjunction with either LARMix or uniform routing, this increment does not significantly enhance the overall adversarial advantage.

7 CONCLUSION

In this work, we introduced three heuristic approaches to direct traffic efficiently to the initial hop of the mixnet, ensuring low-latency routing for clients. Our methods have demonstrated the potential to reduce latency by up to 90%, while increasing the traffic load handled by nodes by up to 4% compared to scenarios with optimal load balancing. Furthermore, we explored the impact of an adversary's attempt to corrupt mixnodes in the mixnet's first layer. Our findings suggest that, despite various strategies and prior knowledge of the CLAM framework, an adversary's ability to significantly enhance its advantage is limited. We believe this paper bridges the gap in providing lower latency connections for clients accessing mixnets, which can be further optimized in combination with other low-latency strategies such as LARMix, aimed at reducing latency across mixnodes within the mixnet, thereby minimizing end-to-end latency.

ACKNOWLEDGMENTS

We would like to thank the anonymous reviewers for their valuable feedback. This research is partially supported by CyberSecurity Research Flanders with reference number VR20192203.

REFERENCES

[1] Masoud Akhoondi, Curtis Yu, and Harsha V Madhyastha. 2012. LASTor: A low-latency AS-aware Tor client. In *2012 IEEE Symposium on Security and Privacy*. IEEE, 476–490.

[2] Mashael AlSabah, Kevin Bauer, Tariq Elahi, and Ian Goldberg. 2013. The path less travelled: Overcoming Tor's bottlenecks with traffic splitting. In *Privacy Enhancing Technologies: 13th International Symposium, PETS 2013, Bloomington, IN, USA, July 10-12, 2013. Proceedings 13*. Springer, 143–163.

[3] Robert Annessi and Martin Schmiedecker. 2016. Navigator: Finding faster paths to anonymity. In *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE, 214–226.

[4] Armon Barton and Matthew Wright. 2016. Denasa: Destination-naive as-awareness in anonymous communications. *Proceedings on Privacy Enhancing Technologies* 2016, 4 (2016).

[5] Armon Barton, Matthew Wright, Jiang Ming, and Mohsen Imani. 2018. Towards predicting efficient and anonymous Tor circuits. In *27th USENIX Security Symposium (USENIX Security 18)*. 429–444.

[6] David L Chaum. 1981. Untraceable electronic mail, return addresses, and digital pseudonyms. *Commun. ACM* 24, 2 (1981), 84–90.

[7] Debajyoti Das, Sebastian Meiser, Esfandiar Mohammadi, and Aniket Kate. 2018. Anonymity trilemma: Strong anonymity, low bandwidth overhead, low latency-choose two. In *2018 IEEE Symposium on Security and Privacy (SP)*. IEEE, 108–126.

[8] Claudia Diaz, Harry Halpin, and Aggelos Kiayias. 2021. The Nym Network. (2021).

[9] Claudia Diaz and Bart Preneel. 2004. Taxonomy of mixes and dummy traffic. In *Information Security Management, Education and Privacy: IFIP 18th World Computer Congress TC11 19th International Information Security Workshops 22–27 August 2004 Toulouse, France*. Springer, 217–232.

[10] Roger Dingledine, Nick Mathewson, and Paul Syverson. 2004. *Tor: The second-generation onion router*. Technical Report. Naval Research Lab Washington DC.

[11] John Geddes, Mike Schliep, and Nicholas Hopper. 2016. Abra cadabra: Magically increasing network utilization in tor by avoiding bottlenecks. In *Proceedings of the 2016 ACM on Workshop on Privacy in the Electronic Society*. 165–176.

[12] Dogan Kesdogan, Jan Egner, and Roland Büschkes. 1998. Stop-and-go-mixes providing probabilistic anonymity in an open system. In *International Workshop on Information Hiding*. Springer, 83–98.

[13] Katharina Kohls and Claudia Diaz. 2022. {VerLoc}: Verifiable Localization in Decentralized Systems. In *31st USENIX Security Symposium (USENIX Security 22)*. 2637–2654.

[14] Rishab Nithyanand, Oleksii Starov, Adva Zair, Phillipa Gill, and Michael Schapira. 2015. Measuring and mitigating AS-level adversaries against Tor. *arXiv preprint arXiv:1505.05173* (2015).

[15] Andriy Panchenko, Fabian Lanze, and Thomas Engel. 2012. Improving performance and anonymity in the Tor network. In *2012 IEEE 31st International Performance Computing and Communications Conference (IPCCC)*. IEEE, 1–10.

[16] Ania M Piotrowska, Jamie Hayes, Tariq Elahi, Sebastian Meiser, and George Danezis. 2017. The loopix anonymity system. In *26th USENIX Security Symposium (USENIX Security 17)*. 1199–1216.

[17] Mahdi Rahimi, Piyush Kumar Sharma, and Claudia Diaz. 2024. LARMix: Latency-Aware Routing in Mix Networks. In *The Network and Distributed System Security Symposium*. Internet Society.

[18] Florentin Rochet, Ryan Wails, Aaron Johnson, Prateek Mittal, and Olivier Pereira. 2020. CLAPS: Client-location-aware path selection in Tor. In *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*. 17–34.

[19] Claude E Shannon. 1949. Communication theory of secrecy systems. *The Bell system technical journal* 28, 4 (1949), 656–715.

[20] Micah Sherr, Matt Blaze, and Boon Thau Loo. 2009. Scalable link-based relay selection for anonymous routing. In *Privacy Enhancing Technologies: 9th International Symposium, PETS 2009, Seattle, WA, USA, August 5-7, 2009. Proceedings 9*. Springer, 73–93.

[21] Fatemeh Shirazi, Milivoj Simeonovski, Muhammad Rizwan Asghar, Michael Backes, and Claudia Diaz. 2018. A survey on routing in anonymous communication protocols. *ACM Computing Surveys (CSUR)* 51, 3 (2018), 1–39.

[22] RIPE Ncc Staff. 2015. Ripe atlas: A global internet measurement network. *Internet Protocol Journal* 18, 3 (2015), 2–26.

[23] Yixin Sun, Anne Edmundson, Nick Feamster, Mung Chiang, and Prateek Mittal. 2017. Counter-RAPTOR: Safeguarding Tor against active routing attacks. In *2017 IEEE Symposium on Security and Privacy (SP)*. IEEE, 977–992.

[24] Gerry Wan, Aaron Johnson, Ryan Wails, Sameer Wagh, and Prateek Mittal. 2019. Guard placement attacks on path selection algorithms for Tor. *Proceedings on Privacy Enhancing Technologies* 2019, 4 (2019).

[25] Tao Wang, Kevin Bauer, Clara Forero, and Ian Goldberg. 2012. Congestion-aware path selection for Tor. In *Financial Cryptography and Data Security: 16th International Conference, FC 2012, Kralendijk, Bonaire, Februray 27-March 2, 2012, Revised Selected Papers 16*. Springer, 98–113.

APPENDIX

A NOTATIONS

The comprehensive list of key notations and variables utilized in this paper is delineated in Tab. 4.

Table 4: List of Key Notations and Variables

Symbol	Meaning
W	Number of mixnodes
M	Number of clients
S_{Mix}	Set of available mixnodes
α	Randomness parameter in $K\alpha$-Closeness approach
V_K	Identity function for determining K closest mixnodes
α	Balance parameter in LPR approach
K	Number of closest mixnodes in $K\alpha$-Closeness approach
R	Rank function
λ	Parameter of an exponential distribution
X	A random variable based on the exponential distribution
β	Fraction of corrupted mixnodes
b_j	Balancing factor for mixnode j
r_{ij}	Probability of routing traffics from client i to mixnode j
Δ_i	Mixing delay introduced at ith hop
l_{ij}	Link delay between nodes i and j
l_{d_i}	Link delay between ith and $i+1$th mixnodes
l_C	Link delay between client and mixnet
l_D	Link delay between mixnet and recipient D
T	Transformation Matrix

B ALGORITHMS

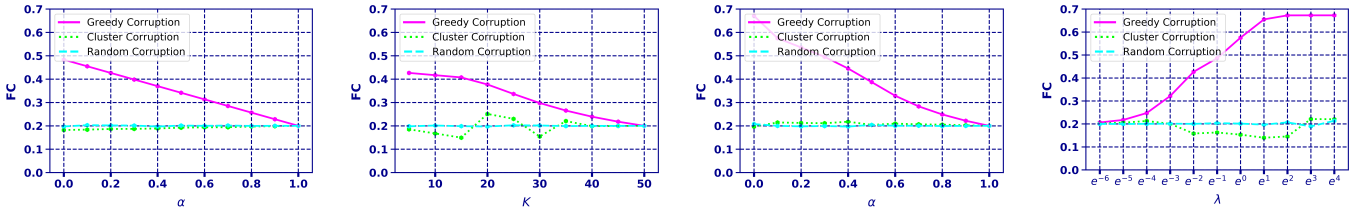
This section outlines the algorithms employed in CLAM to identify compromised mixnodes within the network.

Algorithm 1 Cluster Corruption

Input: Mixnode set S_{mix}, corruption factor β, Latency Table
Output: Set of corrupted mixnodes S_C
Cluster S_{mix} based on proximity in the Latency Table.
Let C be the largest cluster identified.
for each i in the set of $\beta \cdot |S_{mix}|$ **do**
    Select a mixnode at random from C and add it to S_C.
end for
return S_C

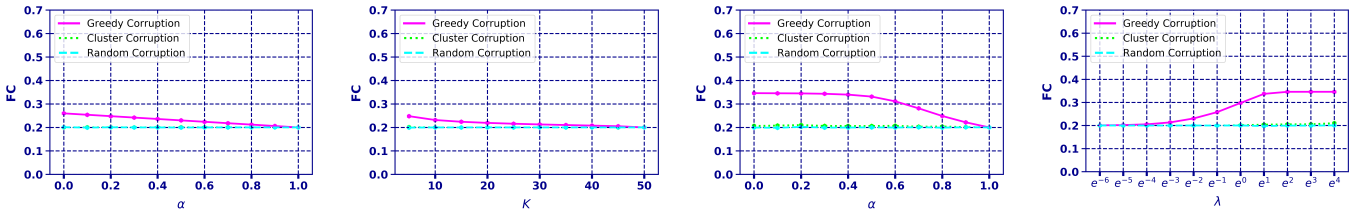
Algorithm 2 Greedy Corruption

Input: Mixnode set S_{mix}, corruption factor β, Routing Table
Output: Set of corrupted mixnodes S_C
for each i in the set of $\beta \cdot |S_{mix}|$ **do**
    $S' \leftarrow$ select a subset from S_{mix} based on β.
    Calculate the fraction of received traffic for S' using the Routing Table.
    Add the mixnode with the highest traffic fraction from S' to S_C.
end for
return S_C



(a) $K\alpha$-Closeness, with $k = 5$ **(b) $K\alpha$-Closeness, with $\alpha = 0.2$** **(c) LPR** **(d) EPR**

Figure 6: Fraction of received traffic by corrupted mixnodes when $\beta = 0.2$, using RIPE dataset.



(a) $K\alpha$-Closeness, with $k = 5$ **(b) $K\alpha$-Closeness, with $\alpha = 0.2$** **(c) LPR** **(d) EPR**

Figure 7: Fraction of received traffic by corrupted mixnodes when $\beta = 0.2$, using Uniform dataset.

C EXPANSION ON ADVERSARIAL ANALYSIS

In this section, we delve deeper into the results previously discussed in Section 5, where we evaluated adversarial settings using three corruption sstrategies: Greedy, Cluster, and Random, utilizing the NYM dataset alongside various routing approaches proposed in CLAM. In this extension of our analysis, we incorporate results from employing both RIPE and Uniform datasets to simulate latency between clients and the initial hop within the mixnet. Figures 6 and 7 depict corruption scenarios observed when employing the RIPE and Uniform datasets, respectively. Consistent with the NYM dataset outcomes, both figures demonstrate that, irrespective of the dataset and routing strategy, Greedy Corruption exhibits a higher volume of traffic received by compromised mixnodes, assuming a 20% mixnode compromise rate. This scenario posits Greedy Corruption as an upper bound, occurring under the premise of pre-corruption access to the routing table, which adversaries typically lack.

In all datasets, Random and Cluster Corruption scenarios yield similar results in terms of the adversary's traffic reception, with exact parity in the Uniform dataset. However, in the RIPE dataset, employing $K\alpha$-Closeness with $\alpha = 0.2$ slightly increases the adversary's traffic reception advantage to 25% upon compromising 20% of the mixnodes. Similarly, the EPR strategy exhibits a minor gain in adversarial traffic reception, whereas for LPR and $K\alpha$-Closeness with $K = 5$, both Random and Cluster Corruption result in identical traffic volumes intercepted by adversaries.

Further analysis reveals that $K\alpha$-Closeness with $\alpha = 0.2$ slightly benefits adversarial traffic reception compared to $K\alpha$-Closeness with $K = 5$. This highlights the significant role of α in introducing randomness into the mixnode selection process, thereby reducing the adversary's chances of targeting mixnodes with higher client traffic volumes.

For both RIPE and Uniform datasets, the LPR and EPR routing strategies exhibit nearly identical behavior as α increases or λ decreases, respectively. A notable exception is a slight increase in adversarial traffic reception observed with EPR when λ is significantly high.

Upon comparing the NYM, RIPE, and Uniform datasets, it becomes evident that the NYM and RIPE datasets exhibit similar patterns in adversarial traffic reception across different corruption strategies and CLAM routing approaches. Conversely, the Uniform dataset significantly reduces the upper bound of corruption effectiveness, demonstrating nearly half the adversarial traffic reception rate observed in the RIPE or NYM datasets. This suggests that a uniform distribution of latency not only enhances client protection against adversarial nodes but also optimizes latency reduction using CLAM. Therefore, we recommend mixnet operators to incentivize participation from diverse global locations to achieve a uniform latency distribution, thereby maximizing performance while minimizing adversarial advantages.