LAMP: Lightweight Approaches for Latency Minimization in Mixnets with Practical Deployment Considerations

Mahdi Rahimi
COSIC, KU Leuven
mrahimi@esat.kuleuven.be

Piyush Kumar Sharma
University of Michigan
piyushks@umich.edu

Claudia Diaz
COSIC, KU Leuven
NYM Technologies SA
cdiaz@esat.kuleuven.be

***Abstract*—Mixnets are a type of anonymous communication system designed to provide network privacy to users. They route client messages through multiple hops, with each hop (mix) perturbing the traffic patterns, thus making message tracing difficult for a network adversary. However, privacy in mixnets comes at the cost of increased latency, limiting the applications that are usable when accessed through a mixnet. In this work we present LAMP, a set of routing approaches tailored for minimizing the propagation latency in mixnets with minimal impact on anonymity. The design of these approaches is grounded in practical deployment considerations making them lightweight, easy to integrate with existing deployed mixnets and computationally realistic. We evaluate the proposed approaches using latency data from the deployed Nym mixnet and demonstrate that LAMP can reduce latency by a factor of 7.5 (from 153.4ms to 20ms) while maintaining high anonymity. LAMP even outperforms the state-of-the-art system LARMix, providing $3\times$ better latency-anonymity tradeoffs and significantly reducing the computational overhead by $\approx 13900\times$ in comparison to LARMix.**

I. Introduction

Mix networks, or *mixnets*, are advanced communication networks that provide network-level anonymity to their users in the face of a global passive adversary that has visibility over all the links in the Internet [6], [17], [23], [8]. Such systems consist of an overlay network that routes packets via multiple intermediary hops and disrupts tracking by *mixing* packets, i.e., transforming and reordering them, at each hop or *mixnode*. While there are various types of mixnets and mixnet architectures [23], we have only recently seen actual deployments [8] for Loopix-based continuous mixnets [17]. Such mixnets have a layered architecture, i.e., mixnodes are grouped in consecutive layers and valid routes traverse exactly one mixnode per layer. Such deployed mixnets can easily support access for delay tolerant applications such as email, messaging, transaction broadcast, *etc.* Applications with real-time latency constraints such as web browsing, however, become less usable when accessed over such mixnets, preventing their wider adoption. Reducing end-to-end latency in mixnets can thus increase the range of Internet activities that can be privately conducted, eventually enhancing end users' privacy.

In terms of end-to-end latency in mixnets, we note that there are two primary sources. First is the *mixing latency*, a delay intentionally introduced at each hop of the mixnet to ensure the mixing (reordering) of packets being routed. While this latency safeguards users from adversaries trying to deanonymize packets, it has a known fundamental tradeoff with anonymity (as described in the anonymity trilemma [7]). Thus, any reduction in this latency would directly and proportionally impact the anonymity guarantees. The second element is the *propagation latency* that is incurred when the packets travel across hops in the mixnet. This latency, in contrast, does not have a direct tradeoff with anonymity guarantees.

A recent solution called LARMix [20] has been proposed to reduce the *propagation latency* in mixnets. LARMix proposes methods that are applied at different steps of a mixnet operation, including (1) clustering mixnodes (based on their location), (2) arranging mixnodes in layers to facilitate availability of diverse routing paths with potential for faster routes, (3) routing policy that biases the path selection towards faster routes, and (4) load balancing approaches to proportionally distribute the load among nodes in the network despite a biased routing policy implementation. We have however identified multiple design choices and underlying assumptions that pose hurdles towards a realistic and practical deployment of LARMix. First, it requires performing computationally intensive operations that exponentially grow with the size of the network (ref. section VI-C for a detailed analysis). For example, calculating the routing policy for 7500 nodes (comparable to the size of most popular anonymous communication system Tor) takes 3 hours using LARMix, which is unusable in a practical deployment.[1] Second, LARMix focuses on minimizing the propagation latency only within the mixnet, missing out on optimization of client-to-mixnet latency, which can be a significant source of overhead if the first hop is located far off from the client. Finally, integrating the LARMix approach with real mixnets such as Nym requires considerable changes to the default design, limiting its chances of practical adoption.

Thus, in this work, we develop LAMP, a set of lightweight, scalable, and easily integratable routing methodologies for the

Network and Distributed System Security (NDSS) Symposium 2025
24-28 February 2025, San Diego, CA, USA
ISBN 979-8-9894372-8-3
https://dx.doi.org/10.14722/ndss.2025.241156
www.ndss-symposium.org

[1] Tor recalculates routing weights every hour and Nym also currently reassigns nodes to layers after one hour.

mixnet. We make two fundamental design choices for the developed approaches. First, the routing approaches assume that the mixnodes have already been arranged in layers, following the vanilla mixnet arrangement (random assignment of mixnodes to layers). This assumption is based on the observation that the techniques for controlling the clustering and arrangement of mixnodes (as developed in LARMix) do not significantly help reduce latency (as reported in LARMix [20]) but pose a high computational burden and require significant changes in the vanilla mixnet for integration. Second, we take two steps to reduce the computational overhead and simplify the design of routing approaches: (1) we do not perform explicit load balancing in the mixnet and demonstrate that doing so does not affect the stability of the network (2) we develop approaches that require only a subset of the global network for routing policy computation. These design decisions collectively contribute towards developing efficient latency minimization schemes. We develop and present three concrete routing schemes based on these principles as part of LAMP, named Single Circle (SC), Multiple Circle (MC), and Regional Mixnet (RM).

The SC approach involves the client considering a circle with itself at the center and a radius defined by a latency bound. Mixnodes are placed in the circle at a distance from the center that represents the propagation latency from the client to that node. The client then selects among the set of mixnodes that lie within the latency-bounded circle to create low-latency routing paths. The circle's radius can be tuned based on the desired latency reduction. We consider multiple methods to select a routing path within the latency circle (selection proportionally to latency, random selection, *etc.*) taking into account additional constraints, such as ensuring the availability of a minimum number of mixnodes for selection in each layer (Sec. III-C1). By design, the SC approach is lightweight and easy to implement, requiring the client to measure its latency to mixnodes and locally calculating the routing policy for a limited set of mixnodes within the circle.

The MC approach is an improvement over SC in covering corner cases where the inter-mixnode latency could still be high despite the low latency between the client and each mixnode in the circle. Similar to SC, MC also creates logical and latency-bounded circles. However, these circles are created at each hop in the end-to-end path to select the appropriate next hops. The client first forms a circle and selects the first hop. The second hop is then selected by creating another circle around the first hop, and so on, till the complete path is obtained. Here, the selection among multiple mixnodes in each layer within the circle is also made based on criteria such as random selection, proportional to latency, *etc.* Note that while MC is an improvement over the SC approach, it slightly increases computation load due to additional processing requirement and reliance on inter-mixnode latency data.

Both the SC and MC approaches require the client to measure and obtain latency information about mixnodes in the network. The RM approach removes that burden from the client by limiting its computation and choices to selecting one from a few available mixnets. The RM approach divides one large global mixnet into multiple smaller regional mixnets. The client routes its packets through a mixnet closer to its location, ensuring reduced latency as the client packets will be routed via mixnodes that are in close geographical proximity.

We perform a thorough evaluation of the three routing schemes using an analytical approach (which allows us to isolate the latency and anonymity impacts due to routing) and a simulation approach (which helps us study the combined effect of routing policy and mixing latency). We use a realistic latency dataset from the deployed Nym mixnet to evaluate the proposed latency optimizations and their anonymity impacts. We measure the latency in seconds and use entropy [9] to quantify anonymity (entropy is measured in bits, and x bits are equivalent to an anonymity set of 2^x equally likely subjects).

Our evaluation shows that the SC approach reduces the propagation latency by $\approx 3\times$ on average compared to a vanilla mixnet, while reducing entropy by ≈ 1.5 bits. The MC approach reduces the latency by about $7.5\times$ with 2 bits loss of entropy. The RM approach with a mixnet consisting of mixnodes in the European Union (EU) reduces latency by $8\times$ with about 2 bit loss of entropy. We also measure the trade-off between latency and anonymity by calculating the E/L (Entropy/Latency) ratio for the three approaches and find that the three LAMP approaches outperform LARMix with the MC and RM providing $\approx 3\times$ better trade-offs than LARMix (see Sec. VI-A for details).

After studying the latency and anonymity trade-offs of the proposed solutions, we analyze the adversarial advantage due to the latency-optimizing routing approaches. We consider a *global passive adversary* that can passively analyze all the network links on the Internet and a *mixnode adversary* that maliciously controls a fraction of the total nodes in the mix network. The adversarial advantage is measured by calculating the fraction of fully corrupted paths (FCP), defined as those paths where all the hops are adversarial, leading to the complete deanonymization of packets traversing them. We calculate the FCP under various adversarial strategies, such as controlling a set of mixnodes in a single location, and find that the developed routing schemes do not significantly increase the adversarial advantage. For a 20% adversarial corruption of mixnodes in the network, the worst case FCP for both SC and LARMix is 0.15.

Lastly, we measure the computational overhead of the proposed schemes by performing a theoretical complexity analysis (see Appendix VI-C for details) and by calculating the normalized computational time required by each approach to calculate the routing policy. For a given network size, we find that SC takes the least time. This is expected due to the simplicity of its design. MC incurs $56\times$ more time than SC, whereas the RM takes $8\times$, showing that these approaches are slightly more computationally intensive. However, all three approaches significantly outperform LARMix as it incurs an overhead of $\approx 13900\times$ compared to SC (see Sec. VI-A), making LAMP approaches good candidates for practical deployment purposes. Overall, the proposed LAMP strategies efficiently minimize latency and are significantly faster than LARMix. LAMP does not confer any undue advantage to adversaries in practical scenarios, offers great tradeoffs, is lightweight and easier to integrate, and remains computationally efficient.

II. Background

A. Mixnets

Mix networks [6], [8], [23], [17], or *mixnets*, are overlay networks that route packets via multiple hops, called *mixnodes*. Mixnets are designed to provide network-level anonymity to their users towards a global passive adversary that observes all communications. To achieve this, each hop in the mixnet performs some form of 'mixing' of incoming packets, by shuffling batches of packets or randomly delaying them, effectively reordering the flow of packets traversing the hop. Together with cryptographic transformations of the packets, this mixing process makes it hard, even for a network adversary observing the inputs and outputs of the mixnode, to map input packets to their corresponding outputs – thereby providing anonymity to the routed packets. Note, however, that anonymity in mixnets comes at the cost of additional latency (necessary for mixing at each hop), limiting the kind of applications that are usable over such networks.

B. State-of-the-Art for Reducing Latency

Reducing latency in anonymous communication systems has majorly been studied for Tor [10], one of the most popular and widely deployed systems. There exists a plethora of strategies [1], [22], [15], [2], [11], [25], [4], [3], [21], [12] that aim at either directly or indirectly improving Tor's performance. The most recent solution and the state-of-the-art for reducing latency in Tor is CLAPS [21]. It provides a framework based on linear programming to optimize for latency reduction with constraints on anonymity and client location leakage. The framework outputs a routing policy for selected values of constraints that must be followed by all the nodes of the Tor network.

While the proposed solutions work well for Tor, they cannot be trivially applied to mixnets due to differences in the assumed threat model; local or partial network adversary for Tor and global adversary for mixnets. Thus, most of Tor's strategic routing mechanisms aim to reduce latency or enhance anonymity by avoiding paths that are or can be fully under the adversary's control, whereas mixnets routing scheme already assumes that all the paths on the Internet could be compromised. Additionally, routing in Tor is established per session, while in a mixnet, a new route is selected for each packet, requiring a different way of developing a routing schemes for both the systems.

Thus, a recent work, LARMix [20], developed methods tailored for reducing latency in mixnets while minimizing the impact on anonymity. LARMix provides an end-to-end solution with methods for arranging the mixnodes in layers, latency-aware routing policies, and load balancing techniques. However, we note that LARMix faces several challenges. It requires computationally intensive operations that grow exponentially with network size (refer Appendix. VI-C), making it impractical for large network sizes. Additionally, integrating LARMix with existing mixnets like Nym would require substantial design overhaul with complete reworking of all the mixnet components, limiting its deployment potential. Moreover, LARMix optimizations for the selection and arrangement of mixnodes does not significantly contribute towards latency reduction (as described in their own analysis [20]), while substantially increasing the computational load. LARMix also only optimizes latency within the mixnet, neglecting the client-to-mixnet latency, missing out on further latency reduction.

C. Problem Formulation

According to the anonymity trilemma formalized for anonymous communication systems [7], mixnets have a fundamental trade-off among latency and anonymity for a fixed volume of client traffic. This essentially mandates a direct and proportionate impact on anonymity for any corresponding reduction in latency. However, as characterized in LARMix, latency in mixnets consists of various components including mixing (μ), propagation ($\bar{l}$) and processing (δ), where μ and δ are added at each hop and $\bar{l}$ is added when traversing from one hop to another. While μ has a fundamental trade-off with anonymity, $\bar{l}$ does not directly impact it and thus can be potentially optimized for performance.[2]

Propagation latency ($\bar{l}$) can be further broken down into three components: client-to-mixnet ($l_{c,mix}$), mixnet (l_{mix}) and mixnet-to-destination ($l_{mix,d}$). The latency $l_{c,mix}$ and $l_{mix,d}$ are incurred when the packets traverse from the sender to the first hop in the mixnet and from last hop of the mixnet to the destination respectively. Whereas l_{mix} is the latency incurred when traversing hops within the mixnet. This work aims at reducing the latency from the client to the last hop in the mixnet ($l_{c,mix} + l_{mix}$) which is a step beyond LARMix that targets reducing only l_{mix}. We call this combined latency l_{cmix}

III. Approach

A. Design Goals and System Model

The primary objective of this work is to develop methods for reducing propagation latency in the mixnet while minimizing the impact on anonymity. This is achieved by designing novel routing strategies to bias path selection towards faster routes. We achieve the primary goal with constraints on (1) computation load for calculating routing policies and (2) the mixnet following a layered topology, where a total of N mixnodes are arranged in L layers, with each layer consisting of W mixnodes ($N = L \times W$). The constraint for limiting the computational load on the entities in the network is essential for making the approaches practically deployable. Thus, LAMP does not consider the mixnode selection and arrangement algorithm as developed in LARMix, which contributed to the computational load without a substantial gain in latency or anonymity (as reported in LARMix's evaluation). Whereas following a layered mixnet topology is motivated by the recent deployment of such mixnets in the real-world [8], [17] with the additional advantage of scaling the anonymity sets with an increase in the number of users.

Note that LAMP considers optimizing $l_{c,mix}$ in addition to l_{mix} to further minimize end-to-end latency (unlike LARMix that focuses only on l_{mix} and thus only offers a limited reduction of latency). This requires trading off equal load constraints, as some first hops will be selected more frequently than others, depending on clients' locations and choices. Note that selecting entry hops with disproportionate probability could impact other dynamics of the system, such as potential

[2] δ is negligible in comparison to μ and $\bar{l}$ and is thus not considered.

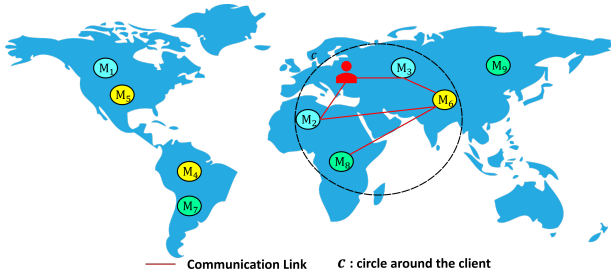


Fig. 1: SC: In this method, a client creates a circle (bounded by latency) around itself to include mixnodes from the mixnet, providing low-latency connections (different node colors represent them being in different layers).

targeting of popular first hops by the adversary and lower mixing anonymity at unpopular first hops. While targeting of certain nodes is harder to prevent, lower mixing anonymity at first hop could be compensated with higher mixing at subsequent hops minimizing impact on the end-to-end anonymity.

B. Threat Model

Mixnets are evaluated against two primary adversary types. Firstly, there is the *global passive adversary*. This adversary has the capability to observe all communication links within the mixnet, allowing it to monitor the packets entering and leaving a particular mixnode. However, it does not have visibility in the mixing operations performed at each mixnode. Thus, the adversary can probabilistically attempt to map the input and output packets in the mixnet. The mapping success can be quantified by measuring the entropy over the probabilities of possible senders for any given output message.

The second type of adversary is the *mixnode adversary*. This adversary aims to strategically compromise a fraction of nodes in the mixnet to reduce the anonymity of messages or, in some cases, completely de-anonymize the messages. This can be achieved by maximizing the chances of being present in as many end-to-end paths as possible. This adversary could exploit the biased routing developed in LAMP to obtain an unprecedented advantage in intercepting client paths.

Our analysis considers both types of adversaries and thoroughly assesses the efficacy of approaches developed in LAMP. Note that this is the standard threat model against which mixnets are evaluated. We keep an active adversary that can manipulate and interfere with ongoing connections as out of the scope of our current study.

C. Routing Methodologies

We propose three distinct routing strategies in LAMP. We assume a given layered mixnet in which mixnodes have already been assigned in different layers following the default random mixnode arrangement policy. Routing strategies in LAMP are thus modular and can be applied to any given topology. [3] We now describe the routing approaches.

1) Single Circle (SC): As the name suggests, this approach requires defining a logical circle around the client based on latency to other mixnodes. Thus, the radius of the circle r defines the boundary within which the client would select the mixnodes for creating the routes. Additionally, to ensure that even if the client selects a very small r it still has some minimum number of mixnodes to select from, we define another parameter α, where $0 \leq \alpha \leq 1$. α represents the fraction of the closest mixnodes from every layer that must be available for selection.

Consider the set of all the available mixnodes in a mixnet to be denoted by S. We introduce a latency function G, which takes as input a set of nodes and returns the latency between them. Additionally, we define a ranking function R that assesses the closeness (in terms of latency) between two nodes; for instance, if a node n_1 has latencies of 5 ms, 10 ms, and 30 ms from nodes n_2, n_3, and n_4 respectively, then the ranks of n_2, n_3, and n_4 relative to n_1 would be 0, 1, and 2. A client thus obtains the pool of nodes to construct a path based on $A \cup B$ where A and B are defined in Eq. (1) and (3), where C_i is the i-th client, M_j is the mixnode j, and B^l is defined for α percent of the closest mix nodes in layer l.

$$A = \{M_j \quad | \quad G(C_i, M_j) \leq r\}, \tag{1}$$

$$B^l = \{M_j \quad | \quad R(C_i, M_j) \leq \alpha \cdot W \wedge M_j \in l\}. \tag{2}$$

$$B = \cup_{l=1}^{l=L} B^l. \tag{3}$$

In simple terms, potential mixnodes are the ones capable of providing a link delay less than r from the client, or they belong to the α closest fraction of mixnodes to the client.

After obtaining the set of low-latency mixnodes to choose from, the client can construct a path for routing its packets. To construct such a path among the available mixnodes, there could be various strategies. Thus, we define a function F as the routing function, which takes the current hop (starting from the client) and provides a probability distribution for selecting the next hop from the current hop. The routing function F can be realized in three different ways: (1) uniform routing, (2) routing proportional to latency, and (3) LARMix routing.

Equation (4), represents the uniform distribution based routing, where $|S_N|$ is the cardinality of the set of potential nodes as the next hop (x) given that the current hop is $x^- = h$. Similarly, equation (5) represents the proportional routing function, where the nodes are selected based on their proportional latency to the clients; lower latency nodes are given more weight and vice versa. The third strategy of routing within SC is based on LARMix [20] formula (6), where τ is a randomness parameter ranging from 0 to 1, transitioning from fully deterministic to random routing. For evaluation purposes, we use specific values of τ as prescribed in LARMix. In practical scenarios any of the the three intra-routing scheme can be used, depending on the desired latency tolerance. Specifically, uniform routing is suitable for latency-tolerant applications as this approach does no effort to minimize latency within the circle, while proportional and LARMix based routing are better suited for applications requiring tighter latency constraints as these approaches further bias selection towards faster routes. Note that LARMix based routing allows for finer control over the optimization with the help of tuning parameter τ and thus can potentially better lower the latency (with lower values of τ) in comparison to proportional routing.

[3] If needed, the network designer can adopt the mixnode selection and arrangement strategies as defined in LARMix.

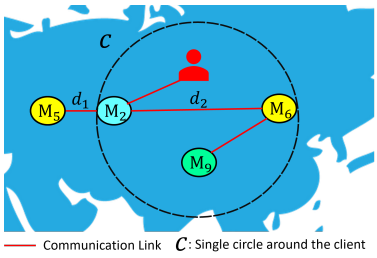


Fig. 2: The figure represents a scenario in SC approach where despite M_2 and M_6 being close to the client incurs $2x$ more latency when sending data from M_2 to M_6. Note that selecting M_5 after M_2 would have been a better choice, which is what would happen in the MC.

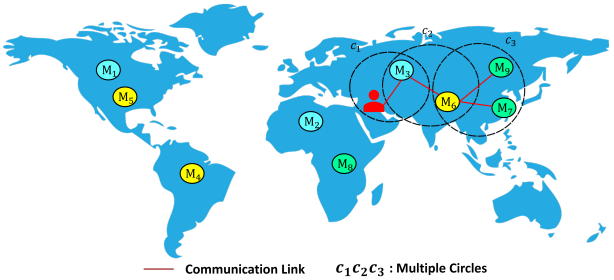


Fig. 3: MC: In this approach, a bounded latency circle is created at each hop starting from the client, to select appropriate low latency next hops (different node colors represent them being in different layers).

$$F\left(x|x^- = h\right) = \frac{1}{|S_N|}, \tag{4}$$

$$F\left(x|x^- = h\right) = \frac{\frac{1}{G(h,x)}}{\sum_{X\in S_N}\frac{1}{G(h,X)}}, \tag{5}$$

$$F\left(x|x^- = h\right) = \frac{\left(\frac{1}{e}\right)^{R(h,x)\frac{(1-\tau)}{\tau}}\left(\frac{1}{G(h,x)}\right)^{(1-\tau)}}{\Sigma_{X\in S_N}\left(\frac{1}{e}\right)^{R(h,X)\frac{(1-\tau)}{\tau}}\left(\frac{1}{G(h,X)}\right)^{(1-\tau)}}. \tag{6}$$

As an example, consider Fig. 1 where we have 9 active nodes layered in three layers of a mixnet, with M_1, M_2, and M_3 in layer one, M_4, M_5, and M_6 forming layer two, and the remaining mixnodes in the third layer. We fix $\alpha = \frac{1}{3}$, meaning that in the circle, there will be at least one node from each mixing layer. Let us assume setting r results in a circle where $A \cup B = \{M_2, M_3, M_6, M_8\}$ are available for clients to make message routes. In this scenario, the client has two options for forwarding the messages to the first layer: either through M_2 or M_3. Applying uniform routing results in choosing any of them with a probability of $\frac{1}{2}$, while LARMix or proportional routing will prefer the selection of closer mixnodes with a higher probability. After choosing the first mixnodes in the first layer, the client has one option in the second layer (M_6) to choose as the second mixnode, and further she chooses the third mixnode (M_8) with a probability of one as the third hop. Note that changing r may include M_9 in the circle, or may exclude M_2, but as applying $\alpha = \frac{1}{3}$ for any value of r, we will have at least M_3, M_6, and M_8 in the circle.

2) Multiple Circle (MC): The SC routing is a simple and easy-to-deploy scheme as it does not require the information about latency between different mixnodes in the network. The client just needs to measure the latency to different mixnodes and, based on the value of r defined for this scheme, select among available mixnodes. However, the SC strategy may not always be optimal. Consider Fig. 2, where despite the three mixnodes having low latency to the client, the latency among them can be as high as twice the latency to client if they fall on the opposite ends of the circle. The SC approach can thus be further improved if the latency dataset between different sets of mixnodes is available to the client.

The MC strategy, rather than confining itself to a singular circle centered around the client, considers a circle around each hop in the end-to-end path for selecting appropriate next hops. This method facilitates an approach to optimizing the selection process across the layers of network.

Let S_k denote the set of available mixnodes within layer k (with $S = \sum_{k=1}^{L} S_k$), adhering to the definition provided in Section III-C1. The mixnodes available for selection in the layer k, are obtained with $A_k \cup B_k$, as defined in equations (7) and (8). Here, N_i represents the previous hop already selected in the path. and for the first layer N_i is assumed to be the client. Once a circle with available mixnodes for a layer k is fixed, the hop can be selected based on the three approaches outlined in (5). (6) and (7). The process repeats till all the hops in the path are selected.

$$A_k = \{M_j \quad | \quad G\left(N_i, M_j\right) \leq r \quad and \quad M_j \in S_k\}, \tag{7}$$

$$B_k = \{M_j \quad | \quad R\left(N_i, M_j\right) \leq \alpha \cdot W \quad and \quad M_j \in S_k\}. \tag{8}$$

As an example consider, Fig. 3, where a client uses the MC strategy with the defined radius r and $\alpha = \frac{1}{3}$. This setup ensures that at least one mixnode is available for selection in each circle. In this scenario, the first circle is formed around the client, which includes possible mixnodes to select as the next hop. In this example, only M_3 lies in the first circle around the client, so this mixnode will be selected as the first hop. For the second hop, a circle is formed around M_3, which includes the mixnodes that can be selected as the second hop; in this case, it is M_6. Finally, a circle around M_6 will determine the mixnodes that can be selected in the third layer; in this example, M_7 and M_9 are available as the third hop. If the routing within circles is uniform, then either M_7 or M_9 can be selected as the last mixnode in the message route. However, using LARMix or proportional routing will prefer M_7 with a higher probability, as it is closer to M_6. This scenario holds as long as the radius r for all cells is fixed. Increasing this radius can include more mixnodes like M_2 and M_8 in the routing paths, while decreasing r may exclude any of M_7 or M_9.

3) Regional Mixnets (RM): Another strategy to reduce latency in the mixnet is to create multiple smaller (region-specifc) mixnets instead of a single global mixnet. The idea is for the clients to select the mixnets closer to their location to route their packets. Since the mixnodes within any given region would be geographically close to each other,

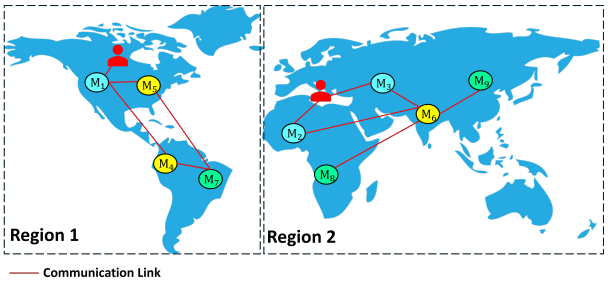


Fig. 4: RM: This approach partitions the global mixnet into smaller regional mixnets. As depicted, the clients will route their packets via their respective regional mixnets.

the latency can be significantly minimized. There could be multiple criteria for dividing the mixnets such as based on client density, mixnode density, traffic volume or geographical regions. For adhering to our goal of the developed approaches being *simple to deploy*, we selected a geographical division of mixnets, as client/mixnode density or traffic volume can significantly change from time to time, requiring a dynamic coping mechanism.

In this approach we consider a continental division of mixnet, with each continent (*e.g.,* North America, Europe, Asia) having a mixnet of its own. A key element in making this approach work is to have a reliable mechanism to know the location of mixnodes, so that they can be appropriately assigned to a mixnet. We rely on Verloc [13] for the same, which is an approach to reliably determine the location of a mixnode in a distributed manner, even in the presence of malicious nodes. Concretely, consider $S_R = \{M_j | D(M_j) \in R\}$ as the set of available mixnodes within region R, where the function D assesses a node and returns its latitude and longitude. Thus, client C_i first identifies its own regional location and then selects RM closest to its location. The routing within the RM can follow the default uniform routing, the LARMix approach or the proportional approach as defined in previous subsections (we evaluate each of them in Sec. IV).

Fig. 4 depicts a sample RM scenario where the original mixnet is partitioned into two regions. The first region includes North and South America, where there are four mixnodes ($M_1, M_4, M_5,$ and M_7) available for a client to create a message route. The node M_1 is part of layer one, M_4 and M_5 is part of layer two and, M_7 in for layer three. Therefore, a client can select among two low-latency paths, $M_1 M_4 M_7$ or $M_1 M_5 M_7$. Based on the routing, either of the paths will be selected with different probabilities. If the routing is uniform, the paths will be selected with a probability of $\frac{1}{2}$.

IV. Evaluation

In this section we evaluate the three approaches under various experimental settings. We first define the evaluation metrics. Thereafter, we detail the experimental setup and finally present the results of the conducted experiments.

A. Metrics

We primarily rely on two types of metrics for evaluation. The analytical metrics are used to measure the isolated effect of *only* the biased routing strategies on anonymity and latency. The simulation-based metrics are used to evaluate the end-to-end anonymity and latency of the mixnet combining the effect of routing as well as mixing.

1) Analytical Metrics: In the analytical approach, we compute all possible routing paths within the mixnet along with their probabilities of selection. Subsequently, we calculate the link latencies associated with sending messages through these paths. This analysis enables us to determine the average latency from the client l_{cmix} and the entropy of the transformation matrix $\mathsf{H}(T)$. As defined in LARMix, $\mathsf{H}(T)$ matrix stores the probability for each node in a given layer to send traffic to a node in the subsequent layer. As $\mathsf{H}(T)$ only stores the routing probability, it captures the isolated effect of the biased routing.

2) Simulations Metrics: In the simulations, a discrete event simulator is implemented as a mixnet system, where messages are actually sent from a client, traversing the mixnet and incurring the propagation latency along with mixing among other packets at each hop. The end-to-end latency in this case is calculated by recording the time spent by the packet in the mixnet (consisting of both l_{cmix} and μ). For calculating anonymity, we sample certain target messages as input to the mixnet and then find the probability distributions of such messages being one of the output messages coming out of the last layer in the mixnet (as also outlined in [5], [16]). Entropy is then calculated using the probability distributions for the target messages. Each entropy sample (denoted by $\mathsf{H}(m)$) is plotted on a boxplot to evaluate the overall anonymity.

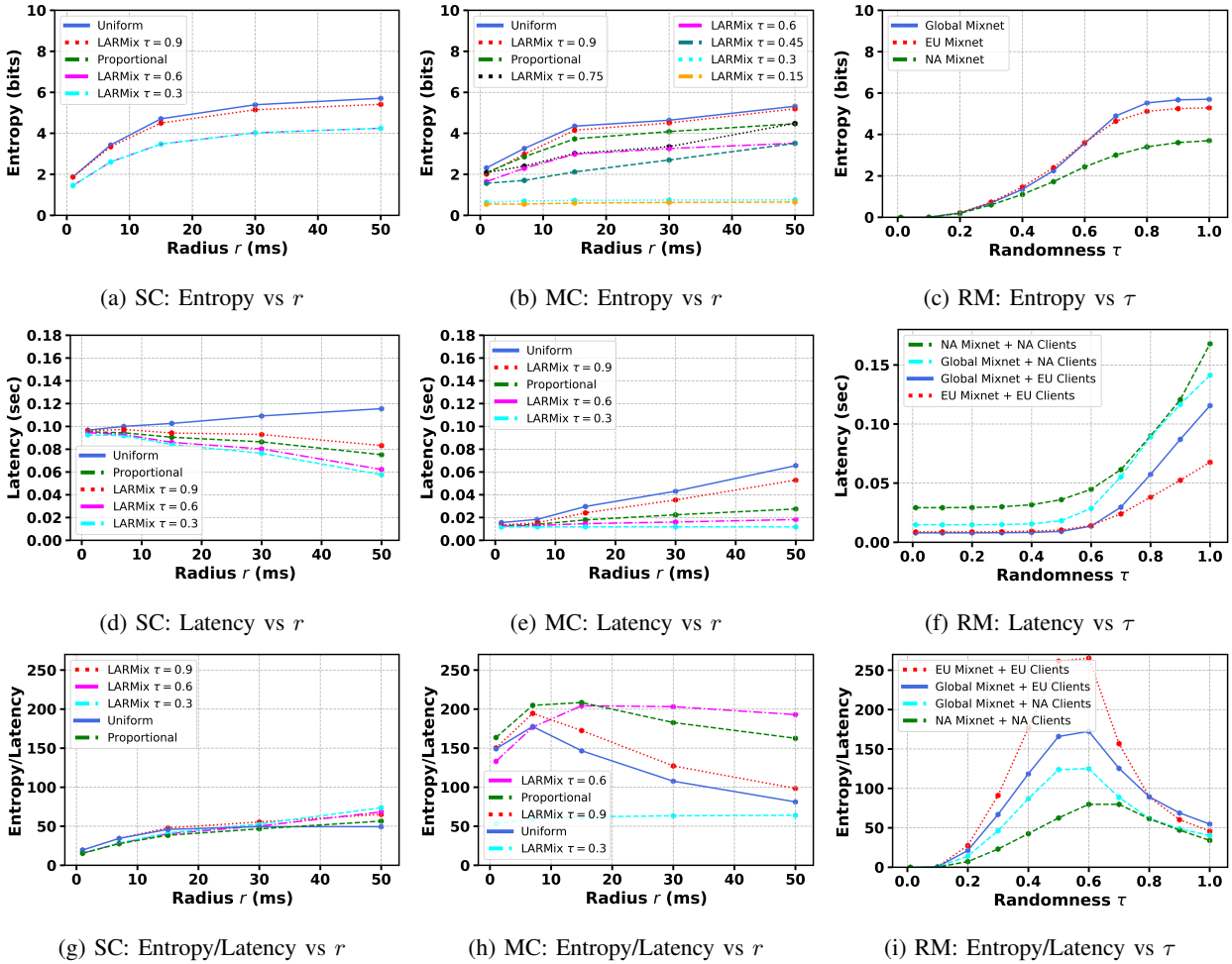
B. Experimental Setup

We instantiate the mixnet with some baseline parameters that remain the same across experiments unless otherwise explicitly stated. The mixnet is formed using the layered, stratified topology, where each of the four layers consists of 60 nodes, culminating in a total network size of 240 nodes (first layer being that of the client). The mixing delay (μ) is set at 50 ms per mixnode, and follows a Poisson distribution. In our simulation, we consider an average influx of 20,000 messages per second in the mixnet. This volume is large enough to effectively simulate the operational dynamics of the mixnet.

We build the mixnode latency dataset for evaluation using the Verloc [13] measurement framework as deployed on the Nym network. We collect the data of latency among mixnodes as part of the Nym mixnet where we query every mixnode on port 1790 and access the verloc measurement data that should ideally consist of measurements to all other mixnodes in the network. However, due to various reasons (such as using an older version of mixnode), not every node in the Nym network has the verloc measurement functional, and thus, we filter out such nodes. We have a final dataset of 253 nodes that all have a latency measurement to each other in the network. We plan to make the simulator code, analysis scripts, and the latency dataset public upon acceptance.

C. Experimental Results

1) LAMP Routing Evaluation: **Analytical:** Fig. 5 depicts the analytical anonymity, latency, and tradeoff (entropy/latency or E/L) for the three routing approaches. The Fig. 5a, Fig. 5d and Fig. 5g on the left side represent the results of the SC



(a) SC: Entropy vs r (b) MC: Entropy vs r (c) RM: Entropy vs τ

(d) SC: Latency vs r (e) MC: Latency vs r (f) RM: Latency vs τ

(g) SC: Entropy/Latency vs r (h) MC: Entropy/Latency vs r (i) RM: Entropy/Latency vs τ

Fig. 5: Analytical evaluation of different routing strategies.

approach, where we vary r between 0 and 50 ms. We set $\alpha = 2\%$, implying that, in the case of having $W = 60$, at least two nodes per layer closest to the clients are considered inside the SC. This ensures the existence of at least 8 paths for routing, if no mixnodes fall inside the chosen r. The Fig. 5b, Fig. 5e, and Fig. 5h represent the results for MC approach, where $\alpha = 2\%$ and r is again varied between 0 and 50 ms. The remaining Fig. 5c, Fig. 5f, and Fig. 5i represent the results for RM approach. We divide the mixnodes among two primary regions, North America (with 13 nodes per layer), Europe (39 nodes per layer) because in the existing Nym mixnet, the majority mixnodes (52/60 per layer) exist in these continents. Further, for effective comparison we assume two scenarios for the clients where they can either select from the default global mixnet or the RM (NA and EU). Unlike the previous routing approaches, in RM the variable is τ as adapted from LARMix, controlling the bias in routing.

Now we will describe the individual results of the three approaches separately and then conclude with a comparison.

SC: In the SC approach, an increase in r leads to a corresponding rise in $\mathsf{H}(T)$ (refer Fig. 5a), confirming that higher number of mixnodes within the circle leads to more choices to select from, resulting in an overall higher anonymity. However, even within the circle, one can choose nodes uniformly, in direct proportion to latency, or based on LARMix routing. As observed from the results, when $r = 50$ms, the uniform routing and LARMix with $\tau = 0.9$ results in higher entropy (≈ 5.5 bits), whereas, the proportional routing approach or LARMix with $\tau = 0.3$, 0.6 result in lower entropy (≈ 4.2 bits).

Similarly, the variation of l_{cmix} with r is represented in Fig. 5d. For uniform routing within the circle, increasing r results in higher average latency. This is to be expected, as a higher r leads to more random mixnodes with high latency available for routing, eventually increasing latency. However, for approaches such as proportional routing and LARMix that bias the selection within the circle, the results are counterintuitive. An increase in r results in lowering the l_{cmix}. On further investigation, we found out that an increase in r results in more nodes available for selection. The new nodes added to the circle may not be close to the client but are closer to each other (in terms of latency), facilitating low-latency paths due to biased routing. Using LARMix with $\tau = 0.3$ maximizes the reduction in latency in our experiments.

We additionally attempt to quantify the tradeoff between latency and anonymity to better understand the advantage of a routing approach. For this, we calculate the ratio of Entropy to Latency (E/L). The value of this ratio should be as high as possible because increasing entropy is beneficial

for anonymity, and lowering latency is good for reducing end-to-end latency. Fig. 5g represents the E/L ratio for the SC approach. We observe that increasing r consistently improves the tradeoff for all the approaches. However, for maximum benefit, either LARMix with $\tau = 0.3$ or LARMix with $\tau = 0.6$ should be selected, resulting in a value of ≈ 80.

MC: Fig. 5b depicts the entropy for the MC approach. Similar to the SC approach, we observe an overall trend of increasing $\mathsf{H}(T)$ for a corresponding increase in r. The different routing strategies employed within the circle (uniform, LARMix, and proportional) also exhibit similar behavior. However, there is an exception for LARMix with $\tau = 0.3$, where the entropy does not increase with r. This is because with $\tau = 0.3$, the routing is highly biased, and thus, the selection of nodes within a circle is almost deterministic despite having options. We confirm this with additional experiments for more values of τ ($0.15, 0.45$ and 0.75). We clearly observe from the results that going any lower that $\tau = 0.3$, for instance $\tau = 0.15$ results in a similar trend with no entropy increase despite an increase in r. Whereas, for values higher than $\tau = 0.3$ ($\tau = 0.45$ and $\tau = 0.75$), the results were as expected, *i.e.,* increasing entropy with a corresponding increase in r where the intensity of increase is more for a higher value of τ.

The variation of l_{cmix} with r is shown in Fig. 5e, where we can clearly observe an increase in latency with an increase in r. This is because the MC approach is already optimized to select low latency nodes for a given r. Thus, any increase in the available mixnodes (with an increasing r) only adds to the average latency. However, we also observe that low values of τ significantly minimize the end-to-end latency, coming down as low as 20 ms on average for $\tau = 0.6$. Overall, the MC approach can provide drastic reductions in latency. We observe an interesting trend for the E/L ratio as depicted in Fig. 5h, which initially increases, peaks for a certain value of r, and then decreases. Overall, LARMix routing within the circle with $\tau = 0.6$ provides the best tradeoff for higher values of r, closely followed by proportional routing, which performs better for low r. Note that the most biased routing (LARMix with $\tau = 0.3$) and totally unbiased routing (uniform) are at the bottom of the curve.

Sensitivity of α: For the above evaluation of SC and MC we assumed a fixed $\alpha = 0.02$. We now evaluate the impact of varying the value of α on the latency and anonymity. We perform experiments for α values of $0.02, 0.05, 0.1$ and 0.2 (differing by an order of magnitude) while keeping r fixed at $15ms$. The results are detailed in Tab. I for different intra-circle routing approaches. Note that increasing α should essentially have the same effect as increasing r because increasing the value of either of the parameters results in increased availability of nodes for selection, where α acts as a lower bound of the nodes that have to be considered for selection. Thus, the results follow the same trend as in baseline experiments where for the MC approach, we observe that increasing α leads to an increase in average latency. The increment is more for uniform routing ($34ms$ to $48ms$), slightly less with proportional ($19ms$ to $26ms$), and constant for LARMix with lower values of $\tau = 0.3$ ($11ms$). Whereas, for the SC approach, increasing α leads to a decrease in latency. As detailed in baseline results this happens because in SC routing, all the nodes in a path are selected based on their proximity with the client and not based on the proximity among the nodes themselves (as followed in MC). Thus, increasing α for SC increases the available node choices and thus increases the chances of combinations where selected nodes in a path would be close to each other. For MC, since the next hop selection is already based on its proximity to the previous hop, increasing α only adds relatively higher latency nodes to the available choices and thus increases the latency on average.

The entropy inevitable increases with α for both the approaches as a higher value of α ensures more nodes for selection, leading to increased path diversity and thus a corresponding increase in routing anonymity. The findings hold across different values of r and α.

RM: We now present the results for the RM routing. Remember that based on the deployed Nym network dataset, we considered two regions for evaluation, namely North America (NA) and Europe (EU). Fig. 5c depicts the entropy for EU, NA, and the global mixnet. Overall, as expected, the entropy increases with τ. The global mixnet provides the maximum entropy as it has the most mixnodes available for routing, closely followed by the EU mixnet.

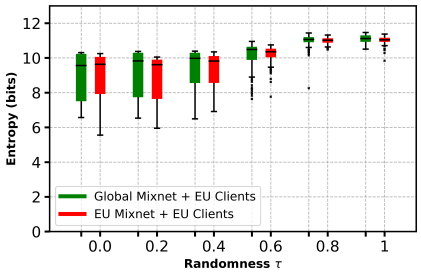
For calculating latency, we consider four scenarios where a client in the EU or the NA can select either the global mixnet or the mixnet close to its location *i.e.,* EU or the NA, respectively. Fig. 5f represents latency for all four cases. Across all cases, we observe an increase in l_{cmix} with an increasing value of τ. We interestingly observe that an EU client using the EU mixnet results in the least latency for all values of τ, highlighting the benefit of selecting mixnodes within the same region in comparison to the global mixnet. However, an NA client selecting an NA mixnet incurs, on average, higher latency in comparison to the NA client selecting the global mixnet. This phenomenon results from relatively fewer nodes to select from in North America, contributing to higher variance in the observed latency. Overall, RM can be advantageous for reducing latency, particularly when the concentration and number of mixnodes are higher in a region (as seen in the EU). Thus, if the mixnets have relatively equal numbers of mixnodes in each region, it can lead to an ideal scenario for optimizing latency within each region. However, a disproportionate distribution of mixnodes can be slightly disadvantageous for regions with low mixnodes.

The E/L tradeoff for the approach is quantified in Fig. 5i. Notably, for all cases, the E/L ratio forms a convex curve, signifying an optimal value for each case. The best tradeoff is achieved for EU clients using the EU mixnet, which has a maximum value close to 270 for $\tau = 0.6$ (corresponding to the entropy of 3.8 bits and the latency of 15 ms). For NA clients, selecting the global mixnet provides a better tradeoff, achieving a maximum at $\tau = 0.6$.

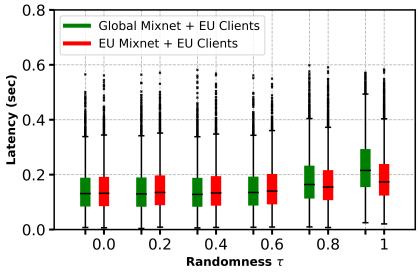
Summary: In our analysis, the SC approach is the simplest to deploy. The clients need to measure the latency from itself to all other mixnodes and select the ones that fall within r to form an end-to-end path. We observe that this approach can bring l_{cmix} to 50 ms for an entropy of ≈ 4.1 bits. However, in comparison to the other two approaches, the reduction is relatively low. This is because the selection of nodes within the circle is not always optimal as the nodes, despite being close to the client, can, in many cases, be far away from each other.

TABLE I: Sensitivity of α: Impact on latency and anonymity for SC and MC approaches with different routing strategies.

α	0.02				0.05				0.1				0.2			
Metric	Latency (ms)		Entropy (bits)		Latency (ms)		Entropy (bits)		Latency (ms)		Entropy (bits)		Latency (ms)		Entropy (bits)	
Approach	SC	MC	SC	MC	SC	MC	SC	MC	SC	MC	SC	MC	SC	MC	SC	MC
LARMix $\tau = 0.3$	80	11	3.4	0.75	70	11	3.7	0.76	59	11	3.9	0.76	41	11	3.9	0.76
LARMix $\tau = 0.6$	84	16	3.4	3.1	73	17	3.7	3.1	61	18	3.9	3.4	42	20	3.9	3.6
Proportional	88	19	3.4	3.9	77	21	3.7	4.1	64	22	3.9	4.2	47	26	3.9	4.5
LARMix $\tau = 0.9$	92	27	4.5	4.4	84	32	4.8	4.7	72	35	5	4.9	54	41	5	5.1
Uniform	100	34	4.7	4.6	97	39	5.1	4.9	91	42	5.4	5.1	81	48	5.5	5.2



(a) Entropy (H(m)).



(b) Latency.

Fig. 6: RM simulations.

This is improved in the MC approach at the expense of more complexity in terms of measurement and state management. The client requires access to the inter-mixnodes latency dataset for routing, using which it can further lower the network's latency. Specifically, it can bring down the overall latency close to 20 ms on average with 3.8 bits of entropy in the best case. The RM approach under favorable conditions can significantly reduce latency but can also lead to higher latency if the density of mixnodes in a region is low. As observed for the EU mixnet, latency on average can be 18 ms with an entropy of 3.8 bits.

Simulation: We now present simulation results, measuring the latency and entropy. In the simulation, we can sample a set of messages in the network and obtain the complete variation of values, plotted using a box plot. Note that simulation combines the effect of routing as well as mixing in the mixnet, essentially providing a complete picture of the tradeoffs. The latency values in simulations include both propagation and mixing latency, while entropy accounts for routing advantage as well as uncertainty due to mixing. The simulation results thus are also dependent on the average delay per mixnode ($\mu = 50$ ms) and the traffic volume (set to 20000 messages per second).

We present the results for RM in Fig. 6, where we study two experimental settings: EU client with an EU/global mixnet. The values in the box represent half of the samples (from 25 to 75 percentile), with the whiskers showing the range (10 to 90) and the dots representing the outliers. For entropy, overall, we observe a large variation for lower values of τ (refer Fig. 6a). Note that the entropy is non-zero even in the worst case ($\tau = 0$) due to traffic volume and mixing on every node. For higher values, the variation drastically decreases, with only the outliers having an entropy less than 10, signifying good anonymity enjoyed by each packet. The global mixnet overall provides a slightly higher entropy (*e.g.,* 10.6 in comparison to 10.4 for $\tau = 0.6$) due to larger number of nodes available for selection, increasing randomness. As can be observed from Fig. 6b, we see a larger variation in samples for latency when compared to entropy. The best cases lead to negligible latency, whereas the worst case may lead to more than half a second of latency. This is likely due to the variability of the long tail exponential distribution for selecting mixing delays.

The end-to-end latency, however, shows that the latency provided by the EU mixnet is slightly lower than that of the global for higher values of τ (closely following each other for lower values of τ). This is similar to the observation from analytical results. Thus, having a RM helps in reducing latency without significant impact on anonymity.

For the SC and MC, we did not observe any notable deviations, resulting in findings similar to the ones observed in the analytical evaluation (MC providing lower latency than SC). We refer to the Appendix. A for further details.

2) End-to-End Latency Constraint: Having seen the general tradeoffs in latency and anonymity for different routing methods in the previous section, we now move towards evaluating the tradeoffs among different types of latency (propagation as well as mixing) when a constraint is given on end-to-end latency in the mixnet. Such an evaluation is useful when certain applications with a specific latency constraint are to be run over the mixnet. For instance, applications such as voice calls requires limits on the one-way delay for them to function (should be ideally less than 150 ms).

Let's denote the average end-to-end latency constraint to be met as $\bar{l}_{\text{e2e}}$, where $\bar{l}_{\text{e2e}}$ consists of the propagation delay from the client to the last hop in the mixnet (l_{cmix}) and the mixing latency (μ) on each mixnet hop (effectively $\bar{l}_{\text{e2e}} = 3{*}\mu{+}l_{cmix}$). For a given value of $\bar{l}_{\text{e2e}}$, we calculate the appropriate tradeoff factor (τ in case of RM and r in case of SC and MC) and the distribution among μ and l_{cmix}, to maximize the anonymity.

For conducting the evaluation we first calculate the value of l_{cmix} for varying values of r and τ. Remember that r and τ dictate the routing policy and thus only affects l_{cmix}. Then, for a given value of r (τ in case of RM) and their corresponding l_{cmix}, we assign the remaining budget ($\bar{l}_{\text{e2e}}$ - l_{cmix}) as mixing latency ($\mu = \frac{\bar{l}_{\text{e2e}} - l_{cmix}}{3}$). After obtaining the values of μ and

TABLE II: Delay constraint: Trade-offs between radius and mixing delay for SC and MC approaches, considering both proportional and LARMix as intra-circle routing.

Radius	1 ms				15 ms				30 ms				50 ms			
Routing	Proportional		LARMix		Proportional		LARMix		Proportional		LARMix		Proportional		LARMix	
Approach	SC	MC	SC	MC	SC	MC	SC	MC	SC	MC	SC	MC	SC	MC	SC	MC
$\mathsf{l_{mix}}$ **(ms)**	96	12	94	12	94	17	85	14	92	22	80	16	83	27	62	18
μ ms	34	62	35	62	35	60	38	61	35	59	39	61	38	57	45	60
H(m) **(bits)**	8.63	8.98	8.57	8.93	10.12	10.14	9.72	9.71	10.46	10.39	9.96	9.87	10.69	10.56	10.19	10.0
H(T) **(bits)**	1.87	2.08	1.44	1.64	4.49	3.72	3.47	2.98	5.14	4.08	4.02	3.25	5.41	4.46	4.24	3.51

TABLE III: EU/Global Mixnet and EU client: Trade-offs between randomness (τ) and mixing delay.

Randomness τ	0.0		0.2		0.4		0.6		0.7		0.8		0.9		1.0	
Regions	EU	Global	EU	Global	EU	Global	EU	Global	EU	Global	EU	Global	EU	Global	EU	Global
$\mathsf{l_{mix}}$ **(ms)**	9	7	9	7	9	7	14	13	23	29	37	56	52	86	66	113
μ ms	63	64	63	64	63	64	61	62	58	56	54	47	49	37	44	28
H(m) **(bits)**	9.59	9.56	9.61	9.61	9.95	9.88	11.02	10.91	11.61	11.5	11.82	11.67	11.79	11.52	11.73	11.23
H(T) **(bits)**	0	0	0.2	0.2	1.42	1.34	3.58	3.57	4.63	4.89	5.11	5.51	5.25	5.67	5.28	5.7

r (τ for RM) we measure the anonymity both in terms of analytical ($H(T)$) and simulation ($H(m)$).

Table II describes the results obtained for the SC and MC approaches, where we fixed $\bar{l}_{\text{e2e}}$ at 200 ms. We evaluate both proportional and LARMix approach for routing within the circle. For SC, we can observe that increasing values of r results in a corresponding increase in $H(T)$, but a decrease in l_{cmix}. The remaining budget for μ thus increases with r, allowing for more mixing per each hop and hence leads to an increasing $H(m)$, as $H(m)$ is dependent on both r and μ.

Further, both $H(T)$ and $H(m)$ have higher values for proportional routing in comparison to LARMix routing (with $\tau = 0.6$), demonstrating that proportional approach is ideal for obtaining higher anonymity for a given latency constraint. Thus, overall we find that for a constraint of $\bar{l}_{\text{e2e}} = 200$ ms, the value of $r = 50$ ms and $\mu = 38$ ms with proportional routing within the circle provides maximum anonymity to messages for the SC scenario.

For the MC approach, We observed that $H(T)$ grows with r, as does l_{cmix}, reducing the overall latency budget for μ. Note that $H(m)$ is dependent on both r and μ, and despite μ decreasing for higher r, $H(m)$ still increases. This is because the reduction in μ is compensated by an increase in l_{cmix}, resulting in less biased routing and increased anonymity. Note that this is true for lower values of $\bar{l}_{\text{e2e}}$, where l_{cmix} notably affects anonymity. For higher $\bar{l}_{\text{e2e}}$, the budget for μ can be considerably high and thus able to achieve good anonymity even when the routing is totally unbiased. We see a similar trend for routing within the circle with the proportional approach outperforming LARMix. Here, as well, the best result is achieved with $r = 50$ ms with a maximum $H(m)$ of 10.56.

For RM, we need to find the appropriate value of τ and μ to maximize anonymity. We conducted experiments for different scenarios varying the client location and the mixnet region. We followed the same process of calculating l_{cmix} for different values of τ and then distributing the remaining latency as mixing delay μ. Then, we measured the entropy for different combinations of τ and μ.

We represent the results for $\bar{l}_{\text{e2e}} = 200$ ms and a global/EU mixnet with the clients being in the EU in Tab. III. We observe that an increase in τ results in a consistent increase in $H(T)$ due to a corresponding increase in the randomness of path selection. Such an increase is accompanied by higher values of l_{cmix}, leaving less room for μ. However, we observe an interesting trend for $H(m)$ where the value grows with an increase in τ till a certain point when it reaches its peak (with $\tau = 0.8$). Going any higher leads to a decrease in $H(m)$ because the l_{cmix} leaves very little room for μ to the extent that the gain in anonymity due to randomness in routing is lower in proportion to the loss in anonymity due to the decreased mixing latency (reduced by almost half when going from $\tau = 0.8$ to $\tau = 1.0$). Thus, the best trade-off is achieved with $\tau = 0.8$ with a $\mu = 54$ ms (note that the highest value of τ was 63 ms) and $H(m) = 11.82$. We observed similar findings when we tested the EU mixnet with an EU client, Here again, we observed the best tradeoff at $\tau = 0.8$ for maximizing anonymity. We observed similar trends for other client locations and mixnet combinations (refer Appendix. B1).

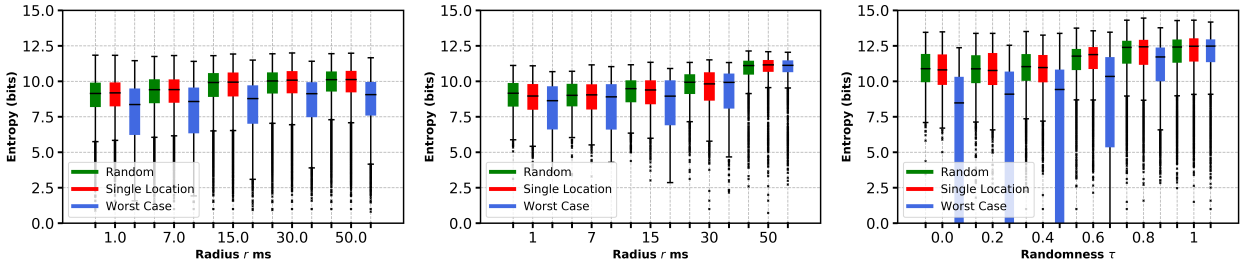
Moreover, we repeated the experiment for different values of $\bar{l}_{\text{e2e}}$ and observed that higher value of latency constraints provide good anonymity even when we do not optimize routing and make it completely random. However, careful consideration is required for tighter constraints as the routing randomness plays a crucial role in such cases in determining the best scenario for maximizing anonymity (as seen in the RM results).

V. Adversarial analysis

In this section, we analyze and quantify the adversary's advantage due to the developed routing schemes that bias path selection towards faster routes. Remember that we consider a global passive adversary that has the capability to analyze traffic on all links over the Internet. Additionally, we consider the adversary to be able to add a certain number of malicious nodes in the network. Such malicious nodes will know the exact mapping of the input and output packets of their node. We start by defining the metrics of adversarial success, followed by different adversarial strategies, and finally, the results.

A. Metrics

If an adversary can identify the originator and destination of any given message, it can be considered completely deanonymized. To do so over a mixnet system requires controlling all the hops in the end-to-end path that the packet



(a) SC: Entropy ($H(m)$) vs r (b) MC: Entropy ($H(m)$) vs r (c) RM: Entropy ($H(m)$) vs τ

Fig. 7: Adversary advantage for different strategies with varying bias in routing.

traverses in the mixnet. Such paths in which all the nodes are corrupted are known as end-to-end corrupted paths. We calculate the *fraction of corrupted paths (FCP)* in the network as a metric for measuring the adversary's advantage. For any given corruption ratio (C), FCP should be as low as possible to limit the possibility of a client message being deanonymized. For scenarios with partial path compromise, we quantify the adversary's advantage by calculating the corresponding decrease in message entropy ($H(m)$).

B. Adversarial Strategies

Worst Case: The worst-case scenario for deanonymization involves the adversary being able to strategically choose mixnodes to maximize the fraction of fully corrupted paths. However, for a mixnet with N mixnodes among which C number of nodes can be corrupted, there are $\binom{N}{C}$ different possibilities for corrupting mixnodes in a mixnet. Finding the best strategy among these cases requires an exhaustive search, which is an NP-hard problem. There are different approaches outlined in LARMix used to approximate the worst-case. We specifically use the greedy approach to find the worst cases. In practical scenarios, with mixnet arrangements periodically changing (every hour in Nym), it is almost infeasible to enumerate all possibilities in the given time frame. However, the worst-case could also occur by chance, and thus, we quantify the adversarial advantage in such a scenario.

Single Location: In this strategy, we assume that the adversary deploys its C mixnodes in close proximity to each other, potentially within the same geographical region. Since the routing policy favors the selection of nodes closer to each other to minimize latency, this strategy could help the adversary gain an advantage in compromising more end-to-end paths.

Random: This is a simple strategy in which the adversary randomly corrupts C mixnodes.

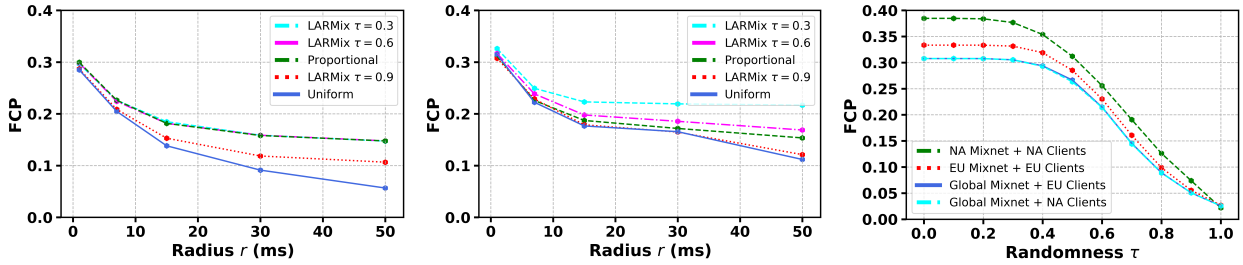
C. Results

We now present the results of the three experiments that we conducted. The first experiment quantifies the adversarys' impact on the anonymity of individual messages. It captures the variation of adversarial advantage (with a box plot) for partial and completely corrupted paths with varying adversary strategies and a fixed corruption rate of 20%. The next experiment captures the FCP for the adversary with a fixed corruption of 20% and with varying biases in the three routing approaches. Lastly, we fix the bias in routing ($r = 30$ ms for both SC and MC while $\tau = 0.6$ for RM) and vary the corruption rate (from 3% to 20%) to observe the adversarial gain.

1) Entropy vs. LAMP Routing: In this experiment, we calculate the entropy of messages $H(m)$ for varying bias factors in the routing approaches. Fig. 7 represents the results for SC, MC and RM approaches. Fig. 7a shows the results for the SC approach with LARMix routing ($\tau = 0.6$) within the circle. We can observe that across adversarial strategies, an increase in r results in a corresponding increase in entropy. If we compare among different strategies, we can observe that the worst-case adversary can force the lowest entropy at an average of 8.9 bits (for $r = 50$ ms). The average entropy is ≈ 1 bits lower than the random and single location adversary, implying that a worst-case setting will not significantly impact the anonymity of messages on average. However, analyzing the complete distribution of entropy values shows that the worst-case can lead to far lower entropy in certain corner cases with a lower 10-percentile close to 4 bits as compared to ≈ 7 bits for single location and random. Among single location and random strategy, the former is only slightly worse than the latter, with approximately the same averages but with slightly low entropy for lower percentiles in the case of single location.

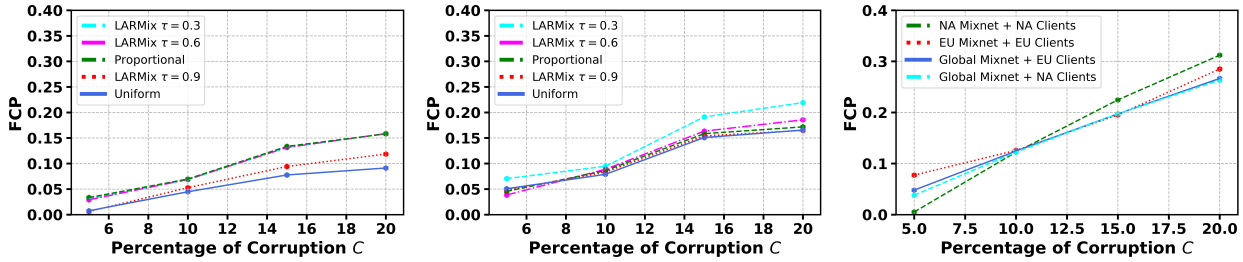
In the MC approach (refer Fig. 7b), we observed a similar trend of increasing entropy for a reduced bias in routing. Among different strategies, the worst-case entropy is again consistently less than the other strategies, with the difference among them decreasing with a higher r—becoming almost comparable for $r = 50$ ms. This closing gap between worst-case and other approaches could be attributed to reduced randomness for path selection. We also observe that, unlike the SC approach, there is a notable difference between single location and random adversary, with the former strategy showing lower entropy on average as well as in different percentiles. This is because MC directly utilizes the proximity among mixnodes for routing (unlike SC where only the proximity of mixnodes to the client is considered for routing) and thus is influenced more by the adversary having nodes in a single region. Note that we observe the same trends for other types of routing within the circle (proportional, random, and LARMix with other values of τ) for both the SC and MC approaches.

For the RM approach, the bias in routing is varied by changing the value of τ. Fig. 7c represents the results for the EU mixnet. We can observe that for the lower values of τ (≤ 0.6), the worst-case entropy is significantly lower with the 25-quartile reaching zero for $\tau \leq 0.4$, implying that in RM,



(a) SC: FCP vs r (b) MC: FCP vs r (c) RM: FCP vs τ

Fig. 8: Worst case FCP for varying randomness in routing.



(a) SC: FCP vs Corruption rate (b) MC: FCP vs Corruption rate (c) RM: FCP vs Corruption rate

Fig. 9: Variation of adversary advantage (in terms of FCP) for different corruption of mixnodes in the network.

the value of τ should be higher for realistic deployments in order to minimize the chances of unfavorable conditions for the users in the worst-cases. Single location strategy is slightly worse than the random as LARMix routing biases the node selection for mixnodes closer to each other.

2) FCP vs. LAMP Routing: In this experiment, we calculate the fraction of fully corrupted paths (FCP) for the three routing approaches (SC, MC, and RM) under different adversary strategies (random, single location, and worst case). Fig. 8 represents the results for the three routing schemes with the worst-case adversary.

For the SC approach (Fig. 8a), we observe that the FCP decreases with an increasing r due to increased randomness in routing, reducing the chances of the adversary corrupting a complete path. Notably, the type of routing within the circle also impacts the adversarial advantage. The random routing has the lowest FCP (9% fully corrupted paths on average) among all approaches closely followed by LARMix with $\tau = 0.9$ (12%). However, the proportional and LARMix routing with $\tau = 0.6$ and $\tau = 0.3$ have the highest FCP at 16%.

Fig. 8b presents the results for MC approach. The decrease in FCP in MC is not as prominent as in SC with 17% corrupted paths for uniform routing at $r = 30$ ms as compared to 9% for SC. This implies that MC leads to overall larger FCP due to its reliance on selecting mixnodes close to each other. Among different types of intra-circle routing, proportional provides the best results, where it closely follows the FCP of uniform routing till $r = 30$ ms and differ by only 3% for $r = 50$ ms.

For RM we present the results for global mixnet with client in the EU and NA along with the EU and NA mixnet with the respective clients in the same region (refer Fig. 8c). Overall, the FCP decreases with τ for all the cases, but follows a steep drop after $\tau = 0.4$ reaching to a mere 2% from >30% at $\tau = 1$. The global mixnet with EU and NA clients provide the lowest FCP for all values of τ, essentially acting as a lower bound. The EU mixnet closely follows the global mixnet with the NA mixnet being the worse with a difference of about 7% at $\tau = 0.4$. On further analysis we found that there is a small clique of mixnodes in NA that were close to each other and carried a higher weight in biased routing resulting in high FCP with cases of high bias in routing ($\tau < 0.4$).

We observed similar trends for single location and random adversary strategies for all the three LAMP routing approaches.

3) FCP vs. Adversary Budget: In this experiment we quantify the variation in FCP with different rate of adversary corruption (C). We specifically test for $3\%, 10\%, 15\%$, and 20% corrupted nodes in the network for the three routing approaches and under different adversary strategies. Fig. 9 shows the results for the worst-case adversary. We see the trend of increasing FCP with an increase in C, as expected—C increases the corrupted nodes in the network resulting in more paths that can be corrupted end-to-end. For SC (refer Fig. 9a) we observe the uniform intra-circle routing to be on the lower end of FCP, followed by LARMix with $\tau = 0.9$ and then by the remaining approaches. In MC (refer Fig. 9b) the lower bound remains the same for uniform routing, but LARMix with $\tau = 0.3$ gives the worse FCP as it is the most biased intra-circle routing. The RM approach (refer Fig. 9c)) follows a rather interesting trend with NA mixnet providing the least FCP for $C < 10\%$, but has the most FCP for all values above it. The EU mixnet follows a reverse trend with pivot at $C = 10\%$. This can be attributed to the larger number of mixnodes in the EU region compared to the NA. Thus, in NA, a low corruption rate ($< 10\%$) results in low FCP due to insufficient malicious nodes for a corrupted path.

TABLE IV: Comparison of different latency aware approaches on various factors.

Approach	**LARMix**		**SC**		**MC**		**Region (EU)**		**Region (NA)**		**Vanilla mixnet**	
Network size (N)	240	512	240	512	240	512	240	512	240	512	240	512
Propagation Latency (ms)	68	71	52	42	20	7	18	9	46	83	153.4	182.5
Entropy of Transformation Matrix (bits)	3.9	4	4.2	4	3.8	4.8	3.75	3	2.4	3.54	5.9	7
E/L (bits/sec)	57.35	57.22	80.77	91.57	190	691.23	208.3	333.3	52.2	42.7	38.5	38.36
Entropy of Messages (bits)	10.6	11	9.2	10.5	9.5	10.9	10.2	10.3	8.8	9	11.1	11.2
FCP Worst Case	0.15	0.141	0.15	0.149	0.17	0.17	0.22	0.28	0.25	0.3	0.008	0.008
Computation time	$13958t$	$359489t'$	t	t'	$56t$	$50t'$	$8t$	$2t'$	t	t'	t	t'

Overall, across different routing approaches, SC has the lowest FCP (30% for highest bias), followed by MC (32% for highest bias) and then RM (38.5% for highest bias). RM shows a sharp decline towards decreasing bias in comparison to the other two. We observe the same trends for single location and random adversary.

VI. Discussion

A. Overall Comparison

In this section, we compare different LAMP routing methodologies among themselves and also with the original LARMix routing [20]. We quantify the latency minimization, anonymity, adversarial advantage, effect of network size and the computational cost for all the approaches. The results are summarized in Tab. IV.

With respect to propagation latency we can observe that all the LAMP routing approaches outperform LARMix that incurs an average high latency of 68 ms. The regional EU mixnet and MC facilitates the lowest routing delay of 18 ms and 20 ms respectively. In comparison to the vanilla mixnet, the reduction in latency is almost $7.5\times$. Thus, if the focus is only on minimizing propagation latency any of the two approaches would be suitable.

However, the highest routing entropy ($H(T)$) is achieved by SC (4.2 bits) and the highest entropy of individual messages ($H(m)$) is provided by LARMix (10.6 bits). There is a reduction of only half bit in LARMix and less than 2 bits in SC and MC approaches, when compared to the vanilla setting. Note that the $H(m)$ of SC approach reduces despite having a higher $H(T)$ overall, while for LARMix it increases. This is because $H(T)$ is a mapping of first layer mixnodes to the last layer and does not capture the randomness of the first hop selected by the client. In SC and MC, the client selects first hop based on latency (reducing anonymity), however in LARMix its random (contributing to entropy). This is why LARMix shows higher $H(m)$ and SC lower.

If we analyze the E/L ratio, we find that the best tradeoff is achieved by the regional EU mixnet (208.3), closely followed by the MC (190). But we can clearly observe that despite LARMix having a good $H(m)$, it does not fare well providing the lowest E/L of $\approx$ 57. Note that vanilla mixnet setting provides worse tradeoffs despite having high anonymtiy. With respect to adversarial advantage, we find SC and LARMix provides the lowest worst-case FCP of 15%, closely followed by the MC approach at 17%. Here, the RM approach with both the EU and NA mixnet has the highest worst-case FCP at 22% and 25% respectively.

With respect to the computational overhead, we recorded the time it takes to compute the routing policy weights for the same size of the mixnet and normalize it for comparison. The lowest computation time is for the SC and RM mixnet approaches. The MC approach also imposes a $56\times$ computation time overhead. However, the original LARMix approach imposes huge overhead with $13958\times$ more computation time. Further, this time grows exponentially with the size of a layer in mixnet (refer Appendix. VI-C for details). In absolute value terms, it takes LARMix about 6 minutes to calculate the routing policy for a network size of 180 nodes (60×3) and that number increases to about 3 hours for a network size of 7500 (2500×3) (comparable to the size of the current Tor network). In contrast LAMP approaches take at max about 5 minutes for computing the routing policy for a 7500 node network.

Lastly, we compare how LAMP would fare in terms of its capabilities to optimize latency and anonymity when network scales. We considered the RIPE nodes latency dataset [14] consisting of 512 nodes (more than double the size of Nym dataset) for this experiment. We perform the evaluations of all the properties for all the approaches with the RIPE dataset and detail the results in Tab. IV. Across all evaluations the findings and trends related to the different approaches (as discussed previously considering the Nym dataset) remain consistent. However, we observe that all the LAMP approaches perform better in comparison to LARMix in optimizing latency and providing better anonymity tradeoffs. For instance, the MC approach can bring down the average latency to $7ms$ with an E/Lratio of 691.23, which is considerably higher ($\approx 10\times$) than that of LARMix ($71ms$ and 57.22 respectively). Notably, LAMP still remains computationally better with an even greater margin; LARMix takes about $359000\times$ more computation time than SC and about $7200\times$ more than MC.

Overall, all the developed LAMP strategies are better than LARMix and comparable to vanilla mixnet across the studies properties. Moreover, LAMP is lightweight and provides significant computational advantage without posing an unprecedented benefit to the adversary in practical scenarios.

B. Latency-Anonymity Trade-Off in LAMP

One of the key design decision in LAMP is to select a subset of nodes (that are closer to either the client or the previous hop) among all the available ones for building a routing path. While this simplifies the routing and intuitively helps in reducing latency, it would also seem to inadvertently affect anonymity (less nodes to choose from so reduced randomness and hence reduced anonymity). In contrast, LARMix design decision is to bias the route selection towards low latency paths based on a tunable parameter, but considering all the nodes geographically spread across the globe. Again, while biasing the selection of nodes intuitively should reduce latency, the reduction would be limited as geographically far nodes despite

the low probability could still be selected resulting in an overall higher average latency in comparison to LAMP— similarly, the anonymity might be higher on average for LARMix. However, to compare the two approaches on similar grounds one needs to look at their latency-anonymity tradeoffs. The E/L ratio quantifies this information and our results show that LAMP provides superior advantage, essentially establishing that LAMP's design choice is able to provide quantifiably more latency reduction, per units of anonymity loss and is thus overall more effective design.

1) On Guard (Node) Placement Attacks: In this section we discuss the possibility of the guard placement (GP) attack on mixnets. Such attacks have been studied for the popular anonymous communication network Tor [24]. The GP attack refers to the adversaries capability to place its compromised nodes in positions of maximum advantage. In Tor, the first hop (also known as the guard node) is a key hop as the client directly connects to it and generally the assignment of the guard is long-term *i.e.,* client uses the same guard for an elongated period of time. A malicious guard could facilitate end-to-end deanonymization attacks on Tor if the last hop (or the exit node) is also controlled by the adversary. Such attacks are more feasible in biased routing strategies as it allows adversaries to exploit the biasness to place the node closer to the clients location, as there is no mixing or delaying of packets to resist traffic corelation and analysis.

However, such attacks in mixnets are not largely applicable due to various reasons. First, mixnets assume a global passive adversary, rendering traffic analysis ineffective. Second, there is no long-term association for a client with any hop, making any long-term gain impossible. Third, all individual packets in a mixnet follow a different path. Note that the analysis of an adversary being able to gain considerable advantage due to biased routing is already covered in our analysis of the worst-case possibilities of FCP in Sec. V-C.

C. Complexity and Overhead of Routing Approaches

In this section, we aim to compare the complexity of node selection and message routing in LARMix and LAMP.

LARMix: In LARMix, the initial step involves clustering all N nodes using the recommended k-medoids algorithm. This process is composed of three phases:

1) Selection of random cluster centers, which has a computational complexity of $O(k)$, where k represents the number of clusters.
2) Iteratively checking for the closest medoids for all nodes, which incurs $O(kN)$ operations per iteration, where N is the total number of nodes..
3) The iterative process significantly contributes to the total computational complexity, summing up to $O(kN^2)$ due to repeated calculations across multiple iterations.

After clustering in LARMix, the mixnet is structured in a diversified manner, employing a diversification algorithm for assigning mixnodes to various layers of the mixnet. The procedure of this algorithm is as follows:

1) Initially, it selects a node randomly from the center of a random cluster to serve as the starting point.
2) The algorithm then chooses the second node by evaluating all available nodes. It selects the one that maximizes the distance from the first node, ensuring diversification.
3) This process continues with each subsequent node being selected based on its maximum distance from all previously chosen nodes, further enhancing the mixnet's diversified structure.

To efficiently deploy such algorithms, each mixnode is first assigned a vector containing the latencies to other mixnodes. This vector needs to be sorted, implying that in the best case scenario (using merged sorting, considering all the N lists of latency) for all N mixnodes, it requires $O(N^2 \log N)$ complexity. However, this complexity can be more accurately estimated as $O(N^3 \log N)$ because, in the diversification algorithm, at each stage we need to check the sum of latency from chosen mixnodes to find one which maximize this sum. In other words we need to redo the sorting which in worst case we need to do so by considering all the latency, computing all the N^2 cases.

Given the mixnet's structure achieved using the diversification algorithm in LARMix, routing within the mixnet is governed by the probability distribution outlined in (6). This distribution must be computed for all $(l-1)\frac{N}{3}$ mixnodes across the $l-1$ layers, starting from the second layer. It is assumed that the first mixnode is chosen uniformly at random. For each mixnode referenced in equation (6), $\frac{N}{3}$ operations are necessary, resulting in a total computational requirement of $O\left((l-1)\frac{N^2}{9}\right)$.

The final stage of LARMix involves a greedy algorithm aimed at balancing the mixnet layers by ensuring equal message input traffic across all mixnodes. This algorithm operates in two phases:

1) Identification of underloaded, overloaded, and balanced mixnodes.
2) Redistribution of load from overloaded to underloaded mixnodes based on the distribution defined in (6), with a computational complexity up to $\frac{mN^4}{16}$, where m represents the number of iterations, which is at least equal to N. In some cases, this process may not converge, highlighting a limitation in the approach.

These steps illustrate the intricate nature of implementing LARMix, emphasizing the balance between algorithm efficiency and network performance. The described complexities not only highlight the computational burden but also underline the need for robust and optimized algorithms to manage these demands effectively.

LAMP: However, in LAMP, we have three different routing policies, for each of which we can show that the complexity is asymptotically $O(N \log(N))$. Let us first consider the SC scenario. In this case, a client first needs to measure the latency to all the mixnodes in the mixnet, which can cause a complexity of $O(N)$, where N is the total number of nodes in the mixnet. Then, the client needs to set the circle by sorting all the mixnodes based on their latency. Using the best sorting algorithm (such as merged sortting), this operation incurs $O(N \log(N))$ computation.

After sorting, the client will select the first mixnode either through uniform selection, which causes $O(1)$ complexity, or through LARMix and Proportional routing, which cause $O(N)$ complexity since these routing policies require some operations on each mixnode's latency to derive the routing probability. This complexity will apply to choosing the nodes in the subsequent layers. Therefore, overall, the complexity is $O(N\log(N)) + l \cdot O(1)$ or $O(N\log(N)) + O(N)$, where l is the number of layers. Although using LARMix or Proportional routing inside the circle will be slightly more expensive, the asymptotic complexity remains $O(N\log(N))$.

Moreover, using the MC scenario requires first measuring the latency from the client to all the mixnodes in the first layer, which causes $O\left(\frac{N}{l}\right)$ complexity, where N is the total number of nodes and l is the number of layers. Further, to sort this list, it incurs $O\left(\frac{N}{l}\log\frac{N}{l}\right)$ complexity. This amount of complexity is needed to sort the list to ensure we include close mixnodes in the circle. For further layers, we also have the same complexity to sort the mixnodes in each subsequent layer from a preceding node. Therefore, the overall complexity is $l \cdot O\left(\frac{N}{l}\log\frac{N}{l}\right)$ just for making the circles.

However, for the latency measurement beyond the first layer, we assume the latency is available through network probing using the Verloc mechanism [13], which should not add more computational burden. Additionally, using either uniform, LARMix, or Proportional routing leads to $O(1)$ or $O(N)$ complexity. Therefore, asymptotically, the MC scenario also leads to $O(N\log N)$ complexity.

For the RM, the scenario is slightly different. First, to make the regions, N comparisons need to be done, which causes a complexity of $O(N)$. However, if within-region routing is uniform, the overall complexity will remain $O(N)$. On the other hand, using either LARMix or Proportional routing requires sorting and applying low-latency formulas, which slightly worsens the complexity to $O(N\log(N))$.

TABLE V: Complexity of LAMP vs. LARMix (Simplified LARMix is the Imbalance LARMix when we construct the mixnet uniformly at random, rather than using diversification algorithm.)

Routing Policies	**Low-latency**	**Uniform**
Greedy LARMix	$O(N^5\log(N))$	$O(N^5\log(N))$
Imbalance LARMix	$O(N^3\log(N))$	$O(N^3\log(N))$
Simplified LARMix	$O(N^2\log(N))$	$O(N)$
SC	$O(N\log(N))$	$O(N\log(N))$
MC	$O(N\log(N))$	$O(N\log(N))$
RM	$O(N\log(N))$	$O(N)$

Tab. V summarizes the complexity of LARMix considering both its Imbalanced and Balanced versions, and LAMP considering all the routing policies. As you can see, using LARMix always causes more complexity than using LAMP approaches. If one uses the full LARMix method considering Clustering + Diversification + Low-latency routing and balanced routing, it can cause a computation burden of $O(N^5\log N)$. This can further decrease to $O(N^3\log N)$ if one excludes the balanced network consideration, and in the best case, if we disregard both clustering and diversification as well, to $O(N^2\log N)$. Meanwhile, using LAMP usually results in a burden of $O(N\log N)$, which is at least N times faster than LARMix. However, using a RM with uniform routing can further decrease the computational load to $O(N)$, a linear function of N, proving the effectiveness and simplicity of the LAMP approaches.

VII. Conclusion

We introduced LAMP, a set of three routing schemes (SC, MC, and RM) geared towards reducing latency in mixnets to facilitate wider application support. We show that the developed techniques are efficient in reducing latency and outperform the current state-of-the-art LARMix, providing $3\times$ better overall tradeoffs. The developed approaches do not significantly increase adversarial advantage and are extremely lightweight in terms of computation. Overall, this work takes a significant stride forward for supporting low-latency applications over mixnet with practical deployment considerations.

Acknowledgments

We would like to thank the anonymous reviewers for their valuable feedback. This work was supported in part by CyberSecurity Research Flanders with reference number VR20192203.

References

[1] M. Akhoondi, C. Yu, and H. V. Madhyastha, "Lastor: A low-latency as-aware tor client," in *2012 IEEE Symposium on Security and Privacy*. IEEE, 2012, pp. 476–490.

[2] M. AlSabah, K. Bauer, T. Elahi, and I. Goldberg, "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, 2013, pp. 143–163.

[3] R. Annessi and M. Schmiedecker, "Navigator: Finding faster paths to anonymity," in *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE, 2016, pp. 214–226.

[4] A. Barton, M. Wright, J. Ming, and M. Imani, "Towards predicting efficient and anonymous tor circuits," in *27th {USENIX} Security Symposium ({USENIX} Security 18)*, 2018, pp. 429–444.

[5] I. Ben Guirat, D. Gosain, and C. Diaz, "Mixim: Mixnet design decisions and empirical evaluation," in *Proceedings of the 20th Workshop on Workshop on Privacy in the Electronic Society*, 2021, pp. 33–37.

[6] D. L. Chaum, "Untraceable electronic mail, return addresses, and digital pseudonyms," *Communications of the ACM*, vol. 24, no. 2, pp. 84–90, 1981.

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

[8] C. Diaz, H. Halpin, and A. Kiayias, "The nym network," 2021.

[9] C. Diaz, S. Seys, J. Claessens, and B. Preneel, "Towards measuring anonymity," in *Privacy Enhancing Technologies: Second International Workshop, PET 2002 San Francisco, CA, USA, April 14–15, 2002 Revised Papers*. Springer, 2003, pp. 54–68.

[10] R. Dingledine, N. Mathewson, and P. Syverson, "Tor: The second-generation onion router," Naval Research Lab Washington DC, Tech. Rep., 2004.

[11] J. Geddes, M. Schliep, and N. Hopper, "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*, 2016, pp. 165–176.

[12] K. Hogan, S. Servan-Schreiber, Z. Newman, B. Weintraub, C. Nita-Rotaru, and S. Devadas, "Shortor: Improving tor network latency via multi-hop overlay routing," in *2022 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2022, pp. 1933–1952.

[13] K. Kohls and C. Diaz, "Verloc: Verifiable localization in decentralized systems," *arXiv preprint arXiv:2105.11928*, 2021.

[14] R. NCC, "Latency and location data of probes and anchors in ripe atlas project," https://atlas.ripe.net/measurements/, 2015.

[15] A. Panchenko, F. Lanze, and T. Engel, "Improving performance and anonymity in the tor network," in *2012 IEEE 31st International Performance Computing and Communications Conference (IPCCC)*. IEEE, 2012, pp. 1–10.

[16] A. M. Piotrowska, "Studying the anonymity trilemma with a discrete-event mix network simulator," in *Proceedings of the 20th Workshop on Workshop on Privacy in the Electronic Society*, 2021, pp. 39–44.

[17] A. M. Piotrowska, J. Hayes, T. Elahi, S. Meiser, and G. Danezis, "The loopix anonymity system," in *26th {USENIX} Security Symposium ({USENIX} Security 17)*, 2017, pp. 1199–1216.

[18] M. Rahimi, "CLAM: client-aware routing in mix networks," in *Proceedings of the ACM Workshop on Information Hiding and Multimedia Security, IH&MMSec 2024, Baiona, Spain, June 24-26, 2024*, F. Pérez-González, P. C. Alfaro, C. Krätzer, and H. V. Zhao, Eds. ACM, 2024, pp. 199–209. [Online]. Available: https://doi.org/10.1145/3658664.3659631

[19] ——, "Larmix++: Latency-aware routing in mix networks with free routes topology," in *International Conference on Cryptology and Network Security*. Springer, 2024, pp. 187–211.

[20] M. Rahimi, P. K. Sharma, and C. Diaz, "Larmix: Latency-aware routing in mix networks," in *The Network and Distributed System Security Symposium*. Internet Society, 2024.

[21] F. Rochet, R. Wails, A. Johnson, P. Mittal, and O. Pereira, "Claps: Client-location-aware path selection in tor," in *Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security*, 2020, pp. 17–34.

[22] M. Sherr, M. Blaze, and B. T. Loo, "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, 2009, pp. 73–93.

[23] F. Shirazi, M. Simeonovski, M. R. Asghar, M. Backes, and C. Diaz, "A survey on routing in anonymous communication protocols," *ACM Computing Surveys (CSUR)*, vol. 51, no. 3, pp. 1–39, 2018.

[24] G. Wan, A. Johnson, R. Wails, S. Wagh, and P. Mittal, "Guard placement attacks on path selection algorithms for tor," *Proceedings on Privacy Enhancing Technologies*, vol. 2019, no. 4, 2019.

[25] T. Wang, K. Bauer, C. Forero, and I. Goldberg, "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, 2012, pp. 98–113.

APPENDIX

A. Additional Results

1) Expanding on Simulation Results: In this section, we expand on the simulation results provided for RM in Section IV-C1 by extending these to SC and MC scenarios.

In Fig. 10, the simulation results depict the entropy of messages for SC and MC approaches. The first two upper figures illustrate the entropy of messages in simulations for SC and MC. In both SC and MC scenarios, as the radius of the circle increases, the entropy of messages also increases. This is a direct outcome of having more mixnodes within the circle, resulting in more routing options and increased randomness, which in turn leads to higher entropy.

Specifically, simulations were conducted for these approaches considering Uniform, Proportional, and LARMix with $\tau = 0.6$ routing within circles. The simulation results show that the Uniform approach, regardless of the radius of cells, provides the highest entropy, as Uniform routing maximizes uncertainty for adversaries. In contrast, when using Proportional routing, entropy slightly decreases compared to the Uniform approach. LARMix with $\tau = 0.6$ routing shows a further slight decrease in entropy compared to the Proportional case, indicating that LARMix routing tends to bias routes towards low-latency paths. Generally, these results align with analytical findings.

In Fig. 10, the lower figures illustrate the latency results obtained from the simulation. These figures reveal that for both SC and MC scenarios, as the radius of cells increases, the simulation latency l_{e2e} also increases. This is due to having more mixnodes, resulting in more randomness and a higher chance of being routed through high-latency paths. However, an interesting exception occurs when the radius of a SC exceeds 10 ms. In this case, as explained earlier, the presence of mixnodes not close to the clients but near other mixnodes inside the circle leads to a further decrease in latency.

Furthermore, the Uniform approach, while increasing the entropy of messages for both SC and MC cases, results in higher latency compared to the Proportional approach. The latency of the Proportional approach is also slightly higher compared to routing with LARMix $\tau = 0.6$ within circles.

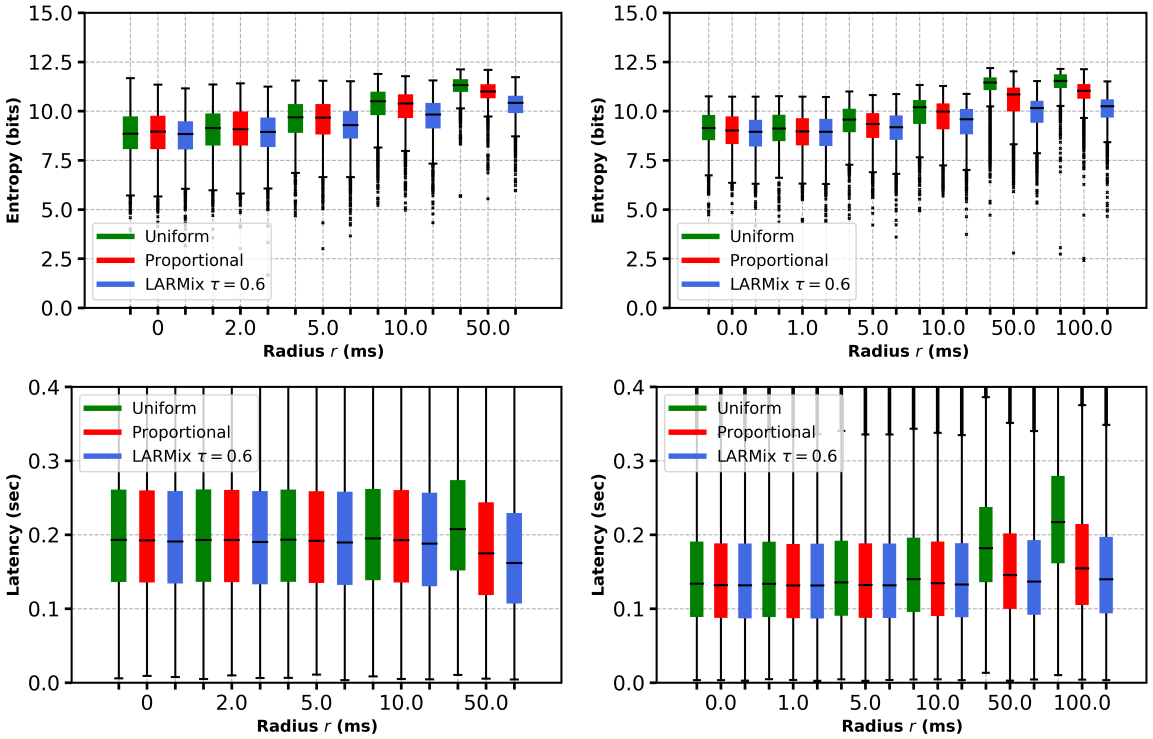
B. Additional Discussion Points

1) End-to-End Delay Constraint Results: In this section we extend the result of end-to-end limits in latency, mainly described for EU region in section IV-C2,to NA region through tables VI and VII.

2) Consideration of mixnet to destination latency for optimization: In this work, we optimize latency from the client up to the last node in the mixnet, but we do not consider optimizing $l_{mix,d}$ due to multiple challenges and drawbacks. For instance, the client (who chooses the entire route) may not know the latency between all possible mixnet exits and the destination, in advance, as it will require adding another set of measurements for all possible exit nodes – destination pair making it impractical for realistic scenarios. Furthermore, choosing paths based on source-destination pairs implies that the path leaks information about the client and destination pair to the adversary (compared to the routing choice being independent of destination), e.g., the first node, knowing the sender and the second node, would get some information about the location of the destination (as the second node is chosen for its proximity to the destination).

C. Comparison with Some Concurrent Works

Concurrent to LAMP, recent works have been published on low-latency routing, which we briefly explain here. One such work, CLAM [18], employs low-latency routing approaches to offer strategic selection of the initial nodes by clients within a mixnet. Specifically, it extends LARMix's functionality by incorporating low-latency routing starting from the client. However, the routing approaches in CLAM are based on computationally expensive strategies, such as linear programming or methods similar to LARMix, which, as we have discussed, are challenging to deploy in practice. In contrast, LAMP



(a) SC

(b) MC

Fig. 10: Entropy and latency from simulations.

TABLE VI: Global mixnet and NA clients: trade-offs between randomness (τ) and mixing delay

Randomness τ	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
Link Delay (l_{mix}) ms	14	14	14	14	15	18	30	58	91	118	141
Mix Delay (μ) ms	61	61	61	61	61	60	56	47	36	27	19
Entropy of Messages (H(m)) bits	10.36	10.37	10.35	10.33	10.41	10.55	10.8	11.15	11.21	10.94	10.41
Entropy of Transformation Matrix (H(T)) bits	0	0	0.2	0.6	1.34	2.27	3.57	4.88	5.51	5.67	5.7

TABLE VII: NA mixnet and clients: trade-offs between randomness (τ) and mixing delay

Randomness τ	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
Link Delay (l_{mix}) ms	24	24	24	24	26	31	42	62	89	121	170
Mix Delay (μ) ms	58	58	58	58	57	56	52	45	36	26	9
Entropy of Messages (H(m)) bits	10.51	10.5	10.52	10.56	10.74	11.0	11.25	11.42	11.42	11.29	10.48
Entropy of Transformation Matrix (H(T)) bits	0	0	0.15	0.5	1.14	1.79	2.45	3.04	3.42	3.61	3.7

enables efficient selection of initial nodes using lightweight algorithms, avoiding the complexity of these methods while maintaining practical usability.

Additionally, LARMix++ [19] attempts to extend LARMix to a Free Routes topology of the mixnet. Unlike the Stratified topology, where nodes are structured in layers, Free Routes topology allows any node to connect to any other node. LARMix++ introduces modifications to LARMix to provide low-latency routing for clients in this topology. However, similar to LARMix, LARMix++ suffers from scalability and deployment challenges. An interesting avenue for future research would be demonstrating that LAMP can be adapted to the Free Routes topology to provide efficient low-latency routing for such scenarios.

D. Artifact Appendix

1) Description & Requirements: We assessed LAMP in two distinct environments: an analytical setting and a simulation setting. In the analytical setting, we modeled the mixnet using matrices and executed the necessary operations. In the simulation setting, we represented the mixnet as a discrete event simulator with the Simpy Python framework, defin-

TABLE VIII: Baseline parameters for experiments.

Parameter	Value
Mix layers (L)	3
Size of network (N)	240
Layer size (W)	60
Number of Regions (N_R)	2
Mix latency (μ)	50 ms
Input traffic rate	20000 msgs per sec
Target messages	200

ing multiple classes to simulate the functionality of network components. These included messages moving through the network, mixnodes processing these messages, and the overall mixnet governing its structure and operation. Various scenarios, such as single circle, multiple circles, and regional mixnets from LAMP, were modeled. These components interact to conduct simulations, perform tasks, and generate results. In total, the analysis scripts and evaluation code amount to around 15K lines of Python.[4]

How to access You can access the artifact of LAMP through the following link on permanent storage with DOI: https://doi.org/10.5281/zenodo.14218376 or also at the Github link: https://github.com/LightAMP/LAMP.

Hardware dependencies The original experiments were performed on a server with 128 GB RAM and 2 TB storage. However, a scaled-down version of the experiments, with fewer iterations, can be run on standard systems with 16 GB RAM and 50 GB of disk space. We emphasize that you should adhere to the specified dependencies to ensure the experiments run within the mentioned time limits.

Software dependencies To run the codes the server needs to have Python 3.8+ (< 3.11) installed.

Benchmarks For both settings, latency and geographical data for mixnodes were required. To ensure a realistic evaluation, we used latency datasets from the deployed NYM mix network.

2) Artifact Installation & Configuration: To evaluate the artifact, one needs to set up a Python environment with Python version 3.8 or higher. The dependencies of the code can be easily installed using the `requirements.txt` file bundled with the code. Once this is achieved, the results can be easily generated by executing the `Main.py` file.

3) Major Claims: In this section, we highlight the main claims of LAMP:

- (C1): ENTROPY TREND Figs. 5a and 5b show that increasing the radius (r) enhances entropy, with the highest entropy around 5 to 6 bits when r is at its maximum and routing is uniform. Analogously, Fig. 5c indicates an increase in entropy when increasing the parameter (τ), reaching close to 6 bits in the global mixnet.
- (C2): LATENCY TREND Fig. 5d demonstrates that increasing r reduces latency across all routing methods except for uniform routing, where latency tends to increase with rising r. However, Figs. 5e and 5h reveal that latency increases as r or τ rise across all routing methods.
- (C3): SIMULATION RESULTS Simulation results in Fig. 6a suggest that increasing τ leads to a general increase in entropy. Specifically, the entropy in the "EU Mixnet + EU Clients" configuration lags behind that of the "Global Mixnet + EU Clients." In Fig. 6b, increasing τ results in an overall rise in latency, with "EU Mixnet + EU Clients" showing slightly lower latency than "Global Mixnet + EU Clients."
- (C4): ADVERSARY ANALYSIS The fraction of corrupted paths (FCP), as shown in Fig. 8, indicates that increasing the radius (r) or τ (in Fig. 8c) reduces the FCP. Figs. 8a, 8b, and 8c show that uniform routing and global mixnet scenarios produce the lowest FCP, respectively.

4) Evaluation: In this section, we align each claim from the previous section with an experiment that verifies it. All results will automatically be stored in the `Results` folder. Further, you can take two different approaches to run the experiments: either you can run all experiments automatically in one go, or you can opt to run them one by one in case you are interested in specific experiments. Following this, we will provide details on both approaches.

Experiment (E) [All Figures] [500 min]: This experiment executes all the code necessary to generate all figures presented in the paper, supporting all the claims made. Additionally, all subfigures of Figures 7 and 8, as well as Figures 5g, 5h, and 5i, will be generated for the sake of completeness. (Refer to the following subsections if you are interested in running the experiments individually.)

[Preparation and Execution] To run this experiment, execute `Main.py` using the following command. After running this, you will need to provide the `Input` argument, which should be set to 0 for this experiment.

```
python3 main.py
```

[Results] The results will automatically be saved in the "Results" folder.

Experiment (E1 & E2) [Figure 5] [200 min]: This experiment provides the basic results of LAMP, verifying the trends in latency and entropy as described in **C1** and **C2**.

[Preparation and Execution] To run this experiment, execute `Main.py` using the following command. After running this, you will need to provide the `Input` argument, which should be set to 12 for this experiment.

```
python3 main.py
```

[Results] The results will automatically be saved in the "Results" folder as Fig. 5a, Fig. 5b, Fig. 5c, Fig. 5d, Fig. 5e, and Fig. 5f.[5]

[4]Note that the current artifact appendix was evaluated by artifact evaluators based on the initial version of the paper.

[5]Additionally, Fig. 5g, Fig. 5h, and Fig. 5i will also be generated.

Experiment (E3) [Figure 6] [60 min]: This experiment examines the simulation results for latency and entropy, confirming the trends described in **C3**.

[Preparation and Execution] To run this experiment, execute `Main.py` using the following command. After running this, you will need to provide the `Input` argument, which should be set to 3 for this experiment.

```
python3 main.py
```

[Results] The results will automatically be saved in the "Results" folder as Fig. 6a and Fig. 6b. [6]

Experiment (E4) [Figure 8 (and 7)] [100 min]: This experiment provides the adversarial results of LAMP, validating the outcomes described in **C4**.

[Preparation and Execution] To run this experiment, execute `Main.py` using the following command. After running this, you will need to provide the `Input` argument, which should be set to 4 for this experiment.

```
python3 main.py
```

[Results] The results will automatically be saved in the "Results" folder as Fig. 8a, Fig. 8b, and Fig. 8c. Additionally, this experiment generates Fig. 7a, Fig. 7b, and Fig. 7c, which do not directly support **C4** but are included for completeness of adversarial analysis.

Experiment (E5) [Figure 9] [100 min]: This experiment examines the adversarial results of LAMP as the adversarial budget increases. Generally, an increasing budget results in greater advantages for adversaries. (Note: This experiment is included for completeness and does not directly support any specific claims made in the paper.)

[Preparation and Execution] To run this experiment, execute `Main.py` using the following command. After running this, you will need to provide the `Input` argument, which should be set to 9 for this experiment.

```
python3 main.py
```

[Results] The results will be saved automatically in the "Results" folder as Fig. 9a, Fig. 9b, and Fig. 9c.[7]

5) Warnings: While running the experiments, you might encounter the following warnings. Please ignore them as they do not impact the execution or results of the process:

```
~/Regional.py:1177: RuntimeWarning: invalid

value encountered in double_scalars

 max_load = (np.sum(LIST_LOAD))

 /(len(LIST_LOAD))
```

```
./sklearn_extra/cluster/_k_medoids.py:275:

UserWarning: Cluster 1 is empty!

self.labels_[self.medoid_indices_[1]]

may not be labeled with its

corresponding cluster (1).

./sklearn_extra/cluster/_k_medoids.py:275:

UserWarning: Cluster 2 is empty!

self.labels_[self.medoid_indices_[2]]

may not be labeled with its

corresponding cluster (2).
```

6) Experiment Selection: Finally, we would like to highlight that, to support the main results of the paper, we opted to include only a subset of experiments rather than all possible experiments. This choice was made to ensure that running the experiments is faster, more straightforward, and less resource-intensive. Additionally, the results of other experiments can often be inferred from the main experiments or derived with relative ease. However, if you require access to additional experiments, please feel free to contact the authors.

7) Notes: Please note that the AEC evaluated the version of the work before undergoing revision. Specifically, experiments on the sensitivity of the Single Circle and Multiple Circles approaches to the α parameter, as well as additional tests on larger datasets, were added during the process of evaluation. We include the additional code for reproducing these results in our final public artifact.

[6] Additionally, Fig. 7a and Fig. 8a will also be created in the same folder.

[7] Note that if you wish to construct Figures 5, 6, 7, 8, or 9, simply set the `Input` parameter to be equal to the respective figure number.