On the Anonymity of Peer-To-Peer Network Anonymity Schemes Used by Cryptocurrencies

Piyush Kumar Sharma
imec-COSIC, KU Leuven
Belgium
pkumar@esat.kuleuven.be

Devashish Gosain
Max Planck Institute for Informatics
Germany
dgosain@mpi-inf.mpg.de

Claudia Diaz
imec-COSIC, KU Leuven
Nym Technologies SA
Belgium
cdiaz@esat.kuleuven.be

ABSTRACT

Cryptocurrency systems can be subject to deanonimization attacks by exploiting the network-level communication on their peer-to-peer network. Adversaries who control a set of colluding node(s) within the peer-to-peer network can observe transactions being exchanged and infer the parties involved. Thus, various network anonymity schemes have been proposed to mitigate this problem, with some solutions providing theoretical anonymity guarantees.

In this work, we model such peer-to-peer network anonymity solutions and evaluate their anonymity guarantees. To do so, we propose a novel framework that uses Bayesian inference to obtain the probability distributions linking transactions to their possible originators. We characterize transaction anonymity with those distributions, using entropy as metric of adversarial uncertainty on the originator's identity. In particular, we model Dandelion, Dandelion++ and Lightning Network. We study different configurations and demonstrate that none of them offers acceptable anonymity to their users. For instance, our analysis reveals that in the widely deployed Lightning Network, with just 5 strategically chosen colluding nodes the adversary can uniquely determine the originator for $\approx 67\%$ of the transactions. In Dandelion, an adversary that controls 15% of the nodes has on average uncertainty among only 4 possible originators. Moreover, we observe that due to the way Dandelion and Dandelion++ are designed, increasing the network size does not correspond to an increase in the anonymity set of potential originators, highlighting the limitations of existing proposals.

1 INTRODUCTION

Cryptocurrencies are digital currencies that are neither issued nor backed by a centralized banking or financial authority. Instead, they rely on the decentralized verification of cryptographic transactions using blockchain technology, allowing everyone to join and contribute to securing the transaction ledger [30]. Decentralized currencies attempt to address concerns of the existing banking system, where centralization implies having entities (banks) with disproportionate power to exclude users, control financial flows and amass a wealth of personal financial information on their customers (habits, lifestyle, spending behaviour, degree of financial desperation) that can be mined for profiling, sold and end up being used against the person. A growing number of blockchain-based cryptocurrencies have been proposed and deployed, with Bitcoin [3, 5], active for almost a decade now, being the most popular currency as well as the seminal system that popularized the concept.

The impressive emergence of blockchain-based cryptocurrency systems (currently with a combined valuation of more than one trillion dollars) has attracted growing interest to various aspects of the underlying technologies. The decentralization and scalability aspects of blockchains have received considerable attention and are by now well understood [20, 22, 40]. On the other hand, understanding the privacy properties of these systems presents additional complexity. Transaction anonymity requires protection both on-chain and in the underlying peer-to-peer network used to transport the transaction. Ideally, if a transaction is considered private it should not be possible for third parties to identify its source or destination, neither by analyzing blockchain data, nor by analyzing network traffic data available to peers. The default process of flooding transactions in the Bitcoin network has however been shown to be prone to network-level deanonymization attacks [1, 2, 27], where the public identifier of a transaction originator (public key) is mapped to its IP address. The subsequently proposed diffusion technique (a more sophisticated version of flooding) has also been shown to be vulnerable to deanonymization by adversaries that control a number of nodes in the Bitcoin network [15, 16].

Peer-to-peer schemes with privacy-enhanced routing features have been proposed to improve transaction anonymity in Bitcoin towards malicious peers. Dandelion [4] and Dandelion++ [14] have been specially designed to provide anonymity when broadcasting transactions through the Bitcoin peer-to-peer layer; whereas the Lightning Network (LN) [23] is a layer-2 payment channel network that addresses both scalability and privacy challenges in Bitcoin. Characterizing and quantifying the anonymity provided by such peer-to-peer routing schemes, such that they can be systematically evaluated and compared, has so far remained an open challenge.

To address this challenge, this paper proposes a Bayesian framework to model and analyze the anonymity provided by peer-to-peer networking schemes towards corrupted peers. Given a network, the scheme's routing features and constraints, and an adversarial observation, the framework outputs the probability distributions linking transactions to their possible originators. Following prior work on network anonymity metrics [11, 37], we quantify the uncertainty of the adversary in determining a transaction's originator with the entropy of the distribution that relates the transaction to its potential originators.

We remark that our Bayesian approach to modeling and evaluating anonymity in peer-to-peer schemes is generic and can be adapted to any anonymous peer-to-peer routing scheme, even if it is unrelated to cryptocurrencies (*e.g.,* intended instead for private browsing or messaging [33]), since we rely *exclusively* on network-level traffic data and not on any transaction-specific data. Thus, the analysis is identical if originators use the peer-to-peer routing scheme to anonymously broadcast (or to send to selected recipients) a 280-character text message, instead of a blockchain transaction.

In terms of accommodating the main existing peer-to-peer concepts for anonymous routing, we demonstrate the generality of our approach by applying it to three schemes: Dandelion, Dandelion++ and LN, noting that they present a major difference in the way they operate—Dandelion and Dandelion++ implement *hop-by-hop* probabilistic routing (that ends in *broadcast*), whereas in LN transactions are *source-routed* (all the way to the *intended recipient*).

While our approach to computing originator probabilities is generic, we have additionally developed scheme-specific heuristics for the two different types of routing schemes under evaluation. With these heuristics, an adversary can perform a more nuanced analysis for identifying the source of the transaction. For example, since LN is source routed and transactions are relayed along the shortest path to the transaction counterpart, the immediate predecessor and successor of a malicious node in the transaction path provide valuable information to better estimate the originator probabilities. In §4.2 we describe in detail the rationale behind this heuristic and how it is used in our anonymity evaluation. Similarly, we develop heuristics for Dandelion and Dandelion++ by utilizing coordination among adversarial nodes to reduce anonymity sets (ref. §4.1).

We developed our own simulator (that also incorporates the aforementioned heuristics) to evaluate the anonymity of these schemes. The results of our analysis raise concerns on the level of protection offered by these systems. We observed that in the LN, the median entropy is *zero* when a few high centrality (or high degree) nodes are adversarial, meaning that the adversary can confidently identify the originator of more than half the transactions they route as intermediary. Moreover, these few adversary nodes ($\approx 0.06\%$ of total nodes) can intercept about 60% of all transactions making the conclusion even more worrisome. Interestingly, we observed a comparably high impact even when selecting medium centrality nodes as adversaries. In this case, the median entropy for the transactions observed by the adversary was again zero. However, these nodes captured a relatively smaller fraction ($12\% - 32\%$) of all transactions. We conducted additional experiments to further assess the anonymity offered by LN in different scenarios (*e.g.,* entropy estimations by a budget constrained adversary *etc.*). In all such experiments, we again observed median entropy of *zero* (ref. §5.2 for details).

Furthermore, our results also indicate that Dandelion and Dandelion++ do not offer high anonymity either. For example, if an adversary controls 15% of the nodes in the Bitcoin peer-to-peer network, Dandelion offers a median entropy of about two bits (*i.e.,* equivalent to uncertainty among *four* possible originators per transaction) while capturing 63.8% of transactions on average. With the same fraction of adversaries, for Dandelion++ the median entropy is about five bits (*i.e.,* equivalent to 32 possible originators per transaction) while capturing 59% of transactions on average, thus demonstrating better anonymity in comparison to Dandelion even if still limited. Finally, we increase network size and analyze whether anonymity increases accordingly, as it would be expected given that network scaling enables larger anonymity sets [12]. We find however that the entropy offered by both Dandelion and Dandelion++ does not increase with network size, indicating that scaling the Bitcoin network will not result in better anonymity.

To summarize the main contributions of this work:

- We propose a generic Bayesian framework to evaluate network-level anonymity in peer-to-peer networks, including both hop-by-hop and source routing schemes.
- We modelled and evaluated in our framework three concrete schemes *i.e.,* Dandelion, Dandelion++ and Lightning Network, that have been proposed and deployed to support transaction anonymity in Bitcoin.
- We present a detailed evaluation of the aforementioned schemes and observe that they generally offer extremely poor anonymity to transactions.
- We discuss and recommend changes that can lead to improving anonymity in these schemes.

2 BACKGROUND AND RELATED WORK

2.1 Bitcoin Network

Bitcoin consists of a P2P network of nodes that communicate via TCP [30]. When a Bitcoin node receives (or generates) a transaction, it further broadcasts the transaction to its neighbours within the network. The neighbours then broadcast the transaction to their neighbours and so on. This process is iterative and after some time the transaction information reaches all the Bitcoin nodes.

At the application layer, a Bitcoin node is identified by its public key, whereas at the network layer, it is identified by its IP address. In order to provide anonymity for the originator of the transaction, the originator node's IP address and public key should remain unlinkable. This is important because all the Bitcoin transactions generated by an originator are stored on a public blockchain in plain text along with the their public key. Notably, if the originator's public key can be linked to its IP address, the transaction would be completely deanonymized.

Prior work has demonstrated various deanoymization attacks on the Bitcoin network [1, 2, 27]. These attacks typically introduce a "supernode" (adversary node pretending to be a normal Bitcoin node) that connects to all Bitcoin nodes and observes the timing of transactions broadcast by other nodes. In such a situation the originator node is likely the first to be seen broadcasting the transaction. Since transactions include their originator's public key, the supernodes are able to associate transaction public keys to their originator IP addresses with an accuracy of up to 30%. To mitigate such attacks, Bitcoin introduced an alternative approach known as *diffusion*, where each node waits for a randomized amount of time (chosen independently from an exponential distribution) before broadcasting transactions to its neighbors on the Bitcoin network. It was however later shown that this diffusion does not provide much anonymity either [15, 16].

More recently, Dandelion [4] and Dandelion++ [14] have been proposed to offer theoretical anonymity guarantees to cryptocurrency networks. These schemes are not yet deployed on the Bitcoin network but are however under consideration [7]. In addition, the Lightning Network is a payment channel network that aims to address scalability as well as privacy concerns of Bitcoin, including network anonymity. Lightning Network is a functional and deployed system (with about 10K active nodes) that is currently integrated with Bitcoin as a layer-2. In this paper we model and evaluate the anonymity properties offered by Dandelion, Dandelion++ and Lightning Network. We now briefly introduce these approaches.

2.2 Dandelion

Dandelion [4] was designed to enhance network anonymity for Bitcoin by making it harder to link a transaction to the IP of the node that originated it. When a node generates a transaction in Dandelion, it does not directly diffuse it to the Bitcoin network, but instead forwards it to just one of its Bitcoin network neighbours. The neighbour node then tosses a biased coin and decides to either forward the transaction to one of its own neighbours, or to diffuse it. If it forwards the transaction to a neighbour, the process is repeated, until a node eventually diffuses the transaction. Transactions are thus forwarded a random number of hops (following a geometric distribution) before being finally diffused in the Bitcoin network. The adversary may still identify the diffuser node [15], but since this is a different node than the originator, the identity of the originator remains obfuscated.

Dandelion thus propagates transactions in two phases: (i) stem (or *anonymity*) phase, and (ii) a fluff (or *diffusion*) phase. For routing transactions in the stem phase, a privacy-subgraph graph is constructed from Bitcoin's P2P graph by selecting a subset of edges. This privacy-subgraph should ideally be a Hamiltonian circuit (a line graph) consisting of all the Bitcoin nodes. The fluff phase of Dandelion is simply the diffusion process of the Bitcoin network.

In Dandelion, the node that generates a transaction never directly diffuses it directly to the Bitcoin network, and instead always forwards it to its successor in the line graph. Probabilistic forwarding is applied from the first intermediary on: when a node receives a transaction from their predecessor, it forwards it to its successor with forwarding probability p_f (where $0 < p_f < 1$) and diffuses it with probability $1 - p_f$. Thus, each transaction propagates over the line graph for a random number of hops before entering the fluff phase, where it is diffused in the Bitcoin network. The number of hops follows a geometric distribution with average $\frac{1}{1-p_f}$.

2.3 Dandelion++

Dandelion++ [14] builds and improves upon Dandelion. It relaxes some of the idealistic assumptions made in Dandelion that may not likely hold in practice, in particular that: (1) each node generates exactly one transaction, (2) all nodes strictly adhere to the protocol, (3) all nodes run the Dandelion protocol. The authors of Dandelion++ further demonstrated that violations of these assumptions can lead to serious deanonymization attacks in Dandelion.

Similar to Dandelion, Dandelion++ operates in two phases—*stem* and *fluff*. Unlike Dandelion however, it does not use a line as the privacy-subgraph of the stem phase; instead, it constructs a quasi 4-regular graph where each node should have both indegree and outdegree of two (*i.e.,* two predecessors and two successors). Thus, when a node receives a transaction from any of its predecessors, it forwards it to any one of its two successors with probability $p_f/2$, and diffuses the transaction into the Bitcoin network with probability $1 - p_f$ (*i.e.,* transaction enters into fluff phase).

2.4 Lightning Network (LN)

LN [32] is a payment channel network (PCN) that was primarily proposed to address the scalability concerns of Bitcoin. In this network, two peers use the Bitcoin blockchain to open and close payment channels between them. Using these channels, peers can make payments between themselves without having to use the Bitcoin blockchain. LN not only supports direct transactions between peers that share an open channel, but indirect as well. Users who have not established a direct payment channel can still transact through other LN participants that act as intermediaries in the PCN. These intermediaries are incentivized to route payments by a fee that they can charge for the payments they forward.

LN can be easily represented as a graph consisting of (1) nodes (users), (2) edges (payment channels between users) and (3) edge weights (fee charged for routing transaction via that channel *etc.*). An up-to-date snapshot of the full LN graph is maintained at each node. When some node (Alice) wants to make a payment to another node (Bob) to which it is not directly connected, Alice first computes the shortest path to Bob (using Dijkstra) in the network that charges the lowest fee (while also minimizing other factors such as the wait time in case of payment failure). Once Alice has computed the path, she encrypts the transaction multiple times using the Sphinx packet format [8]. Sphinx conceals the position of the node in the path and thus Alice's successor cannot tell that it is the first node after the originator, and neither can Bob's predecessor determine that Bob is the recipient based on Sphinx packet headers.

3 THREAT MODEL AND ANONYMITY METRIC

There are two main threat models of interest in network anonymity: *global passive adversaries* and adversaries that control a subset of *corrupted nodes*. The three schemes studied in this paper offer no anonymity protection against global passive adversaries. Such adversaries can trivially identify transaction originators using timing: nodes forward transactions shortly after receiving them, and thus whenever a node sends a transaction without having recently received one, the node is an originator; if on the other hand a node sends a transaction shortly after receiving one, then it is an intermediary. Protecting against such adversaries requires some notion of mixing [6] and the introduction of per-node added latency [9].

We thus focus on adversaries that control a subset of nodes, whether by setting up servers, hiring botnets or compromising existing nodes in the network. The adversarial nodes follow the protocols normally (the attacks are passive) but record information that they analyze with the aim to deanonymize the transactions of benign nodes (*i.e.,* identify originator nodes for each transaction).

We consider that the adversary does not have informative priors on the activity of different nodes, and thus use uniform priors over all benign nodes (*i.e.,* we assume that *a priori* all nodes are equally likely to generate a transaction). Note that a non-uniform prior informed by node activity characteristics would further reduce anonymity and facilitate transaction deanonymization.

Starting from the uninformed prior, knowledge of the network graph and of protocol parameters, the adversary records observations from all of its nodes during operations (transactions being forwarded). In particular, in LN the adversary takes note of its immediate predecessor (that forwarded the transaction) as well as its immediate successor (to which the transaction is forwarded), while in Dandelion and Dandelion++ only the predecessor is relevant.

Combining priors, known constraints and observations in a Bayesian framework, the adversary obtains for each transaction an *a posteriori* probability distribution over all possible originators. The entropy of this distribution expresses the uncertainty about the identity of the originator. An entropy of zero means that the transaction originator can be fully and certainly identified, while an entropy of b bits implies that the effective anonymity set of the transaction is equivalent to 2^b possible originators.

Rationale for using entropy: Entropy metrics [11, 37] have been used extensively to quantify network anonymity against traffic analysis, as they allow for an intuitive interpretation of how 'hidden' the true originator is among the other participants in the anonymity set. These metrics take into account not just whether a binary guess by the adversary is correct, but instead express whether the originator is part of a small or big set of suspects, and how salient it appears with respect to the other suspects. The Dandelion and Dandelion++ proposals [4] model anonymity as a classification problem and evaluate it using precision and recall metrics, considering a mapping (perfect match) between transactions and originators. Based on the obtained classification accuracy, they claim that the schemes provide system-wide anonymity guarantees to transactions. In practice however, we find that both Dandelion and Dandelion++ offer zero anonymity to some transactions, which puts in question the nature of any system-wide anonymity guarantees claimed by the schemes. Moreover, the mapping functions are based on some assumptions which may not be reflective of reality or of the best adversarial strategy to follow. For instance the *first-spy-estimator* mapping function simply assumes that the predecessor *is* the originator of the transaction. On the contrary, entropy metrics consider all nodes as possible originators, with their probability of being the true originator based on the available information. This is informative of the adversarial confidence in each deanonymization (entropy zero means that the adversary is completely certain of who is the transaction originator), as well as of the number of suspects that could be associated to a transaction. Prior work analyzing anonymity in LN [21] has also been limited to considering just the adversary's predecessor, rather than the full list of candidates with their distributions. Considering the entropy of the distribution over all possible originators allows to evaluate the size and scaling of anonymity sets beyond binary successful/failed identification. Finally, we note that our method builds simply on observations captured by adversarial nodes, who may only see a subset of all network transactions, rather than require knowledge of *all* transactions in the network to compute possible perfect matchings with *all* the nodes, assuming each node sends exactly *one transaction*, as is the case in Dandelion and Dandelion++.

4 APPROACH

In this section we describe our modeling approach. The considered schemes can be broadly divided into two categories—hop-by-hop (Dandelion and Dandelion++) and source routed (Lightning Network). We explain the common elements of our model first and then the specifics for each type of routing.

Overall, our approach involves calculating the Bayesian (*a posteriori*) probabilities of the possible transaction originators, for any transaction observed by an adversary node. The entropy of this probability distribution provides a measure of transaction anonymity. We begin by introducing the basic notations and definitions that are common to both categories, followed by their detailed anonymity analysis.

- N is the total number of nodes in the privacy-subgraph.
- C is the total number of adversary nodes in the privacy subgraph. Thus, $N - C$ is the total benign (or honest nodes) in the same privacy-subgraph.
- B_i is an event that a benign node i generated a transaction.
- A_j is an event that an adversary node j received a transaction (as intermediary).
- $P(B_i)$ is the probability that a benign node i generated a transaction.
- $P(A_j)$ is the probability that an adversary node j receives a transaction originating from any honest node. *i.e.,* the sum of probabilities of each benign node generating a transaction and forwarding it to the adversary node j.
- $P(A_j|B_i)$ is the conditional probability of an adversary node j receiving a transaction given that honest node i generated it.
- $P(B_i|A_j)$ is the conditional probability of an honest node i being the originator of the transaction given that the adversary node j received it.

Our aim is to find the probability distribution of possible originators for the transactions received by adversary nodes. These probabilities can be calculated (using Bayes' theorem) as:

$$P(B_i|A_j) = \frac{P(B_i) * P(A_j|B_i)}{P(A_j)} \; \forall i, j \tag{1}$$

$P(A_j)$ can be calculated multiplying the probabilities of each node i generating a transaction (*i.e.,* $P(B_i)$) and that transaction reaching the adversary node j (*i.e.,* $P(A_j|B_i)$). This needs to be summed up for all the benign $N - C$ nodes in the network. Thus, $P(A_j)$ is computed as:

$$P(A_j) = \sum_{i=1}^{N-C} P(B_i) * P(A_j|B_i)$$

We consider that *a priori* all benign nodes are equally likely to be the originators and thus $P(B_i) = \frac{1}{N-C}$ for all i benign nodes. Thus Eq. (1) can be reduced to:

$$P(B_i|A_j) = \frac{P(A_j|B_i)}{\sum_{i=1}^{N-C} P(A_j|B_i)} \; \forall j \tag{2}$$

Finally, for each adversary node j, we compute the anonymity of the observed transactions as the Shannon entropy of the *a posteriori* distribution derived above.

$$H = -\sum_{i=0}^{N-C} P(B_i|A_j) * \log_2(P(B_i|A_j)) \; \forall j \tag{3}$$

4.1 Hop-by-hop Routing

We now explain our modelling strategy for Dandelion and Dandelion++ as both forward the transaction using hop-by-hop probabilistic routing.

4.1.1 *Dandelion.*

As described earlier, Dandelion operates in two phases *i.e.,* the stem phase and the fluff phase. In Dandelion anonymity is provided *only* in the stem phase (and not in the fluff phase), and the challenge is thus to identify the originator of a transaction in the stem phase. Hence we model and compute the anonymity properties of Dandelion's stem phase.

In the stem phase all transactions are forwarded over a fixed line graph, also known as the privacy-subgraph, that determines the possible routes followed by transactions in this phase. We assume that this privacy-subgraph is known to the adversary[1] and model the originator probabilities for a line graph. To do so, we first construct a line graph, and then randomly select a few nodes to be adversarial. As a consequence, various partitions are created within line graph, with varying sets of benign nodes in between each pair of adversary nodes. For example, in Fig. 1, there are two partitions—one with benign nodes 1 to n between adversary nodes A_1 and A_2 and other with the remaining benign nodes ($n+1$ to m) between A_2 and A_1. Thus, whenever an adversary node receives a transaction, it knows from which partition the transaction has come, and thus considers only the benign nodes in said partition. For instance, if A_2 receives a transaction and A_1 has not seen this transaction, then A_2 is sure that the originator is one of the nodes 1 to n. This effectively reduces the set of potential originators for any given transaction.

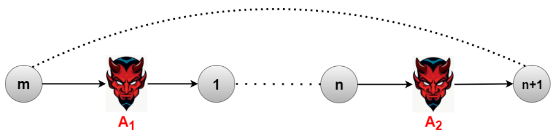


Figure 1: Dandelion privacy-subgraph: There exist n benign nodes between adversary nodes A_1 and A_2; m benign nodes between A_2 and A_1.

After learning the partition N_P that contains the source of the transaction, the adversary needs to compute the conditional probabilities $P(B_i|A_j)$ for individual benign nodes in $B_i \in N_P$. Note that in this case, $P(A_j|B_i) = 0$ for benign nodes $B_i \notin N_P$ (Eq. (2)). For nodes $B_i \in N_P$, $P(A_j|B_i)$ is computed as:

$$P(A_j|B_i) = p_f^{h_{ij}-1}$$

where h_{ij} is the number of hops between a benign node B_i and the adversary node A_j. $P(A_j|B_i)$ represents the probability of the adversary node A_j receiving a transaction given that node B_i generated it. In this case, p_f and h_{ij} both have a role in determining the probability of the transaction reaching the adversary. The farther the benign node is from the adversary (in hops), the smaller the probability of the transaction reaching the adversary. Moreover, since the originator always forwards the transaction to its successor, the adversary receives transactions originated by its predecessor with probability 1.

Our model incorporates an arbitrary number of adversarial nodes. If there are three adversary nodes (A_1, A_2 and A_3) present in the line graph, one would have three partitions (with benign nodes between A_1–A_2, A_2–A_3 and A_3–A_1). In general, n adversary nodes yield n partitions of the line graph and our analysis is performed accordingly for each partition.

[1]We discuss in § 6.1 how the adversary can learn the privacy graph.

4.1.2 *Dandelion++.*

Similar to Dandelion, Dandelion++ also functions in two phases (stem and fluff) and thus many assumptions remain the same. However, there are some differences as well; the most significant one with respect to anonymity is the change in structure of the privacy-subgraph in the stem phase. Dandelion++ constructs an approximate four regular graph as a privacy-subgraph *i.e.,* each node ideally should have indegree and outdegree as two. In practice, when nodes select two of their neighbors as immediate successors in the privacy-subgraph, it can happen that fewer or more than two nodes select same node as successor. Thus, for each node in the privacy-subgraph, outdegree is guaranteed to be two, but indegree may not always be two. For evaluating anonymity in Dandelion++, we construct an approximate 4-regular graph and then randomly select a fraction of these nodes to be adversary nodes (ref. Fig. 2).

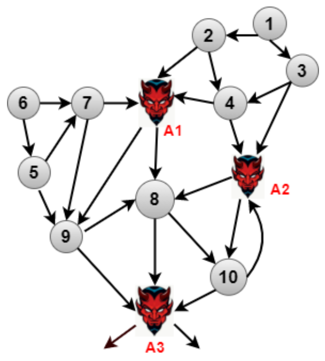


Figure 2: Dandelion++ privacy-subgraph: A_1, A_2 and A_3 represent adversary nodes; rest are benign (honest) nodes.

Unlike with Dandelion, there are no obvious partitions in Dandelion++. This is because in Dandelion, the privacy-subgraph is a perfect line graph (ref. Fig. 1) and thus between any two nodes there exist only one path. In Dandelion++ however, the privacy-subgraph is an approximate four regular graph, thus between any two nodes there normally are multiple paths. An adversary is nevertheless able to reduce the set of possible originators with the following strategy.

Heuristic to reduce anonymity set: Since all adversary nodes coordinate among themselves, they can easily identify duplicate transactions (by simply comparing the transaction plaintext). Adversaries only consider the first instance when they received a transaction, ignoring subsequent observations. This simple criteria greatly benefits the adversary in deanonymizing transactions. For instance in Fig. 2, if adversary node A_3 receives a transaction it can surely conclude that benign nodes 1–4 cannot be the originators. This is because if any one of them would have been the source, the transaction would be first intercepted by either adversary nodes A_2 or A_1, and subsequently be ignored by A_3. Thus only the remaining benign nodes (5–10) in the privacy-subgraph are possible originators.

Having stated the assumptions, we now describe the approach to model the originator probabilities. To that end, we can simply use Eq. (2), computing $P(A_j|B_i)$ as:

$$P(A_j|B_i) = \sum_{k=1}^{T_P} \frac{1}{2} * \left(\frac{1}{2} * p_f\right)^{h_{ij}-1} \tag{4}$$

where T_P is the total number of possible paths between benign node i and adversary node j and h_{ij} is the number of hops between i and j for any valid path k. The rationale for computing $P(A_j|B_i)$ with the expression in Eq. (4) can be explained as follows. In Dandelion++, there can be multiple possible paths for a transaction to reach an adversary node, as the approximate four-regular privacy-subgraph is not a straight line (as was the case with Dandelion). Thus, we need to consider all possible paths (represented as T_P) between a benign node i and an adversary node j to calculate the probability of the transaction reaching the adversary node via any one of these individual paths. We then sum individual probabilities over all valid paths to obtain $P(A_j|B_i)$.

As in Dandelion, in Dandelion++ the node generating the transaction always forwards it to the next hop (i.e., the originator never diffuses the transaction). Since each node in the privacy-subgraph has an outdegree two, any one of the two successors is chosen with equal probability of one half. At each subsequent hop in the path the transaction is forwarded with probability p_f (diffused with probability $1 - p_f$), but once again any one of the two successors is selected with equal probability. To account for this, for each of the $h_{ij} - 1$ intermediate steps of a path we multiply by a factor $p_f/2$ of choosing a next node.

Our strategy to reduce the number of potential originators is implicitly incorporated in the computation of $P(A_j|B_i)$. For example, in Fig. 2, $P(A_{A3}|B_1)$, $P(A_{A3}|B_2)$, $P(A_{A3}|B_3)$ and $P(A_{A3}|B_4)$ are *zero* because there is no honest path between benign nodes (1–4) and adversarial node A_3 (without any other adversary node capturing the transaction first). Thus, A_3 can never be the first adversarial node to receive a transaction originated by 1–4. Thus, even if in Eq. (2) all the $N - C$ (*i.e.,* ten honest) nodes are in principle considered as possible originators, in practice we end up with only the six possible nodes (5–10) whenever A_3 is the first adversarial node receiving a transaction.

4.2 Source Routed

Lightning Network (LN) is a functional and scalable payment channel network with close to ten thousand active nodes. This network employs source routing for selecting paths in the network, meaning that the transaction originator decides on the *entire* routing path to reach the node (recipient) with whom it wishes to transact.

To evaluate the anonymity offered by LN, we again use our generic approach, which calculates transaction originator probabilities using Bayes theorem. However, when compared to Dandelion and Dandelion++, modelling of LN has notable differences that need to be accounted for. In hop-by-hop routing probabilistic routing decisions are made by each node forwarding the transaction, whereas in source routing the path is determined by the source and dependent on the destination and the path fees. Route selection prioritizes paths with lower fees along with considering other path features such as (*cltv expiry values*). We now describe how we model LN to calculate $P(B_i|A_j)$, describing originator probabilities for an intercepted transaction, which we then use to characterize LN's transaction anonymity.

Modelling approach: In LN, whenever a node i_1 wishes to transfer some amount to a node i_2, the source node i_1 computes a "best" path along which a payment can be transferred to i_2. The path that has the optimal cumulative weight (for the payment transfer) is selected as best path[2]. This weight primarily comprises of the fees charged by individual nodes (at each hop).

From an adversary's perspective, once it is part of a transaction path as intermediary, it tries to establish the originator of the transaction. To perform such an analysis, the adversary first computes the best paths for each source-destination pair and then calculates the probability for each originator given that a transaction is routed through an adversarial node. To compute the shortest paths it employs Dijkstra's algorithm and the originator probabilities (*i.e.,* $P(B_i|A_j)$) are computed using equation 2.

$$P(B_i|A_j) = \frac{P(A_j|B_i)}{\sum_{i=1}^{N-C} P(A_j|B_i)} \; \forall j$$

It further calculates $P(A_j|B_i)$ as:

$$P(A_j|B_i) = \frac{SP_{ij}}{SP_i}$$

where SP_{ij} is the total number of shortest paths that originate from benign node i and pass through the adversary node j; and SP_i is the total number of shortest paths that originate from i.

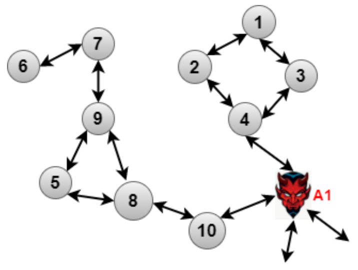


Figure 3: LN Heuristic I: Based on from what predecessor (4 or 10) transaction has been received, potential source nodes can be divided into disjoint sets: 1–4 and 6–10.

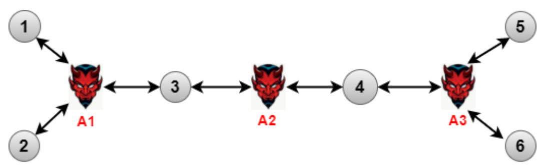
In the aforementioned analysis we considered all benign nodes to be the possible originator of the given transaction. But to further reduce the set of originators, we employ the following strategy.

Reducing the anonymity set (Heuristic I): We take into account *additional information in the form of the predecessor and the successor* of the transaction routed through the adversary node.

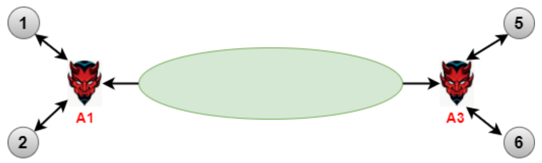
When calculating $P(A_j|B_i)$, for any given j, we further segregate the probability by the successor *and* predecessor node pair. The new probabilities would then be calculated for each subpath defined as $p - j - s$ where p is the predecessor, j is the adversary and s is the successor node. The new probabilities for a subpath $p - j - s$ can then be calculated as:

$$P(B_i|A_{p-j-s}) = \frac{P(A_{p-j-s}|B_i)}{\sum_{i=1}^{N-C} P(A_{p-j-s}|B_i)} \; \forall p, j, s \tag{5}$$

[2]The complete LN topology (nodes, edges and weights) is known to each LN node.



(a) LN path with three adversary nodes A_1, A_2 and A_3



(b) We ignore the all intermediate nodes (including A_2) and consider A_1, A_3 as one adversary node between benign nodes 1, 2, 5, 6.

Figure 4: LN Heuristic II: Subpath computation when multiple adversary nodes are on the same path.

And the expression $P(A_{p-j-s}|B_i)$, can then be computed as follows:

$$P(A_{p-j-s}|B_i) = \frac{SP_{i(p-j-s)}}{SP_i}$$

where $SP_{i(p-j-s)}$ is the total number of paths that originate from benign node i and pass through the nodes $p - j - s$ with j being the adversary node and p and s being the immediate predecessor and successor nodes of j respectively.

We now illustrate with an example that how the inclusion of just a predecessor in computing originator probabilities benefits the adversary in reducing the potential originators for any received transaction. The same reasoning can be applied for the inclusion of a successor as well.

Consider a sample LN topology shown in Fig. 3. With our initial analysis, adversary node A_1 would consider all benign nodes (1–10) as the possible originators. However, if we take into account the predecessor node from which the transaction has been received, the potential originators are divided into two distinct sets: benign nodes 1–4 (corresponding to predecessor 4) and benign nodes 5–10 (corresponding to predecessor 10). Thus, by incorporating predecessor and successor information of a received transaction, an adversary can reduce the set of potential originators.

It must be noted that the inclusion of predecessor and successor for computing originator probabilities is possible in LN but not in Dandelion or Dandelion++. This is because source to destination paths are determined in LN as paths in the graph between two nodes that have the lowest aggregate transaction fees. This is unlike hop-by-hop routing schemes where there is no destination (any node can diffuse) and the path is not even known or predictable to the source as the route is probabilistically chosen hop by hop.

***Reducing the anonymity set (Heuristic II)*:** We employ an additional heuristic to collectively exploit *information gained by multiple adversaries*. We leverage the fact that multiple adversary nodes can recognize that they are part of the same path for a particular transaction in the LN. This is possible because in the LN, every transaction has a unique identifier that can be seen by all the nodes on the said path. Even if such identifier was not available, adversarial nodes may be able to establish they are on the same path by correlating transaction timing and amount.

Thus, to incorporate analysis of paths that contain multiple adversary nodes, we examine the set of shortest paths to find those that have multiple adversaries as intermediaries. We then identify the first (closest to the originator) and last (closest to the destination) adversary nodes on paths that contain multiple adversaries. All the nodes in between these adversary nodes (including other adversary nodes) are ignored. Next, we consider the predecessors of the first and successors of the last adversary node and calculate the combined subpaths. For example in Fig. 4, the combined subpaths for adversary nodes A_1 and A_3 will be 1-(A_1-A_3)-5, 1-(A_1-A_3)-6, 2-(A_1-A_3)-5 and 2-(A_1-A_3)-6.

These extra subpaths help us to further reduce potential originators for some transactions as already explained with the help of Fig. 3. By combining multiple adversaries, we however obtain additional benefits, potentially reducing the set of candidate originators even further. For example in Fig. 4, nodes 3 and 4 will not be considered as originators as they could not have selected a best path that would involve both nodes A_1 and A_3. However, if A_3 and A_1 would have individually performed an analysis, they would have considered node 3 and 4 as originators. Thus, the combination of A_1 and A_3 further narrows down the possible originators compared to adversary nodes that would not share their observations.

Entropy computations for best-k paths: Arguably, in LN a transaction may not always follow the best path because of various network constraints—transaction amount is more than the payment capacity of the path, some node(s) along the path may not be online *etc.* In such a scenario, evaluations solely based on best path estimation may not be accurate.

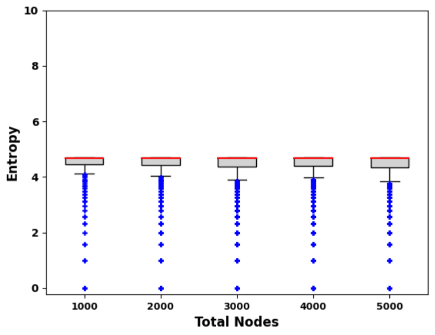
We address this scenario by computing the k-best paths, between each source–destination pair in the LN, rather than simply selecting best path. We assume that the originator of a transaction selects uniformly at random among these best-k paths. Hence, our analysis for calculating the originator probabilities remains the same as that of the single best-path scenario, except now considering the best-k paths between every source-destination pair.

5 ANALYSIS AND RESULTS

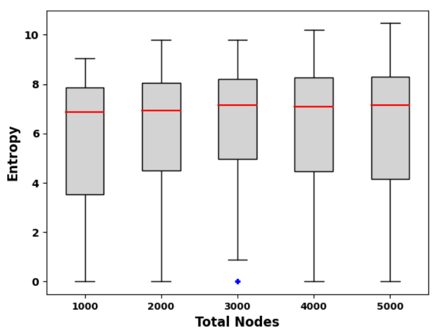
We developed a simulator to evaluate anonymity for the three considered schemes. At a high level, our simulator generates a network, selects adversarial nodes and implements the developed approaches (presented in the previous section). The simulator then uses the Bayesian framework to compute the entropy of the distribution over possible originators. The graphs in this section show the range of entropy values obtained as boxplots, which indicate the range, median, quartiles, and outliers of the obtained entropy values for the set of samples.

5.1 Hop-by-hop Routing

As already mentioned, both Dandelion and Dandelion++ route transactions over a privacy-subgraph, constructed either as a line graph or as an approximate 4-regular graph, depending on the scheme. The three key variables in such routing schemes are: forwarding probability (p_f); total number of nodes (N), and number of adversary/colluding nodes (C). We conduct three sets of experiments to isolate the anonymity impact of each variable:

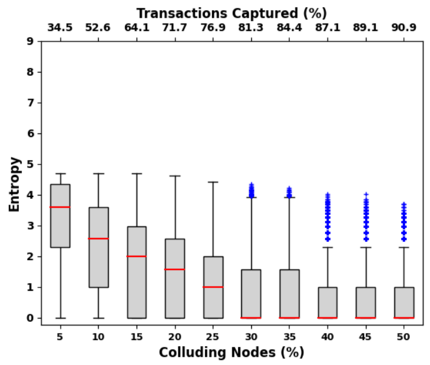


(a) Dandelion

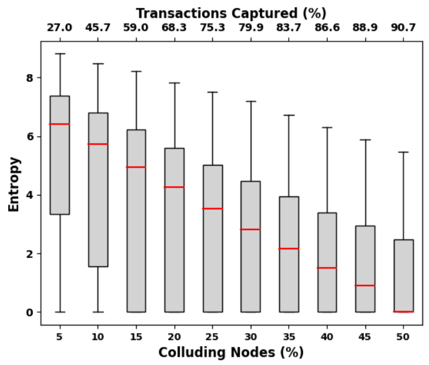


(b) Dandelion++

Figure 5: Entropy vs Total Nodes: With increasing the total number of nodes and keeping the adversary nodes as fixed (1%), the entropy does not increase.



(a) Dandelion



(b) Dandelion++

Figure 6: Entropy vs Colluding nodes: With increasing the fraction of colluding nodes entropy decreases.

(1) Keeping C and N fixed, we varied p_f.
(2) Keeping C and p_f fixed, we varied N.
(3) Keeping N and p_f fixed, we increased C.

1) Impact of forwarding probability (p_f) on anonymity: Previous studies [14] used relatively smaller network topologies (*e.g.,* 100 nodes network) to study the theoretical properties offered by these systems. We take the evaluation a step further and simulate larger networks. We constructed privacy-subgraphs with $N = 1000$ total nodes considering $C = 10$ nodes are corrupted (*i.e.,* 1% of N, which we re-randomized for each experiment). We vary p_f from 0.1 to 0.9 and for each value of p_f we measured the anonymity for 1000 samples. We observe that increasing p_f increases anonymity (ref. Fig. 13 in Appendix A.2 for details), which is to be expected as a higher p_f increases the transaction's path length, which makes it harder for the adversary to determine the source of the target. However, we remark that even with very high $p_f = 0.9$ and just 1% adversary nodes, the median anonymity value is 5 bits for Dandelion and 7 bits for Dandelion++. This means that the effective size of the anonymity set of originators is, respectively, in the order of 32 and 128 originators, out of 990 potential originators.

2) Impact of total number of nodes (N) on anonymity: In our second set of experiments we fix $p_f = 0.9$ (highest anonymity) and instead increase N. For each value of N we simulate and evaluate 1000 samples, each taken with a random placement of adversaries in the graph. Interestingly, as shown in Fig. 5, we observe that increasing network growth (considering a constant fraction of compromise) has no positive effect on the size of the anonymity set, which remains roughly constant. Even when $N = 5000$, the median (and maximum) entropy value is below 5 bits for Dandelion *i.e.,* an effective set of 32 originators. This is because benign nodes that are far from adversary nodes in the privacy-subgraph do not contribute much to the entropy. For example, in Dandelion, assuming a benign node i is 35 hops away from some adversary node j, the probability of a transaction generated by i reaching j, would be ${p_f}^{34}$, which is very small. Thus, this node would have an insignificant contribution in the final computation of entropy because it is highly unlikely that node i could be the originator of any transaction observed by adversary j. In our analysis, we observed that nodes that are more than 30 hops away from the adversary node have a negligible contribution. Thus, even if more nodes join the Bitcoin network, these routing schemes do not leverage network growth to provide larger anonymity sets to transactions.

3) Impact of adversary nodes (C) on anonymity: In the last set of experiments we set $N = 1000$ and $p_f = 0.9$ and vary C from 5% to 50% of N. For each value of C we select 1000 samples (each with randomly placed adversary nodes). Our results paint a grim picture—*e.g.,* in Dandelion (ref. Fig. 6a) with 15% adversary nodes, the median entropy is only two bits *i.e.,* effective anonymity set of

4 potential originators (ideally it should have been 850). Moreover, an adversary with 15% of nodes intercepts 64% of all transactions. By comparison Dandelion++ performs better. Fig. 6b shows that with 15% adversary nodes, the median effective anonymity set is 32 (5 bits), for 59% of transactions that are intercepted. Note that for more than 25% of transactions (third quartile), the provided anonymity is zero, as is in Dandelion, meaning that a quarter of transaction originators are fully identified.

5.2 Source Routing

To measure the anonymity offered by LN, we required a dataset with its topology, since a graph cannot be algorithmically generated to be representative of LN. Thus, we downloaded LN's different real-world topology snapshots from [24]. To give a sense of the size, LN's December 2018 snapshot consists of 1202 nodes and 6196 edges. We compute best paths between every pair of LN nodes in the graph and, using the approach presented in §4.2, evaluate anonymity in different settings.

1) Impact of node degree on anonymity: A node with many neighbors may observe a large number of transactions that are relayed through it. Controlling such nodes is thus ideal for the adversary. Low-budget adversaries may however not have the resources to operate or compromise such nodes, which require the setup of a large amount of payment channels, each requiring collateral. We considered adversaries with varying budgets (reflected as node degrees) to evaluate anonymity in different adversarial configurations. In particular, we partition nodes according to their degree (number of connections to other nodes), bucketing them into four bins of degrees $1-5$, $6-10$, $11-50$, and > 50. For the first three bins we randomly select 1% of nodes as adversary nodes for each sample. In the last bin we deterministically select the top-5 nodes as adversary nodes. Since the last bin represents the case for extremely high degree nodes (> 500) we select fewer nodes (*i.e.,* $\approx 0.5\%$ of N).

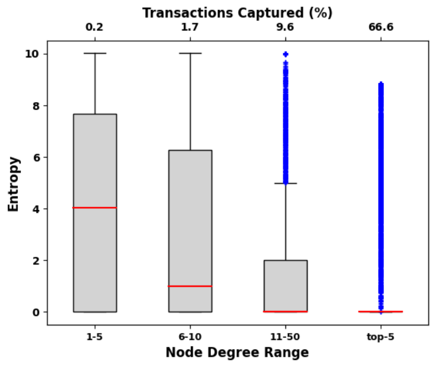


Figure 7: Entropy vs Node Degree: In each sample, 1% nodes were randomly selected as adversary nodes with varying node degree intervals.

Our results in Fig. 7 show that the adversary can fully identify the originator of at least a quarter of transactions (zero entropy for the third quartile), even when just controlling 1% of low-degree LN nodes with just 1–5 connections. The median entropy is 4 bits, meaning effective anonymity sets of 16 possible originators, out of more than two thousand LN nodes. If the adversary controls 1% of nodes with degrees in the range 6–10, the median entropy is down to 1 bit, *i.e.,* an effective anonymity set of two possible originators. With higher degree adversarial nodes (range 11–50), median entropy drops to zero (originator can be uniquely determined for more than half the intercepted transactions), and if the top-5 LN are corrupted, the majority of transactions (except for some lucky outliers) can be uniquely linked to their originator. The degree of the adversary nodes has a strong impact on the fraction of intercepted transactions. Controlling 1% of lowest degree nodes only allows the adversary to observe 0.2% of transactions, while this percentage grows to 9.6% when the adversary nodes have degrees 11–50. The top-5 nodes have the best vantage position and are able to intercept two thirds of all transactions. We discuss in Sec. 6.2 the implications of some nodes having such a disproportionate influence.

2) Impact of node centrality on anonymity: Next, we classify nodes according to the frequency with which they appear in LN paths. A node that frequently appears in network paths can intercept a large fraction of transactions. *Betweenness centrality* captures this property. For a given node, it is determined by the number of shortest paths between any two nodes in the network that pass through the node under consideration. We compute centrality for each node in the LN graph and observe that an overwhelmingly large number of nodes has centrality zero (or close to zero) while a small number of nodes have a high centrality value. More specifically we found 766 nodes with zero centrality, 333 nodes with centrality between zero and one and 103 nodes with centrality values greater than one. We focus on nodes with centrality values of at least one, arrange them in increasing order of their centrality values and divide them into four equal sized bins (of ≈ 26 nodes each). Thus, 'low' contains the first 26 nodes ($1-26$) when arranged in ascending order of centrality, 'low-middle' consists of $26-52$, 'middle-high' contains $53-78$, and 'high' contains the remaining. Again, from each bin, we randomly selected adversary nodes (up to 1% of the network) and computed the entropy (for 128 samples). Notably, we select only the top-5 nodes in the 'high' centrality bin.

Our results in Fig. 8 show that, as expected, adversary nodes with higher centrality capture more transactions, and for those transactions they are able to better identify the originator (lower entropy) than lower-centrality nodes. We note that even for nodes with modest centrality the median entropy is zero, meaning that they are able to fully identify the originator of half the transactions they intermediate, while high centrality nodes can deanonymize most of the transactions they route.

3) Measuring anonymity considering best-k paths: As discussed earlier, it is possible that a node may not always select the cheapest LN path for routing a transaction, instead selecting a more expensive one. Given that this introduces less determinism in LN routing, it should have a positive effect on anonymity for transactions. To evaluate the impact of this effect, we compare scenarios where the top-5 high centrality nodes are adversarial in the two cases: with deterministic path selection and with uniform selection among the top 5-best paths (computed using Dijkstra). In the results shown in Fig. 9 we observe that most intercepted transactions are still fully deanonymized, barring a few outliers.

4) Budget-constrained attack: We have shown that an adversary can deanonymize a large fraction of transactions by controlling

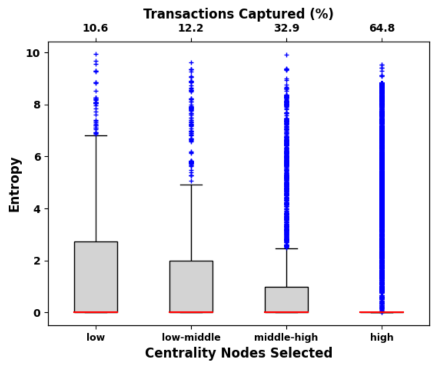


Figure 8: Entropy vs Centrality: In each sample, 1% of total nodes were selected as adversary nodes with varying node central

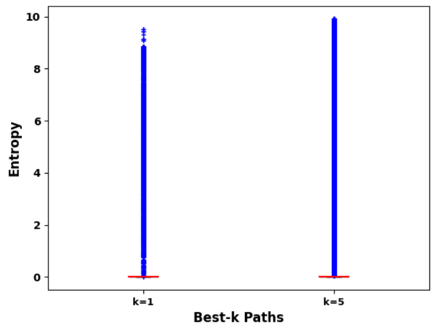


Figure 9: Entropy under best-k paths: Best-1 and best-5 paths were selected; for both of them top-5 centrality nodes were selected as adversary nodes.

nodes that have high degree and high centrality. However, this may involve significant monetary costs, because to have a high degree, the node has to invest money (collateral) opening channels with many other nodes. Given an adversary with a constrained budget for opening channels, we examine attack strategies for adversarial node selection. The budget can be distributed among a number of nodes, and we examine whether it is more effective to operate more adversary nodes with fewer channels (and low centrality) or fewer adversary nodes with more channels (and high centrality).

For simplicity we assume that the adversary can open a fixed number of channels and that the cost of opening any channel is the same. Considering the LN topology snapshot, we use different strategies to select which nodes to consider adversarial. In the first strategy we simply select the top centrality node to be adversarial. We take note of its degree and use this number as budget for the other cases. In the other cases we randomly select nodes with centrality values greater than one, bucketing nodes in three categories from lower to higher centrality. We keep the total budget of channels (aggregate degree of all adversary nodes) the same in all cases as in the top centrality case, meaning that the low centrality case has as many as 25 (low degree) adversary nodes, while the high centrality case has just seven (high degree) nodes. The results are shown in Fig. 10. Given the same budget, the top centrality node is the most successful in intercepting transactions (a whooping

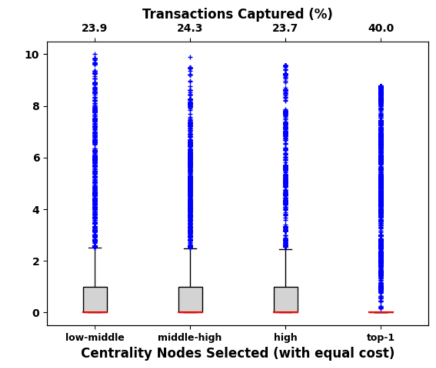


Figure 10: Entropy (with fixed budget): Different adversary combinations based on centrality values were selected but all under the same budget (or cost).

40%) and deanonymizing them all except for some lucky outliers. In the other cases the adversary intercepts almost a quarter of transactions and fully deanonymizes mode than half (median is zero). Moreover, in another quarter of the cases (first quartile) the effective anonymity sets are lower than 1 bit, *i.e.,* 2 possible originators. We repeated these experiments with different selections of adversary nodes (keeping roughly the same budget) and consistently observed similar findings.

5) Measuring anonymity for the latest LN snapshot: The LN has grown significantly in the last years. Compared to the December 2018 snapshot used in the previous experiments, by May 2021 the LN includes more than 9300 nodes and $\approx 52K$ edges, with the largest connected component containing more than 8100 nodes and $\approx 48K$ edges. Here we examine if the scaling of LN has had a positive impact (as one might expect) on the anonymity offered to transactions.

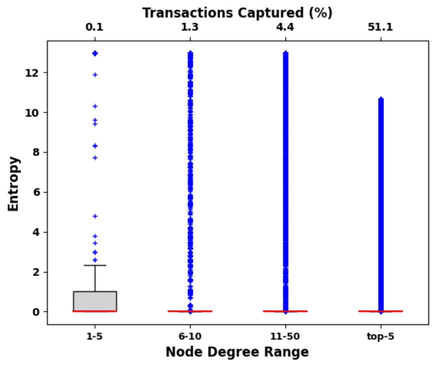


Figure 11: Entropy vs Node Degree (for LN's latest snapshot): In each sample, 1% adversary nodes are selected from nodes of a certain degree.

We partition the nodes into bins according to their node degree and apply the same analysis as before to the new dataset. For the first three bins we randomly select nodes from the corresponding bin to be adversary nodes until 1% of network nodes are adversarial. For the last bin we again select the top-5 nodes as adversary nodes, which in this case amount (combined) to just $\approx 0.06\%$ of all nodes.

Our results in Fig. 11 show that the adversary can deanonymize intercepted transactions with high confidence irrespective of the adversary node degree (the median entropy is zero in all four cases). Compromising high degree nodes however allows intercepting (and thus deanonymizing) a much larger fraction of transactions. The top-5 nodes alone can intercept 50% of transactions in the network and deanonymize the vast majority of them.

Compared to the 2018 LN topology considered in previous experiments, the 2021 topology has around 7 times more nodes and 8 times more edges. This network growth has not however led to larger anonymity sets as one might expect, but on the contrary, the larger and more recent topology presents much lower anonymity than the older version shown in Fig. 7. While the percentage of intercepted transactions decreases slightly with network size, the majority of intercepted transactions have zero anonymity in the larger network, even when the adversary controls nodes with modest degrees. This was not the case in the smaller topology, where a percentage of transactions had several bits of anonymity in some adversarial configurations.

5.3 Implementation Details

We developed our own simulator to evaluate the three considered schemes under different settings. The simulator is written in Python and about 2K lines of code. For generating network topologies, we used `networkx` Python library. The topologies for Dandelion and Dandelion++ are constructed with their specified constraints *i.e.,* line graph for Dandelion and quasi 4-regular graph for Dandelion++. For LN, we used the real world snapshots that were obtained from [24]. We note that LN has multiple client implementations (LND, c-lightning and eclair) with variations in how they construct paths [39]. For our analysis we used LND's implementation, as majority of the nodes ($> 90\%$) use LND [29, 35]. The routing algorithms of the other implementations can be easily incorporated by just modifying the weight function, before calculating best paths using Dijkstra. We already evaluate routing similar to that of eclair's implementation by calculating the 5 best paths between source-destination pairs (eclair computes 3 best paths and randomly chooses one of the to route the transaction).

Our analysis requires heavy computations on graph topologies *e.g.,* in the LN topology consisting of more than 8K nodes and more than 48K edges, we needed to compute Dijkstra shortest paths between all pairs of nodes. Thus, we ensured that our compute engine is a highly paralleled program that spawns different threads. To achieve parallelization, we used `multiprocessing` Python library. We performed our analysis on two servers with Intel Xeon processors, each with 100 GB RAM and 20 physical cores. We plan to open source our simulator upon paper publication.

6 DISCUSSION

6.1 Graph Learning Attacks

In order to successfully deanonymize the source of a transaction, the adversary must know the underlying network topology: the complete network graph in case of source routing *i.e.,* LN; and the privacy-subgraph, *i.e.,* line graph for Dandelion and 4-regular graph for Dandelion++, in hop-by-hop routing schemes. Obtaining LN topology is trivial, as every node maintains a complete up-to-date topology. A locally available topology is essential for computing paths and routing payments through the network. Obtaining a privacy-subgraph is relatively harder, but still possible. In Dandelion, $x\%$ adversary nodes can easily infer the positions of about $2x\%$ benign nodes, as they know their immediate successor and predecessor in the line graph. Similarly, in Dandelion++ the information of approximately $4x\%$ nodes is directly available. The adversary can employ different techniques to infer the rest of the privacy subgraph. One such approach consists of sending transactions to honest nodes whose successors in the privacy subgraph are yet to be identified, then observe who difusses those transactions [16]. The transactions sent to honest nodes to learn the graph may be generated by the adversary, but not necessarily. The adversary may simply relay any transactions generated by other nodes that are sent to its node.

When many transactions are routed via an honest node, the distribution of nodes that diffuse these transactions allows inferring the successors of that target node in the privacy subgraph. This is because the successors' frequency of diffusion is directly dependent on the number of hops in the privacy subgraph between the target and the successor, being highest for immediate successors and decaying geometrically with the number of hops. The adversary can combine this information with knowledge of the bitcoin graph[3] to significantly narrow down the possible graph neighbours and strengthen its inference, taking into account that only the 8 immediate neighbors of a node in the bitcoin graph are potential successors in the privacy subgraph (since privacy subgraph is derived from the bitcoin graph).

We simulated this approach and observed that typically 2 of the 8 bitcoin graph neighbours of a node diffuse the most transactions, and can thus be easily identified as the successors of the honest node in the privacy subgraph. We were able to learn the privacy subgraph of Dandelion++ with more than 90% accuracy when we sent 50 transactions per honest node. In Appendix A.3, we explain our approach in a step-by-step manner and also present some additional heuristics that further optimize the overall accuracy, lowering the number of transactions required to learn the graph.

6.2 Lightning Network Graph Structure

We observed that the vast majority of nodes in the LN (for different snapshots) have (betweenness) centrality zero or less than one, while relatively few nodes have overwhelmingly large values of centrality. This indicates that there is "routing centralization" within LN *i.e.,* a few nodes capture a large fraction of network paths, a fact already noted in prior studies [34, 39]. The analysis by Rohrer et al. [34] of an actual LN topology snapshot of 2019 highlighted that the node degree distribution of LN follows power-law, suggesting a *scale-free* network structure. They also demonstrated that LN can also be classified as *small-world* network. *i.e.,* nodes tend to cluster and have a high density of edges in their cluster. Our results indicate that the growth of LN (> 9K nodes presently) has not diminished "routing centralization". We thus turn to asking whether a change in graph structure would improve the anonymity

[3]Although the bitcoin graph is not publicly available. There are various researches [10, 13, 17, 19, 28] that demonstrate how the bitcoin graph can be reconstructed with high accuracy.

offered by LN: if instead of a free scale topology the LN graph was more balanced, with roughly similar degrees for all nodes, would the network provide better anonymity?

Impact of change in graph structure on LN: We create random graphs with the constraint that the average degree (k) of nodes should remain constant (*e.g.,* k=5). We assign weights to the edges following a distribution similar to the actual LN (the average fee associated with a channel is 1000 millisatoshi) and compute shortest paths between all pair of nodes. We again select the top-5 centrality nodes as adversary nodes and compute anonymity for the transactions intercepted by the adversary. We observe that entropy in these balanced graphs is somewhat higher (median 1–2 bits of entropy with higher standard deviation) in comparison to the actual LN topology (median zero with low standard deviation). The overall entropy values are however low and still allow the adversary to significantly narrow down the identity of transaction originators. On the other hand, despite the overall entropy being low, in random networks the adversary intercepts a smaller fraction of transactions compared to the LN real topology snapshots (ref. Fig. 12). This is because in random graphs all the nodes have similar node degree and their centrality values are balanced (almost none has zero centrality). This is unlike the actual LN snapshots, where ≈ 64% nodes have zero centrality—which allows those high-centrality nodes to become intermediaries for a disproportionate fraction of transactions.

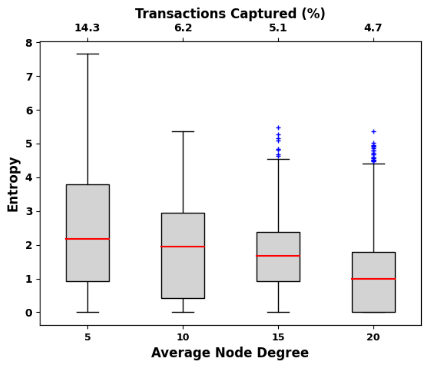


Figure 12: Entropy for random k-degree graph: For 1000 node topology, we increased the average node degree, kept the edge weights roughly same and study its impact on entropy (with top 1% centrality nodes as adversary).

6.3 Improving Dandelion's Anonymity

Our analysis demonstrates that changing the privacy subgraph from line to 4-regular results in better overall entropy. Thus, we investigate if increasing node degree in the privacy subgraph leads to further anonymity improvements. Since the privacy subgraph is derived from the bitcoin graph which is itself 16-regular, this is the maximum degree that any node in the privacy subgraph can have. Thus, we constructed a 16-regular graph of 1000 nodes and computed the entropy as described in Sec. 4.2. We observed that with 10% adversary nodes the median entropy value was 8.5 bit (equivalent to 386 possible senders) in 16-regular privacy graph, which is 4 folds higher than the value of ≈ 4 bit (16 possible senders) observed in 4-regular privacy-subgraphs[4]. Thus, to achieve better anonymity, hop-by-hop schemes like Dandelion should use the bitcoin graph *as-is* for the privacy subgraph. Notably, in this case, the adversary already knows the privacy subgraph without the help of any additional mechanisms (as described in Sec. 6.1), but is still less able to effectively deanonymize the nodes.

7 RELATED WORK

LN is a deployed network and thus there are multiple researches that study its privacy and anonymity aspects. In [36], Romiti *et al.* demonstrate a cross-layer attack where the LN nodes were mapped to their bitcoin addresses. In [38] Tikhomirov *et al.* attempt to perform a primitive analysis of the payment paths that may be vulnerable to deanonymization attacks. They do so by selecting a set of influential adversaries (high degree nodes) and find the potential payment paths intercepted by the adversary nodes. But adversary nodes in the payment paths cannot ensure deanonymization as LN's design ensures that a node on the payment path cannot determine whether the previous hop is the actual originator or just the forwarder. Thus, a more nuanced analysis (as performed in our work) is required to analyze how and to what extent the adversary can deanonymize a transaction for which it acts as intermediary.

Moreover, LN includes additional mechanisms such as 2-of-2 multi-signature transactions (for channel construction), Hashed Time Lock Contracts (for payment management) *etc.* [32]. The exploitation of such mechanisms to deanonymize transactions has been studied in previous works [18, 25, 26, 31, 35]. Our analysis makes abstraction of transaction data and instead relies exclusively on traffic data that can be passively collected. This makes our contribution generalizable for evaluating the anonymity provided by anonymous P2P routing schemes, whether they are used for routing Bitcoin transactions or simply messages.

8 CONCLUSION

Schemes for anonymous P2P routing have been proposed to strengthen network privacy in Bitcoin and other cryptocurrency networks. Notably, hop-by-hop routing solutions (Dandelion and Dandelion++) have been proposed to anonymize the broadcast of Bitcoin transactions, while Lightning Network uses a source routed scheme to provide payment channels as layer-2 of bitcoin. In this work we propose a generic framework to measure the anonymity provided by such P2P schemes. This framework relies on Bayesian inference for computing the probability of potential originators for any given transaction, and computes the overall uncertainty on the actual originator by measuring the entropy of the distribution. Our evaluation of the said schemes reveals some startling facts. For instance, our analysis of a real Lightning Network snapshot reveals that, by colluding with a few (*i.e.,* only 5) influential nodes, an adversary can fully deanonymize more than 50% of total transactions in the network. Similarly, Dandelion and Dandelion++ provide a low median entropy of 2 and 5 bit, respectively, when 15% of the nodes are adversarial.

Overall, our study highlights a pressing need for better network anonymity schemes in cryptocurrency networks as the current

[4]To bound the computations we considered only 5-hop paths from source to the adversary nodes

solutions provide very poor anonymity to users. Our simulation framework can be used not only to evaluate existing proposals, but also for designing and evaluating new solutions with a principled understanding of how much anonymity they will provide in concrete configurations.

REFERENCES

[1] Elli Androulaki, Ghassan O Karame, Marc Roeschlin, Tobias Scherer, and Srdjan Capkun. 2013. Evaluating user privacy in bitcoin. In *International conference on financial cryptography and data security*. Springer, 34–51.

[2] Alex Biryukov, Dmitry Khovratovich, and Ivan Pustogarov. 2014. Deanonymisation of clients in Bitcoin P2P network. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*. 15–29.

[3] Rainer Böhme, Nicolas Christin, Benjamin Edelman, and Tyler Moore. 2015. Bitcoin: Economics, technology, and governance. *Journal of economic Perspectives* 29, 2 (2015), 213–38.

[4] Shaileshh Bojja Venkatakrishnan, Giulia Fanti, and Pramod Viswanath. 2017. Dandelion: Redesigning the bitcoin network for anonymity. *Proceedings of the ACM on Measurement and Analysis of Computing Systems* 1, 1 (2017), 1–34.

[5] BTC [n. d.]. Bitcoin. https://bitcoin.org/en/.

[6] David Chaum. 1981. Untraceable Electronic Mail, Return Addresses, and Digital Pseudonyms. *Commun. ACM* 24, 2 (1981), 84–88. https://doi.org/10.1145/358549.358563

[7] danbip [n. d.]. Dandelion proposal submitted to Bitcoin for integration with Bitcoin core. https://github.com/bitcoin/bips/blob/master/bip-0156.mediawiki.

[8] George Danezis and Ian Goldberg. 2009. Sphinx: A Compact and Provably Secure Mix Format. In *2009 30th IEEE Symposium on Security and Privacy*. 269–282. http://research.microsoft.com/en-us/um/people/gdane/papers/sphinx-eprint.pdf

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

[10] Sergi Delgado-Segura, Surya Bakshi, Cristina Pérez-Solà, James Litton, Andrew Pachulski, Andrew Miller, and Bobby Bhattacharjee. 2019. Txprobe: Discovering bitcoin's network topology using orphan transactions. In *International Conference on Financial Cryptography and Data Security*. Springer, 550–566.

[11] Claudia Diaz, Stefaan Seys, Joris Claessens, and Bart Preneel. 2002. Towards measuring anonymity. In *International Workshop on Privacy Enhancing Technologies*. Springer, 54–68.

[12] Roger Dingledine and Nick Mathewson. 2006. Anonymity Loves Company: Usability and the Network Effect.. In *WEIS*.

[13] Jean-Philippe Eisenbarth, Thibault Cholez, and Olivier Perrin. 2021. A Comprehensive Study of the Bitcoin P2P Network. In *2021 3rd Conference on Blockchain Research & Applications for Innovative Networks and Services (BRAINS)*. IEEE, 105–112.

[14] Giulia Fanti, Shaileshh Bojja Venkatakrishnan, Surya Bakshi, Bradley Denby, Shruti Bhargava, Andrew Miller, and Pramod Viswanath. 2018. Dandelion++ Lightweight Cryptocurrency Networking with Formal Anonymity Guarantees. *Proceedings of the ACM on Measurement and Analysis of Computing Systems* 2, 2 (2018), 1–35.

[15] Giulia Fanti and Pramod Viswanath. 2017. Anonymity properties of the Bitcoin P2P network. *arXiv preprint arXiv:1703.08761* (2017).

[16] Giulia Fanti and Pramod Viswanath. 2017. Deanonymization in the bitcoin P2P network. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*. 1364–1373.

[17] Federico Franzoni, Xavier Salleras, and Vanesa Daza. 2021. AToM: Active topology monitoring for the bitcoin peer-to-peer network. *Peer-to-Peer Networking and Applications* (2021), 1–18.

[18] Matthew Green and Ian Miers. 2017. Bolt: Anonymous payment channels for decentralized currencies. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. 473–489.

[19] Matthias Grundmann, Till Neudecker, and Hannes Hartenstein. 2018. Exploiting transaction accumulation and double spends for topology inference in bitcoin. In *International Conference on Financial Cryptography and Data Security*. Springer, 113–126.

[20] Abdelatif Hafid, Abdelhakim Senhaji Hafid, and Mustapha Samih. 2020. Scaling blockchains: A comprehensive survey. *IEEE Access* 8 (2020), 125244–125262.

[21] George Kappos, Haaroon Yousaf, Ania M. Piotrowska, Sanket Kanjalkar, Sergi Delgado-Segura, Andrew Miller, and Sarah Meiklejohn. 2021. An Empirical Analysis of Privacy in the Lightning Network, Vol. LNCS. Springer, 20.

[22] Soohyeong Kim, Yongseok Kwon, and Sunghyun Cho. 2018. A survey of scalability solutions on blockchain. In *2018 International Conference on Information and Communication Technology Convergence (ICTC)*. IEEE, 1204–1207.

[23] LN [n. d.]. Lightning Network: Scalable, Instant Bitcoin/Blockchain Transactions. https://lightning.network/.

[24] LNtopo [n. d.]. Lightning Network Latest Updated Information For Channels, Nodes, Channel Policies and Topology. https://ln.fiatjaf.com/.

[25] Giulio Malavolta, Pedro Moreno-Sanchez, Aniket Kate, Matteo Maffei, and Srivatsan Ravi. 2017. Concurrency and privacy with payment-channel networks. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. 455–471.

[26] Giulio Malavolta, Pedro Moreno-Sanchez, Clara Schneidewind, Aniket Kate, and Matteo Maffei. 2018. Anonymous multi-hop locks for blockchain scalability and interoperability. *Cryptology ePrint Archive* (2018).

[27] Sarah Meiklejohn, Marjori Pomarole, Grant Jordan, Kirill Levchenko, Damon McCoy, Geoffrey M Voelker, and Stefan Savage. 2013. A fistful of bitcoins: characterizing payments among men with no names. In *Proceedings of the 2013 conference on Internet measurement conference*. 127–140.

[28] Andrew Miller, James Litton, Andrew Pachulski, Neal Gupta, Dave Levin, Neil Spring, and Bobby Bhattacharjee. 2015. Discovering bitcoin's public topology and influential nodes. (2015).

[29] Ayelet Mizrahi and Aviv Zohar. 2021. Congestion attacks in payment channel networks. In *International Conference on Financial Cryptography and Data Security*. Springer, 170–188.

[30] Satoshi Nakamoto. 2008. Bitcoin: A peer-to-peer electronic cash system. *Decentralized Business Review* (2008), 21260.

[31] Utz Nisslmueller, Klaus-Tycho Foerster, Stefan Schmid, and Christian Decker. 2020. Toward Active and Passive Confidentiality Attacks On Cryptocurrency Off-Chain Networks. (2020).

[32] Joseph Poon and Thaddeus Dryja. 2016. The bitcoin lightning network: Scalable off-chain instant payments.

[33] Michael K Reiter and Aviel D Rubin. 1998. Crowds: Anonymity for web transactions. *ACM transactions on information and system security (TISSEC)* 1, 1 (1998), 66–92.

[34] Elias Rohrer, Julian Malliaris, and Florian Tschorsch. 2019. Discharged payment channels: Quantifying the lightning network's resilience to topology-based attacks. In *2019 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*. IEEE, 347–356.

[35] Elias Rohrer and Florian Tschorsch. 2020. Counting down thunder: Timing attacks on privacy in payment channel networks. In *Proceedings of the 2nd ACM Conference on Advances in Financial Technologies*. 214–227.

[36] Matteo Romiti, Friedhelm Victor, Pedro Moreno-Sanchez, Peter Sebastian Nordholt, Bernhard Haslhofer, and Matteo Maffei. 2020. Cross-layer deanonymization methods in the lightning protocol. *arXiv preprint arXiv:2007.00764* (2020).

[37] Andrei Serjantov and George Danezis. 2002. Towards an Information Theoretic Metric for Anonymity. In *Proceedings of the 2nd International Conference on Privacy Enhancing Technologies (PET'02)*. Springer-Verlag, Berlin, Heidelberg, 41–53.

[38] Sergei Tikhomirov, Pedro Moreno-Sanchez, and Matteo Maffei. 2020. A quantitative analysis of security, anonymity and scalability for the lightning network. In *2020 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*. IEEE, 387–396.

[39] Saar Tochner, Stefan Schmid, and Aviv Zohar. 2019. Hijacking routes in payment channel networks: A predictability tradeoff. *arXiv preprint arXiv:1909.06890* (2019).

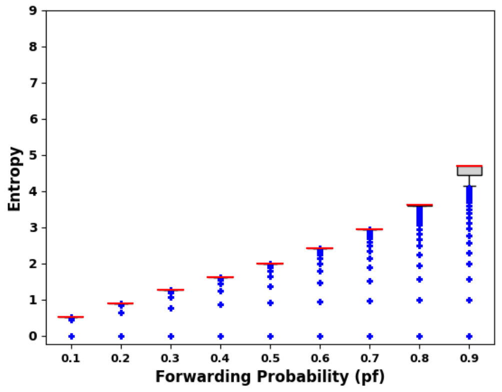
[40] Qiheng Zhou, Huawei Huang, Zibin Zheng, and Jing Bian. 2020. Solutions to scalability of blockchain: A survey. *IEEE Access* 8 (2020), 16440–16455.

A APPENDIX

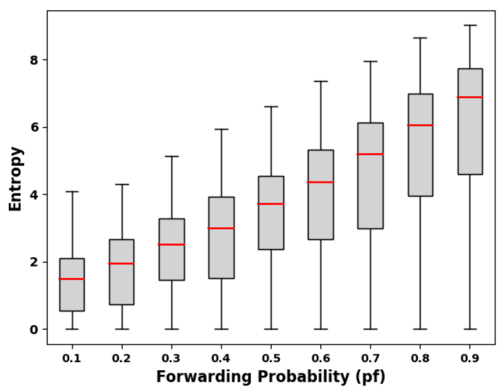
A.1 Reduction of Anonymity Set by a Sophisticated Adversary

1) In LN, are all transactions (irrespective of their amount) equally likely to be deanonymized? To answer this question, we need to revisit how we created shortest paths for our analysis. For computing shortest paths between any two given nodes in LN, we considered all possible paths (consisting of all possible channels) in the LN snapshot. This is because we assumed the minimum amount of transaction to be sent from the source to the destination. Since the smallest transaction can be sent along any available path, we considered all the paths in LN. Thus our results provide an upper bound on the paths that can be selected for Dijkstra's computation.

However, in practice the adversary can perform an even more precise analysis if it considers the *exact amount* being transferred in the transaction. Transactions with larger amount can only be relayed via a subset of all possible paths *i.e.,* only those paths that



(a) Dandelion



(b) Dandelion++

Figure 13: Entropy vs Forwarding probability p_f: With increasing the forwarding probability entropy increases.

have payment capacity at least equal to transaction's amount. This reduces the total number of paths to be analyzed and thus the anonymity set. Moreover, it might also happen that for such bigger transactions, from some nodes there are no paths with adequate capacity to perform the transaction. This could further reduce the anonymity set for higher amount transactions.

2) Can nuanced details of the privacy-subgraph help in reducing the anonymity? In Dandelion++, each node has exactly two immediate successors (and can have more or less than two immediate predecessors) in the privacy-subgraph. Dandelion++ further recommends that transactions received from a predecessor should be forwarded to a fixed successor. However, in our analysis we do not incorporate these intricate details while computing entropy. This is because we believe that its hard for the adversary to know these internal predecessor–successor mappings of each node.

Thus in our analysis, for every node we consider that it sends transactions to *any* one of its two successors, thereby having a more generic analysis, and recall that even without the knowledge of these mappings we obtained very low entropy values. Notably, if some sophisticated adversary somehow obtains this predecessor–successor information, it can do a more precise analysis while estimating the originator of a received transaction.

3) Can different originator probabilities reduce the anonymity? In our current entropy computations, we consider a generic scenario, where we assume uniform priors for the originator probabilities ($P(B_i)$) for each benign node i. However, if a sophisticated adversary has additional knowledge (*e.g.,* knows the (average) frequency of transaction generation for each node i), it can utilize this extra information in the analysis by incorporating it to the priors. In such a case, adversary would consider a different value of $P(B_i)$ for each node i and perform a more informed analysis that further reduces entropy.

Takeaway: In spite of the fact that we do not incorporate the aforementioned details in our analysis (for all the three schemes), we obtained low anonymity values under various different settings. Our results are thus a lower bound, and a more sophisticated adversary can certainly perform a more precise analysis and deanonymize the transactions with even more confidence.

A.2 Impact of forwarding probability on anonymity

Fig. 13 highlights the variation in entropy when different forwarding probability is selected for both Dandelion and Dandelion++.

A.3 Privacy subgraph learning

In this section we now describe the approach in detail (mentioned in Sec. 6.1), that can be used by the adversary to obtain the privacy subgraph in Dandelion++. Subsequently, we also show the results obtained from simulations, and present some heuristics to learn privacy subgraph faster and with lower number of transactions. We now describe the procedure to derive the privacy subgraph in simulations.

- We construct a random 16-regular graph with each node having 8 outgoing neighbours. This is representative of the bitcoin graph (BG).
- Next, we derive a privacy subgraph (PSG) from the BG by selecting 2 outgoing connections randomly for each node as the successor nodes in the PSG.
- Since the adversary nodes already know their predecessor and successor connections, we add them to the derived PSG. Adversary nodes would then send/forward the transaction via the remaining honest nodes. To ensure that at least one adversary node is connected to all honest nodes, every adversary node makes multiple connections to different honest nodes (instead of two).
- The adversary will then pick a honest node through which it will send multiple transactions and will record the nodes that diffuse them. We simulate the stem phase routing for these transactions. Since the forwarding probability assumed in Dandelion++ is 0.9, ≈ 10% of all the transactions will be diffused by the successors of the honest node in the PSG. Thus, we analyze the number of diffusions by neighbours of the honest node in BG. The analysis reveals that the successors of the honest node in the PSG show significantly larger diffusions in comparison to other successors. We thus take the two nodes (out of the eight) with highest frequency

as the actual successors of the node in PSG. We add these edges to the potential PSG derived by the adversary. We send 50 transactions per honest node before analyzing the distribution of diffusions per node.

- The previous step is repeated for all honest nodes and we obtain a PSG.
- We then compare the derived PSG with the actual PSG and calculate the accuracy.

Using the above steps we simulated the graph learning attack on a BG of 1000 nodes with 10% adversary nodes. We observed that more than 90% of the PSG was correctly constructed. Moreover, when we increased the number of transactions to be analysed per node from 50 to 100, the accuracy increased to ≈ 98.5

Additionally, in order to increase the accuracy of results, as well as to minimize the number of transactions to be analyzed per node, the adversary can employ extra analysis. The adversary can not only analyze the distribution of immediate successors of honest nodes, but also the successors of the immediate successors. This would allow the adversary to learn more edges in the graph with the same number of transactions. It would also enable the adversary to further verify that the successor of the honest nodes the adversary identified are indeed correct. This is because if the successors are correctly identified then the successors corresponding to these successors would also show two nodes diffusing large number of transactions

Moreover, once the adversary has learned a large part of the PSG, it can then adopt additional heuristics that may not require sending additional transactions, thereby minimizing the overall transactions required to learn the PSG. The adversary can do so by inferring the edges, by ruling out successors that are already part of the PSG. This elimination will help adversary limit the set of potential successors and even in some cases will only be left with just the two successors that are actually part of the PSG.

Overall, as demonstrated with the above analysis, the adversary can completely construct the PSG by analyzing less than 50 transactions per node.