Secure Embedding of Rooted Spanning Trees for Scalable Routing in Topology-Restricted Networks

Martin Byrenheid
TU Dresden
martin.byrenheid@tu-dresden.de

Thorsten Strufe
KIT Karlsruhe
Centre for Tactile Internet / TU Dresden
strufe@kit.edu

Stefanie Roos
Delft University of Technology
s.roos@tudelft.nl

***Abstract*—Greedy embeddings on rooted spanning trees are the most promising solution to provide sufficiently scalable routing in dynamic networks with restricted topologies, for instance friend-to-friend overlays such as the Dark Freenet and payment channel networks such as Lightning. Yet, they are not deployed in practice, as electing a root and configuring addresses remains an unsolved problem in adverse environments. Indeed, faulty or malicious nodes might provide incorrect coordinates, prevent the network from stabilizing by simulating dynamics, or not start the assignment of coordinates in their subtree at all. All of the above attacks may result in an inability to route.**

To mitigate the above attacks, we design a novel embedding algorithm with an adapted distance metric that only relies on interconnections between benign subtrees for successful delivery. In other words, even if roots of (sub-)trees are malicious or faulty, the remaining nodes still receive coordinates and can communicate with nodes in their tree branch as well as other branches reachable via the neighborhood of their benign ancestors.

Extensive simulations demonstrate that we thus facilitate efficient routing even when seemingly decisive parts of the network are under adversarial control.

I. Introduction

Many overlay networks have connectivity restrictions. Developers in some cases aim to protect the privacy of users and reduce the risk of Sybil attacks by only connecting to trusted nodes. Anonymous and censorship-resistant publication is one key application of such overlays [5]. Another important use case are payment channel networks like Bitcoin's Lightning network that route payments only via links established by locking collateral on the blockchain [18], which leads to another highly restricted topology. Alike censorship-resistant publication, payment services should be resistant to attacks, as well as fast. For both applications, greedy network embeddings present the most promising solution to find low-stretch paths at a low overhead [20], [21].

Greedy embeddings rely on the assignment of a unique logical coordinate to every node, based on a rooted spanning tree. In a decentralized and dynamic network, the embeddings need flexibility to adapt coordinate lengths [8], [9], as they cannot collect global topology information. In the absence of a trusted entity that runs root nodes, existing solutions either resort to performing distributed leader election [20] or – if the overall number of nodes is known – having each node randomly decide whether to operate as root node [14], [24].

Controlling the root is desirable for an attacker, as the current method that protects from interference with the spanning-tree construction, the fundamental functionality to configure the entire network, relies on the root node to be benign [3]. However, we argue that in the presence of network dynamics and malicious nodes interfering with the protocol execution, distributed root election algorithms cannot guarantee successful choice of a root that is non-malicious. Due to the absence of a central authority for admission control, the adversary may create a large number of fake nodes, for each of which he may simulate arbitrary behavior. By owning a large fraction of all nodes in the overlay, the attacker can drastically increase the chance of being elected. Even when running multiple instances of the routing protocol in parallel, control over all roots falling to the adversary cannot be avoided. Thus, in topology-restricted overlay networks, the embedding and routing algorithm need to be tolerant to malicious root behavior.

In this paper, we first analyze the types of attacks that an attacker can perform when they are the root node in a greedy embedding and might control additional nodes. Previous work already evaluated the impact of the attacker dropping messages and showed that having multiple embedding instances can counteract such denial-of-service attacks effectively [20]. In contrast, we define attacks that modify the coordinate assignment by i) assigning incorrect coordinates, ii) keeping the assignment from converging by simulating constant churn and consequently changes in the assignment, and iii) not initiating the assignment (which is commonly the responsibility of the root in such embeddings).

Having argued that these three attacks are the key vulnerabilities, we present two defenses. First, we modify the embedding by limiting the coordinate length to mitigate an attack that causes high routing overhead through assigning too long coordinates. Second, we adapt the distance measure between nodes to facilitate local routing even if coordinates on the path between a node and the root are incorrect or outdated. Furthermore, we explain how our defenses can be combined with existing leader election algorithms such that nodes are guaranteed to obtain coordinates.

We prove that our modified embedding still guarantees successful routing in the absence of attacks. Our simulation-based evaluation shows that our embedding improves the attack resistance considerably. Indeed, for both synthetic and

real-world network topologies, our embedding reduces the number of failed routings by at least a factor 2.

Thus, we notably improve the attack resilience of greedy embeddings, making them a suitable candidate for connectivity-restricted networks that require both high efficiency and resistance to denial-of-service attacks.

II. Related work

We summarize the existing work on tree-based network embeddings in terms of their resilience to attacks.

In the context of Internet routing, existing research on tree embeddings focuses solely on robustness against crash failures. Sahhaf et al. [22] proposed the usage of colored trees [19] and the proactive construction of backup paths to enable successful delivery in case of a single link failure or node crash. Later on, Sahhaf et al. proposed a new embedding algorithm that adapts to network dynamics and crash failures [23]. In this algorithm, nodes explicitly notify their neighbors in case of a failure in order to adapt the embedding.

To increase routing success in the presence of link and node failures, Houthooft et al. [10] proposed a distributed algorithm that constructs multiple embeddings in parallel. The choice of parents depends on a cost function. It considers the length of the coordinates and the number of trees in which this neighbor is already a parent. Based on the weight of each factor, a trade-off can be made between fault-tolerance and routing path length. The authors later extended their algorithm such that it also takes link loads into account during construction [11]. Again, malicious behavior with regard to, e.g., reporting coordinates, is not addressed.

In the context of friend-to-friend overlays, Höfer et al. [9] proposed a modified variant of the embedding by Herzen et al. [8] to enable distributed content storage. As this work does not address faults or attacks, Roos et al. [20] extended the embedding algorithm to provide robustness against faults by constructing multiple parallel embeddings and using backtracking during routing. Furthermore, the authors showed that their embedding enables a high routing success ratio in the presence of nodes that drop received packets, even if the malicious nodes are root nodes. However, it is left open to which extend malicious participants can increase the effectiveness of their attacks by actively interfering with the embedding construction. SpeedyMurmurs, an extension of the embedding specific to payment channel networks, allows privacy-preserving routing but its evaluation does not consider denial-of-service attacks [21].

Sun et al. [25] proposed a data sharing framework for friend-to-friend overlays that uses greedy embeddings for routing. Data publishing is designed such that any new content first passes a universally trusted (or as they call it, *secure*) node that checks if this message may be part of an attack. However, the authors leave open how to select trusted nodes and assume that the root node of the embedding is a trusted node.

In summary, existing work on tree embeddings focuses on crash faults of nodes or links. Constructing multiple parallel embeddings protects against malicious participants that drop any incoming messages, but active attacks against the construction of the embedding have not yet been addressed.

III. Model and Notation

In the following, we present our system model together with the necessary terminology, followed by a description of the adversary model considered in this work.

A. System Model and Terminology

We model a *topology-restricted overlay network* with bidirectional connections at a fixed point in time as an undirected graph $O = (V, E)$, where V denotes the set of nodes and E denotes the set of connections between them. Connected nodes communicate by message passing. However, we consider reordering and loss of messages due to failures of the underlying infrastructure out of scope, as we focus on attacks by malicious overlay participants.

The network is dynamic, such that nodes may join or leave the overlay network at any time. Furthermore, nodes may establish or disestablish connections to other nodes over time. In the following, we call a change of the overlay network, such as the departure or arrival of one or more nodes, a *churn event*. Nodes may also crash, thus leaving the network without notifying their neighbors about their departure.

In our setting, nodes do not have a priori knowledge about the number of nodes $|V|$ or the number of edges $|E|$. However, we assume that all nodes know an upper bound D on the diameter of the network, which we later use to defend against the propagation of maliciously chosen coordinates by attackers. We consider this assumption to be realistic for networks with trust-based connectivity restrictions, as their structure resembles the social graph of its participants and previous studies indicate that even social graphs with millions of nodes have a diameter below 30 [15].

Currently known greedy embeddings rely on the construction of a rooted spanning tree to enable routing. Given an overlay $O = (V, E)$, a rooted spanning tree of O is a tuple $T = (V_T, E_T, r)$ such that $r \in V$ and (V_T, E_T) is a connected subgraph of O with $V_T = V$, $E_T \subseteq E$ and $|E_T| = |V| - 1$. Node r is called the *root node* of T.

For every node $u \in V \setminus \{r\}$, there is only a single path from u to r in (V_T, E_T). Given such a path $u, v_1, v_2, .., v_k, r$ in (V_T, E_T), we say that v_1 is the *parent* of u and that u is a *child* of v_1. We call the nodes $v_i, i \in \{1, 2, .., k\}$ *ancestors* of u and for each v_i, we say that u is a *descendant*.

Furthermore, we denote the length of the path from u to the root r as the *level* of u. In the following, we refer to the number of edges on the longest path from r to any node in (V_T, E_T) as the *depth* of the spanning tree.

B. Greedy network embeddings

Routing based on greedy network embeddings relies on two core parts: an *embedding algorithm* and a *distance metric*. Given a coordinate space **ID**, the embedding algorithm assigns a unique *logical coordinate* from **ID** to every node. In the

following, we refer to such an assignment via a function $C : V \rightarrow \mathbf{ID}$ that maps each node to its logical coordinate.

The distance metric then defines the distance between two logical coordinates and thus also serves as measure of distance between nodes. An assignment of logical coordinates to nodes is called a *greedy embedding* if greedy routing, i.e., forwarding the message to the neighbor whose logical coordinate has the lowest distance to the coordinate of the target, is guaranteed to succeed.

In this work, we focus on vector-based coordinates of varying length, i.e., $\mathbf{ID} = S^*$ for some set S. The assignment is executed in a fully distributed manner as follows: First, a single node r out of all nodes in the network is selected and receives a coordinate. Starting from r, a spanning tree of the network rooted at r is constructed. Children receive a coordinate that is the parent coordinate and one additional element. In other words, whenever a node u becomes the child of another node v with logical coordinate $(c_1, c_2, .., c_k)$, u's coordinate is of the form $(c_1, c_2, .., c_k, c_u)$ for some $c_u \in S$. Thus, the vector assigned to u encodes a path from u to the root node [8], [9], [20].

Since a rooted spanning tree is a connected subgraph over the entire network, all nodes can reach each other by routing over the tree edges. Thus the distance between two logical coordinates C_1 and C_2 is given by means of the *tree distance*

$$\delta_{TD}(C_1, C_2) = |C_1| + |C_2| - 2 \cdot CPL(C_1, C_2) \qquad (1)$$

, where $CPL(C_x, C_y)$ denotes the length of the common prefix of C_x and C_y.

However, the routing of messages is not limited to tree edges only. When routing a message to a coordinate C_t, nodes do not only consider the distance of the logical coordinates of their parent and children to C_t but those of *all* their neighbors in the network. In the following, we call non-tree edges *shortcuts*, as these links can be used to reduce the number of hops needed to reach C_t.

C. Adversary model

In this work, we consider an adversary that aims to perform a large-scale denial of service attack against the overlay network. For overlay networks such as Freenet or GNUnet, the adversary might be a malicious actor that aims to perform censorship. In Lightning, the attacker might want to block payments such that parties make use of other payment methods with higher fees.

We consider an *internal* attack, where the adversary controls a subset of the nodes in the overlay. In the following, we call nodes under control of the adversary *malicious nodes* and the remaining nodes are called *benign nodes*.

As the initial setup of connections in topology-restricted overlay networks requires prior social engineering, which we assume to be costly to perform on a large-scale, the adversary can only establish a bounded number of connections between malicious and benign nodes. In the following, we call connections between malicious and benign nodes *attack edges*.

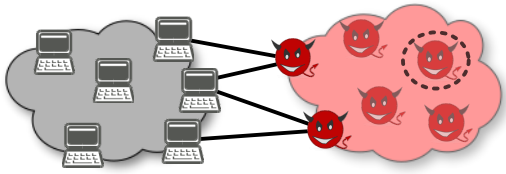


Fig. 1: The adversary is able to introduce fake nodes (indicated by transparency) with arbitrary interconnections. Thus, the root node, marked by a dashed line, may also be a fake node.

The malicious nodes may deviate arbitrarily from the correct behavior, e.g. by dropping and delaying messages or spreading misinformation. In particular, we consider the scenario that the malicious nodes are able to undermine the election of the root node, thus establishing a malicious node as root.

Since we do not assume a centralized admission control, the adversary is furthermore able to simulate additional, arbitrarily interconnected nodes in the network, as illustrated by Figure 1. In the following, we call the network of adversary-controlled nodes together with the simulated nodes the *adversarial region* of the overlay. The network of non-adversarial nodes is called the *benign region* of the overlay.

However, we assume that the adversary does not have any a priori knowledge about the total number of benign nodes and their connections. Rather, he is initially only aware of those benign nodes that are connected to malicious nodes. He is furthermore non-adaptive in the sense that he establishes his connections initially and does not add or remove connections later.

IV. Attacks and Countermeasures

Given that our adversary can only establish a bounded number of attack edges to benign nodes, the adversary uses these edges to perform active attacks in order to maximize the disruption of communication. In the scenario that a malicious node has been elected as root, as shown in Figure 1, the attacker can perform different attacks by varying the length and elements of the logical coordinates sent via the attack edges as well as the timing of these messages.

Given these attack vectors, the adversary may perform the following attacks:

1) **Coordinate duplication:** Malicious nodes propose the same logical coordinate to multiple benign neighbors.
2) **Simulate high diameter:** Malicious nodes announce extremely long coordinates to their benign neighbors.
3) **Simulate high dynamics:** Malicious nodes simulate extreme dynamics in the adversarial region by repeatedly announcing different logical coordinates to their benign neighbors.
4) **Simulate root fault:** Malicious nodes never announce any logical coordinates to their benign neighbors, thus pretending to have lost connectivity to the root node after the election.

The first attack causes routing to fail, as the assignment of logical coordinates is not unique anymore, such that benign nodes forward messages to the wrong nodes. However, we merely include this attack for completeness, as it has already been addressed by Roos et al. [20] by having child nodes obtain their coordinate by appending a random number to the coordinate of their parent.

The second attack does not cause routing to fail, but instead introduces extremely high bandwidth overhead due to excessively long addresses, which significantly lowers throughput.

In the presence of a malicious root node, routing between different subtrees depends the usage of shortcut links. By sending coordinates with different lengths over each attack edge, the third attack causes the benign nodes to frequently change their parents and consequently their logical addresses. As a result, the target coordinate of messages that are in transit become outdated and are routed towards the malicious root node, as benign nodes are unable to detect shortcuts.

In case of the fourth attack, benign nodes will not obtain any logical addresses, thus making routing of messages impossible.

One intuitive countermeasure that limits the damage caused by the aforementioned attacks is to periodically start a new election after a fixed amount of time, starting from a common fixed date. However, while a shorter election period reduces the timespan of the attacks, it also inherently causes higher overhead in the absence of attacks. It is thus desirable to design countermeasures that limit the damage caused by malicious nodes while one of them still acts as root node.

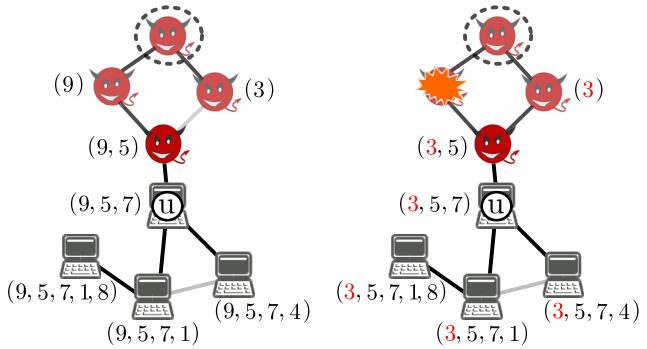
A. *Identifier-Embedding with Bounded-Length Coordinates*

Let our coordinate space be $\mathbf{ID} = \left(\{0,1\}^b\right)^*$, i.e., the set of all vectors with b-bit elements. To address the aforementioned attacks, we modify the assignment of logical coordinates as follows: Upon startup, every node u generates a persistent, random virtual identifier $VID_u \in \{0,1\}^b$. If u becomes the root node, it will use (VID_u) as coordinate rather than an empty vector as in previous algorithms [8], [9]. Otherwise, whenever u chooses a neighbor v with coordinate $(c_1, c_2, .., c_k)$, where $c_k = VID_v$, as parent, it uses $(c_1, .., c_k, VID_u)$ as its own coordinate. If v changes its coordinate to a different sequence $(c'_1, c'_2, .., c'_{k'-1}, VID_v)$ but remains the optimal parent for u (e.g., because it still has the shortest coordinate among u's neighbors), then u sets $(c'_1, c'_2, .., c'_{k'-1}, VID_v, VID_u)$ as its own coordinate.

Figure 2 shows an example, where the adversarial nodes simulate churn to cause non-adversarial nodes to change their coordinates. Due to the identifier-based coordinate computation, all nodes in the subtree rooted at u only change a prefix of their coordinate.

In our modified embedding, the root node r includes VID_r in its coordinate. If there is no attack, the longest coordinate in a spanning tree of depth d will thus have $d+1$ elements.

Without an upper bound on the maximum depth of the rooted spanning tree in the overlay, it is impossible to defend against the case that malicious nodes announce extremely long coordinates, as there is no criterion based on which



(a) Before the simulated churn event. (b) After the simulated churn event.

Fig. 2: For pretended failures in the adversarial region, benign nodes only change the prefix of their coordinates. Black lines denote tree edges and grey lines shortcut edges.

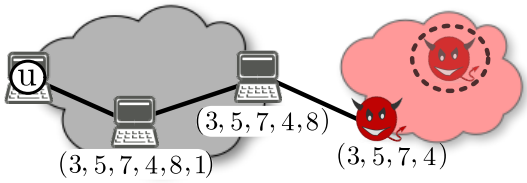


Fig. 3: High diameter attack ($D = 5$). For malicious prefix choice, u's neighbor has a coordinate with maximum length. To u it is opaque if the neighbor or an ancestor is malicious.

the length of the logical coordinates can be limited. Given that the nodes in our setting know the upper bound on the network diameter D, which also serves as upper bound for the depth of the constructed spanning tree, it is a sign of an ongoing attack when the shortest virtual coordinate in a node's neighborhood has length $D+1$ or more. However, as illustrated by Figure 3, a node recognizing this situation is not able to tell which node along the path to the root actually is acting maliciously. Therefore, our embedding algorithm adapts to this situation as follows: If a node u chooses a node v with logical coordinate $C_v = (c_1, c_2, .., c_{D+1})$ as parent, node u removes the first element from C_v before appending its identifier, yielding $C_u = (c_2, c_3, .., c_{D+1}, VID_u)$ as its own coordinate. In the scenario shown in Figure 3, node u will thus use $(5, 7, 4, 8, 1, VID_u)$ as logical coordinate.

If the adversary simulates a high diameter, the spanning tree algorithm should ideally guarantee for every benign node u that the depth of the subtree rooted at u never exceeds D. Otherwise, routing is not guaranteed to be successful. Unfortunately, it can be shown that when parent selection is done solely based on the logical coordinates of a node's neighbors, it is impossible to provide this guarantee in general, as illustrated by Figure 4.

To reduce the likelihood that the subtree rooted at a benign node exceeds the diameter bound, we introduce the heuristic that if two neighbors have one or more common ancestors, nodes prefer the neighbor closest to the nearest common an-

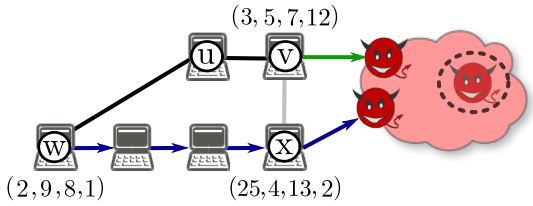


Fig. 4: Example: Depth of subtree rooted at a benign node exceeds the diameter bound ($D = 3$). Arrows denote tree edges with parent-child relationships, color indicates a distinct benign subtree. Grey edges denote shortcut links. As the coordinates of v and w are distinct and of equal length, both are equally well-suited as parent for u. If u randomly chooses w as parent, the depth of the subtree rooted at x exceeds D.

cestor. If neighbors of a node u have non-distinct coordinates with maximum length, this heuristic ensures that u prefers neighbors close to the node with the overlapping identifier. For example, consider the case that v instead chose x as parent in the setting shown in Figure 4 and thus obtained $(4, 13, 2, 12)$ as coordinate. Then, u would detect that v is a child of the node with identifier 2, whereas w with coordinate $(2, 9, 8, 1)$ is 3 hops apart from the node with identifier 2 and thus u would prefer v as parent.

B. Overlap Routing

We now consider a way to use the overlap between coordinates as an indicator of the distance. In our algorithm, we rely on distances between nodes or more specifically node coordinates. Generally, a distance function should map two node coordinates to a non-negative real number. We slightly modify the concept such that the distance function can fail to compute a distance but only in the presence of an attack.

Definition 1. *Let **ID** be a set of potential node coordinates. A function $\delta : \textbf{ID} \times \textbf{ID} \to \mathbb{R}_+ \cup \{\perp\}$ is called an* attack-aware distance function *if $\delta(VID_1, VID_2) = \perp$ indicates that VID_1 and VID_2 can not both be valid coordinates in the same embedding.*

For two coordinates $C_u = (c_1^u, c_2^u, .., c_k^u)$ and $C_v = (c_1^v, c_2^v, .., c_l^v)$, we write $C_u \cap C_v$ to denote the set of common elements, i.e., $C_u \cap C_v = \{c \mid \exists 1 \leq i \leq k, 1 \leq j \leq l : c_i^u = c_j^v\}$. Furthermore, let $s(C_u, c)$ denote the suffix of C_u starting with element c. We define the function $\delta(C_u, C_v)$ as follows:

$$\delta(C_u, C_v) = \begin{cases} \min\limits_{c \in C_u \cap C_v} |s(C_u, c)| + |s(C_v, c)| - 2 & \text{if } C_u \cap C_v \neq \emptyset \\ \perp & \text{otherwise} \end{cases}$$

We write $\delta(u, v) := \delta(C_u, C_v)$ for simplicity.

δ is not a distance function as it does not necessarily return real numbers. However, if δ returns $\perp$ for all coordinates in u's neighborhood (including its own), then u does not drop the packet but instead forwards it to its parent.

We now show that in the absence of attacks, network dynamics and duplicate virtual identifiers, δ results in a greedy embedding, similar to [8], [9]. Hence, the routing is guaranteed to succeed in a stable and attack-free network.

Theorem 1. *δ is an attack-aware distance function in the sense of Definition 1. Furthermore, given that every node u in the overlay has a unique virtual identifier VID_u as well as a correct upper bound D for the overlay diameter and all nodes are benign, the identifier-based embedding algorithm produces a greedy embedding.*

Proof. Recall that the root coordinate is (VID_r) and, since D is larger than the network's diameter, every node's coordinate contains the root identifier VID_r as the first element. Hence, in the absence of attacks, δ does not return $\perp$ and thus is a distance function.

To show that an embedding is greedy, it is sufficient to show that each non-terminal hop has a neighbor closer to the destination [13]. In the following, we write $parent(u)$ to denote the parent of node u in the underlying spanning tree.

Let t denote the target and $u \neq d$ a node. We need to show that u has a neighbor v such that $\delta(v, t) < \delta(u, t)$. As stated above, u's coordinate shares at least the first element with t's coordinate. In the absence of attacks, a node's coordinate corresponds to the parent coordinate and one additional element. Hence, if C_t does not contain VID_u, it shares the same prefix with $parent(u)$ as with u itself but has a shorter suffix. As a consequence, $\delta(parent(u), t) < \delta(u, t)$. It remains to consider the case that C_t contains VID_u. As there are no duplicate identifiers and $t \neq u$, sharing the identifier indicates that t is a descendant of u. By construction of the coordinates C_t has to contain the identifier VID_v of one of u's children v. So, the last common element between t and u is VID_u and the last common element for t and v is VID_v. Since v is the child of u, it holds that $|s(C_t, VID_v)| = |s(C_t, VID_u)| - 1$. In other words, the last common element between t and v appears later in C_t and hence has a smaller suffix. As $|s(C_v, VID_v)| = |s(C_u, VID_u)| = 1$, we have

$$\begin{aligned} \delta(v, t) &= |s(C_v, VID_v)| + |s(C_t, VID_v)| - 2 \\ &= 1 + (|s(C_t, VID_u)| - 1) - 2 \\ &= |s(C_t, VID_u)| - 2 \\ &= \delta(u, t) - 1 \end{aligned}$$

This completes the proof. □

C. Countermeasures Against Root Fault Simulation

To protect against the case that the malicious nodes do not announce any logical coordinates to their benign neighbors, our embedding procedure can be combined with a self-stabilizing leader election procedure, like the one proposed by Datta et al. [7]. These algorithms ensure that during the election, every node learns its distance to the leader node. Thus, the attacker can either remain passive, leading to a benign node being elected or participate in the election, thereby allowing nodes to discover paths to its nodes and thus obtain logical coordinates.

TABLE I: Number of nodes n, average degree $\overline{deg}$, maximum degree deg_{max}, degree variance σ_{deg}, diameter D and average shortest path length $\overline{spl}$ of the graph datasets used for the simulation study.

Graph	n	$\overline{deg}$	deg_{max}	σ_{deg}	D	$\overline{spl}$
Brightkite (BK)	56,739	7.5	1134	424.3	18	4.92
Facebook (FB)	63,392	25.8	2196	6420.5	15	4.32
SPI	9,222	10.58	294	513.7	12	4.67
Barabási-Albert (BA)	9,222	9.99	317	129.8	6	3.72
Erdös-Renyi (ER)	9,222	10.57	27	10.4	7	4.12

V. Evaluation

We implemented a simulation model using the discrete event simulator OMNet++ [26] to evaluate the effectiveness of our countermeasures under adversarial behavior as well as the impact of duplicate node identifiers.

A. Metrics and Datasets

For each of our simulation experiments, we repeatedly selected a random pair of nodes u and v and let u initiate the routing of a packet with v's current coordinate as target. To evaluate the effectiveness of our countermeasures, we measured the *routing success ratio*, defined by the number of packets that have been delivered correctly divided by the total number of created packets. Furthermore, we measured the *number of parent changes*, which is the sum of the number of times each node in the network changed its parent in the spanning tree for the complete simulation run.

We used various real-world and synthetic datasets to assess the impact of network structure on the efficacy of our countermeasures. Since anonymous publishing overlays are explicitly designed to hide the network structure to outsiders as well as overlay participants, there are currently no network snapshots available for simulation studies. Given that the overlays considered in our work are determined by social trust relationships, we instead relied on graphs obtained by crawling online social networks.

Table I summarizes the graph datasets used for our study. All graphs are undirected, as we consider overlays with bidirectional connections. *Brightkite* (BK) stands for a graph acquired from the Brightkite location-based online social network [4]. *Facebook* (FB) denotes the largest connected component of a graph obtained by crawling a fraction of the Facebook social network [27]. Furthermore, *SPI* represents graph data from the online social network of a German university [17]. In each of these graphs, a node represents a user and an edge corresponds to a friendship between the respective users.

All of the considered real-world graphs have a strong variance regarding node degrees. To measure the effect of the degree sequence of the network on our routing scheme, we used `igraph` [12] to create two synthetic networks with the same number of nodes and roughly the same number of edges as the SPI graph. The first synthetic network was generated using the Barabasi-Albert (BA) model [1] with linear preferential attachment, such that its degree sequence follows a power law. The second graph was generated by connecting randomly chosen node pairs, commonly referred to as the Erdös-Renyi (ER) model [2]. Due to the uniform selection of node pairs, this graph has normal distributed degrees.

B. Model and System Parameters

For comparison, we implemented the embedding procedure described in Section IV-A as well as a variant of the state-of-the-art embedding by Roos et al. [20]. Concretely, this embedding variant differs from our embedding as follows:

1) The tree distance metric presented in Section III-B is used for routing.
2) No maximum coordinate length is enforced.
3) Instead of always appending the node's identifier to the logical coordinate of its parent, a new random number is generated and appended each time a node changes its parent.

Our variant of the embedding by Roos et al. disregards privacy protections and backtracking as they are irrelevant for our evaluation. Note that these changes do not affect the routes taken, they merely enable a comparison based on the key differences: the embedding and the distance function. While the embedding by Roos et al. supports routing via multiple trees in parallel, we only considered the case with one tree, as the construction of multiple trees is orthogonal to our defenses. In order to clearly distinguish this variant, we will refer to it as *routing based on tree distance*.

For each of the aforementioned embeddings, we furthermore implemented adversarial behavior. We concentrated on the case that all malicious nodes are colluding and thus modeled them as a single adversarial node.

Note that we do not focus on the decline in routing success as the number of attack edges increases. Rather, the evaluation quantifies how much an attacker with few edges can keep nodes from communicating by simulating churn. We thus simulated an attacker with only two attack edges but vary how the adversary uses these edges to disrupt routing. A key parameter of interest was the *victim distance* d, defined as the overlay hops between the benign endpoints of two attack edges in the benign part of the network.

Adversarial nodes always drop all messages they are supposed to forward on behalf of benign nodes. In addition, we implemented the adversarial behavior that, given network diameter bound D, the adversary announces a coordinate of length 3 to one of the two benign nodes he is connected to and a coordinate of length D to the other. Every Δt_{churn} simulated seconds, the adversary announces new coordinates while also interchanging the length of the announced coordinates.

To evaluate the impact of an attacker with the alternating coordinate length behavior described above, we performed simulations on the SPI, BA and ER graphs for all combinations of the victim distance $d \in \{1, 3, 6\}$ and simulated churn interval $\Delta t_{churn} \in \{4.0, 2.0, 1.0\}$. For the SPI graph, we furthermore performed simulations for $d = 9$. For comparison,

we also obtained results for the case that the adversary does not interfere with the construction of the embedding and only drops incoming packets. In all of the above experiments, we used 63 bits for the length of the identifiers. Due to a lack of time available for the study, we only obtained results for the graphs with 9,222 nodes in the scenarios with an attacker.

To evaluate the impact of duplicated node identifiers on the success ratio of routing in the absence of attacks, we furthermore performed simulations on all of the previously described graphs with a varying number of bits $b \in \{32, 24, 16\}$ used for node identifiers.

C. Set-up

In the absence of attacks, our simulation proceeded as follows:

1) For a given graph $G = (V, E)$ as input, create a network with $|V|$ nodes and for each edge in E, add a corresponding connection to the network. Furthermore, assign every node a randomly chosen b-bit identifier.
2) Select a root node and let this node begin the construction of the embedding.
3) As soon as at least two nodes have a logical coordinate, start choosing random node pairs u, v out of those nodes that already have a coordinate and initiate a transfer from u to the coordinate of v.
4) After 100.000 messages have been delivered or dropped, end the simulation and output the success ratio.

In the scenarios with an attack, we replaced step 2. The simulation first added the adversarial and then connections to two randomly chosen node pairs with the given distance d. Afterwards, the adversary node was set as root. In the scenarios without attack, a randomly chosen benign node was set as leader.

The time between consecutive initiations of transfers is drawn from an exponential distribution with mean value 0.005, which ensures that in the event of an attack, a significant number of packets is affected by changes of the spanning tree due to malicious actions. The simulated delay for the transmission of messages, including those for maintaining the embedding, is 50 milliseconds, as previous studies on latency in the Internet indicate round-trip times around 100 milliseconds [6].

D. Robustness against simulated churn

Since greedy routing based on tree distance is very sensitive to coordinate changes near the root node, we expect the routing success ratio to decrease mainly due to constant prefix changes resulting from simulated churn. In contrast, our algorithm remains mainly unaffected by changes close to the root unless the only path to the destination is via the root node. However, if the structure of the spanning tree changes drastically, routing should be impacted as parent-child relations might be reversed, which can lead to routing in the wrong direction. For our protocol, we thus expect a higher success ratio in the presence of simulated churn, where the gain in success ratio over routing based on tree distance decreases as the distance d between the endpoints of the attack edges increases.

Figure 5 shows the mean routing success ratio for the simulation runs on the SPI, ER and BA graph. There are no results for victim distance $d = 9$ on the BA and ER graph, as these graphs have a diameter below 9.

When greedy routing based on tree distance was used, routing success decreased drastically on all graphs as the time between consecutive coordinate changes by the adversary decreased. In the case that the benign nodes connected to the adversary were neighbors, the mean routing success ratio on the SPI graph decreased from 1 for the case without churn to 0.6 when the coordinates announced by the adversary changed every second. On the BA graph, the mean routing success ratio decreased to 0.68 and for ER, it decreased to 0.58. As expected, the decrease in routing success ratio was similar for all considered victim distance values on each graph. Note that the success ratio without churn is clearly below 1 for $d > 1$, indicating that messages are sent via the root. The larger the distance of the two nodes that the adversary connects to are, the more likely it is that the adversary is seen as the best choice for reaching a different part of the network.

With our embedding and routing algorithm, the mean success ratio remained 1 in all runs with victim distance 1 for the SPI graph as well as for the BA and ER graph. In the following, we explain the reason for the consistently high success ratio and also why routing based on tree distance fails even for $d = 1$.

For $d = 1$, the two neighbors a and z of the attacker are themselves neighbors. Hence, if the attacker pretends to have a coordinate when communicating with a that is at least 2 elements longer than the one he pretends to have when communicating with z, a selects z as a parent and vice versa. As a consequence, the message is never routed via the root and hence never dropped.

When the role of a and z in the parent-child relation switch, routing based on tree distance fails as the common prefix length to each old coordinate is 0. If we use the embedding based on overlapping coordinates, the routing is still successful.

In order to see that, first observe that due to the reuse of the persistent identifiers, the target coordinate C_t of every message must either contain a's or z's identifier. Furthermore, C_t still encodes a path in the graph due to the persistent identifiers making up the coordinates, even if the path is not part of the spanning tree for the current embedding. Hence, once the packet reaches a node on that path, it can be routed to t. More specifically, if a node $u \neq t$ has an identifier corresponding to the i-th element of C_t, u is a neighbor of the node v whose identifier is the $i+1$-th element of C_t. v is perceived as closer to t and hence u routes the packet to v. The process continues until t is reached. When a node forwarding the packet does not have an identifier included in C_t or a neighbor for whom that is the case, it forwards the packet towards the root. If no node on the path to t is reached beforehand, the packet must eventually traverse a or z, as these are the only nodes

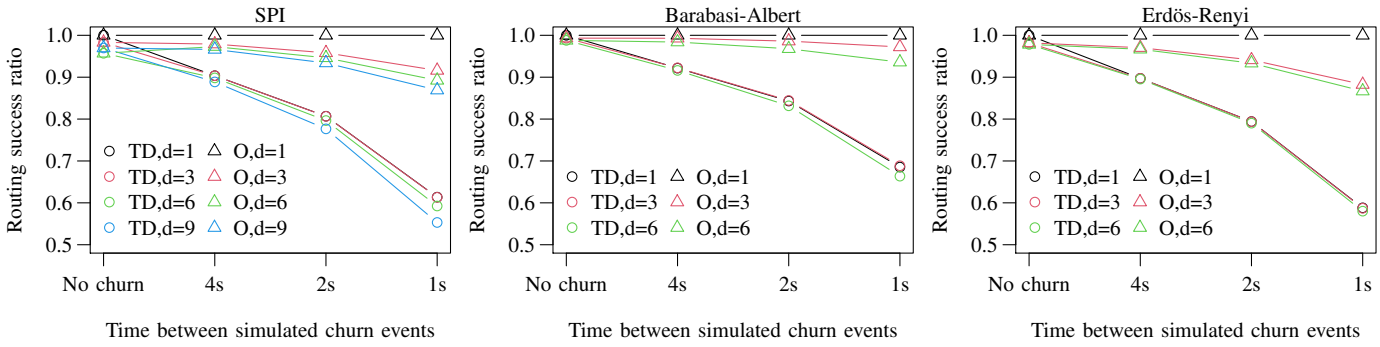


Fig. 5: Every point shows the mean routing success ratio over 50 randomly chosen node pairs with distance d, given that routing based on tree distance (TD) or routing based on overlap (O) was used. 99% confidence intervals were omitted due to their small size.

connected to the malicious root. If a receives the message and C_t contains a's identifier, then a forwards the packet such that it eventually reaches t, as argued above for any node u whose identifier is contained in C_t. If C_t does not contain a's identifier, it must contain z's identifier. Thus, z has a lower distance value to C_t than a. a forwards to z and the packet is then routed to t as explained above. The behavior for the case that z receives the message first is analogous.

In all of the investigated settings, our routing algorithm yielded considerable improvements in the presence of adversarial bahavior. We attribute the stronger improvement for the BA graph to two topological properties: First, due to the lower average shortest path length, packets need to traverse fewer hops. As every hop incurs a delay of 50ms, a lower number of hops leads to a lower timespan in which the transmission may be affected by a simulated churn event. Second, because of the heterogeneous degree distribution, the nodes in subtrees rooted at high degree nodes are less likely to change their parent, as the high-degree node is decisive for their distance to the root node.

To substantiate the above claims, Figure 6 shows the mean number of parent changes per second for the different scenarios. For readability, we omit the results for the routing based on tree distance, as they do not differ significantly from the values for routing based on overlap. In the scenario that the benign nodes connected to the adversary were 3 hops apart, the mean number of parent changes per second on the SPI graph increased from 0 to 3560 when churn was simulated every second. With the same victim distance on the BA graph, the mean number of parent changes per second increased to 4172 when churn was simulated seconds. Conducting the same simulation on the ER graph led to a mean number of parent changes 7181, when coordinates were changed every second. When the benign nodes connected to the adversary were 6 hops apart, the mean number of parent changes per second on the SPI graph increased to 4494 when churn was simulated every second. On the BA graph, the mean number of parent changes per second increased to 5106 and on the ER graph, the mean number of parent changes per second increased to 7392 when churn was simulated every second.

Because the degree sequence of the SPI graph also follows a power law, the increase in the number of parent changes per second is even slightly lower than for the BA graph. Thus, it appears more likely that the stronger decrease in routing success ratio on the SPI graph stems from its significantly higher average shortest path length. In contrast, the average shortest path length of the ER graph is closer to the value of the BA graph, but the mean number of parent changes increases much stronger. Consequently, we consider the stronger decrease in routing success ratio compared to the BA graph to stem mostly from the stronger dynamics of the underlying spanning tree.

E. Impact of coordinate length

Since nodes in our setting generate their identifiers randomly, it is possible that two or more nodes obtain the same identifier. Duplicated node identifiers cause routing to fail in our embedding, as messages will be routed towards the wrong nodes. For our simulations of the scenarios without an attacker, we thus expect the routing success ratio to decrease as the number of bits used for the identifier decreases and when the number of nodes in the network increases.

Figure 7 summarizes the routing success ratio for the scenarios without an attack but a varying number of bits for node identifiers. In all scenarios where 24 bits or more were used for node identifiers, the decrease in routing success is negligible besides the presence of duplicated identifiers.

Routing failures occurred whenever a message with target coordinate $C_t = (c_1, .., c_n)$ reached a node u that coincidentally chose the same identifier VID_u as one of the nodes on the path encoded in C_t. If $VID_u = c_n$, then u erroneously considers itself the target of the message. Otherwise, if $VID_u = c_k$ for $1 \leq k < n$, then u is unlikely to have a connection to a node with identifier c_{k+1}.

We attribute the low decrease in routing success ratio in our simulations to the low average shortest path length, as the spanning tree thus has a low depth and nodes therefore obtain short logical coordinates. Furthermore, due to the low network density, most nodes have a low degree.

In particular, observe that if a node u at level l has k neighbors and all nodes have unique identifiers, then the total

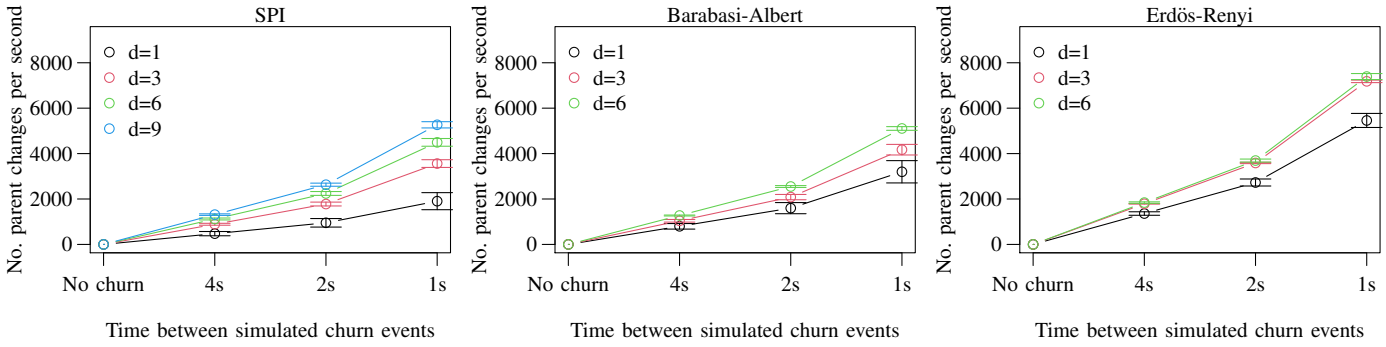


Fig. 6: Every point shows the mean number of parent changes per second over 50 randomly chosen node pairs with hop distance d for the scenario that routing based on overlap was used. The bars around each point denote 99% confidence intervals.

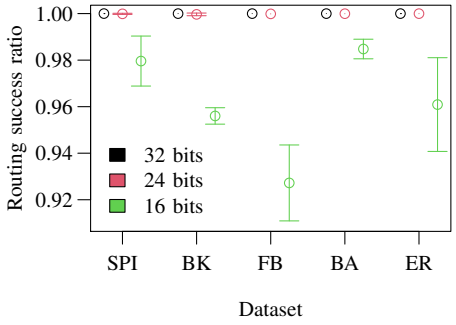


Fig. 7: Each point denotes the mean routing success ratio over 100 runs for the given graph and number b of identifier bits. The bars above and below each point denote 99%-confidence intervals.

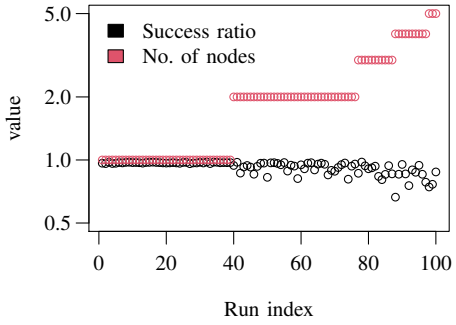


Fig. 8: Number of nodes with the identifier of the leader node and the corresponding routing success ratio for each run on the Facebook graph. The results are ordered according to the number of nodes with the leader's identifier.

number $|N_{VID}(u)|$ of distinct virtual identifiers in u's neighborhood (including itself) can be at most $l + (k-1) \cdot (l+1)$, as nodes choose neighbors with minimal distance to the root node. If identifiers are not unique, the number of distinct virtual identifiers may be slightly higher as in the aforementioned term, because a node u in our embedding may not choose the neighbor v with the lowest distance to the root as parent if a node with the same identifier as u already occurs on the path from v to the root node.

Since nodes generally have a low degree and short coordinates, the set of distinct virtual identifiers $N_{VID}(u)$ that is compared to the packet's target coordinate (which also only contains few elements) at every node u on the path is very small compared to the set of all identifiers in the network. In the graphs with high-degree nodes (BA, Facebook and Brightkite), the nodes with high degrees were always very close to the root node and thus had short logical coordinates. Furthermore, due to their low distance, most nodes that were neighbors of high-degree nodes chose them as parent, thus only adding one identifier to the set of identifiers in the neighborhood of the high-degree nodes.

When 16 bits were used for the node identifiers, the mean routing success ratio on the real-world graphs decreased notably as the number of nodes increases. It turns out that in these settings, it happened that multiple nodes generated the same identifier as the root node. In our model, all nodes with the identifier of the leader act as root node and thus start the construction of a rooted spanning tree. The decrease in routing success then resulted from the fact that nodes in different trees are unable to reach one another.

As an example, Figure 8 shows the routing success ratio together with the number of nodes with the leader's identifier on the different runs on the Facebook graph. Please note the logarithmic scale of the plot. In all runs where only one node had the identifier of the leader, the routing success ratio was very close to 1.0, whereas in the runs with multiple root nodes, routing success dropped by up to 0.34.

The ER graph was more strongly affected by the low number of identifier bits than the SPI and BA graph. We attribute this result to the fact that due to the mostly similar node degrees, the different spanning trees resulting from duplicated leader identifiers have roughly the same size, thus separating almost equally large numbers of nodes from one another.

F. Summary of Results

We evaluated the routing success of our embedding algorithm in the presence of a malicious root node as well as in the absence of attacks. Our results show that our embedding has an up to 31.6% higher probability of successful routing in the presence of actively malicious nodes than existing embedding algorithms. Furthermore, our results indicate that

the routing success ratio is mostly unaffected by the presence of duplicated identifiers, given the network is sparse and has a low diameter and as long as the identifier of the root node remains unique.

VI. Privacy considerations

One of the key goals of the embedding in [20] is privacy. In particular, the authors provide receiver anonymity through the use of keyed hashes and change coordinates frequently to prevent tracing and inferences of relationships. In the following, we discuss to what extent we can achieve the same privacy guarantees.

a) Recipient privacy: In the original scheme for anonymous return addresses, the receiver computed an obfuscated coordinate using a random key and a preimage-resistant hash function. For each element, the hash function derives a corresponding element for the anonymous address by hashing the element together with an input derived from the key and hashes of preceding elements. In our scheme, the routing should still work even if previous coordinates change, which is impossible if the input to the hash function depends on the remaining elements of the coordinate. Hence, we adapt the computation of anonymous coordinates as follows. Given a logical coordinate $C = (c_1, c_2, .., c_n)$ and a random key $\tilde{k}$, we compute the return address $y = (d_1, .., d_n)$ with $d_j = h(\tilde{k} \oplus c_j)$ for all $1 \leq j \leq n$. If h is a preimage-resistant hash function, the attacker can infer the original element only with negligible probability.

In addition, anonymous return addresses use padding to hide the length of their coordinates. In this manner, a node cannot be sure if a message is for a neighbor or a descendant of the neighbor. If the adversary only proposes coordinates with maximum length, nodes cannot pad their coordinates in our algorithm without further measures. To mitigate this issue, nodes may choose a random number k, remove the first k elements of their coordinate and add k random numbers as padding. However, such an action might further decrease the routing success ratio.

b) Traceability: The persistent addressing introduced in Section IV allows long-term traceability of nodes based on identifiers. In other embeddings, nodes change their coordinate frequently, making it intuitively harder to trace their activities. Furthermore, overlays such as Freenet [5] aim to hide the topology of the graph as they correspond to personal trust relationships. When using an identifier with multiple parent nodes, each such parent-child relation reveals a link that over time might enable the attacker to reconstruct the underlying social graph. Knowing the social graph can enable the attacker to infer the otherwise hidden real-world identity of another user [16], which is generally undesired.

However, even the original algorithm reveals relationships, though likely to a lesser degree. Future work is needed to quantify the information leakage to determine whether additional protections are required. In situations such as payment channel networks that do not consider topology information sensitive, our algorithm is certainly an adequate solution.

VII. Conclusion

We proposed and evaluated an alternative embedding and routing protocol for connectivity-restricted overlays. With our new protocol, we considerably improve the resilience to attacks by focusing on the overlap of coordinates rather than the common prefix. In that manner, we mitigate severe but realistic attacks from a malicious root node.

While the current algorithm is appropriate for networks like Bitcoin's Lightning network that publish topology information, it remains unclear if the additional information leakage is acceptable for networks that explicitly aim to hide such information. For such networks, future work regarding the privacy implications of our design is necessary.

References

[1] Albert-László Barabási and Réka Albert, *Emergence of scaling in random networks*, Science (1999), 509–512.

[2] Béla Bollobás, *Random graphs*, Cambridge university press, 2001.

[3] M. Byrenheid, S. Roos, and T. Strufe, *Attack-resistant spanning tree construction in route-restricted overlay networks*, SRDS, 2019.

[4] Eunjoon Cho et al., *Friendship and mobility: user movement in location-based social networks*, Knowledge discovery and data mining, 2011.

[5] Ian Clarke et al., *Private communication through a network of trusted connections: The dark freenet*, 2010.

[6] Frank Dabek, Russ Cox, Frans Kaashoek, and Robert Morris, *Vivaldi: A decentralized network coordinate system*, ACM CCR **34** (2004).

[7] Ajoy K Datta, Lawrence L Larmore, and Hema Piniganti, *Self-stabilizing leader election in dynamic networks*, SSS, 2010.

[8] Julien Herzen, Cedric Westphal, and Patrick Thiran, *Scalable routing easy as pie: A practical isometric embedding protocol*, International Conference on Network Protocols, IEEE, 2011.

[9] Andreas Höfer, Stefanie Roos, and Thorsten Strufe, *Greedy embedding, routing and content addressing for darknets*, NetSys, 2013.

[10] Rein Houthooft et al., *Fault-tolerant greedy forest routing for complex networks*, Reliable Networks Design and Modeling, 2014.

[11] ———, *Robust geometric forest routing with tunable load balancing*, INFOCOM, 2015.

[12] igraph core team, *igraph - The network analysis package*, https://igraph.org/, May 2020.

[13] Robert Kleinberg, *Geographic routing using hyperbolic space*, INFOCOM, IEEE, 2007, pp. 1902–1909.

[14] Jernej Kos et al., *U-sphere: Strengthening scalable flat-name routing for decentralized networks*, ComNets (2015).

[15] Alan Mislove et al., *Measurement and analysis of online social networks*, IMC, 2007.

[16] Arvind Narayanan and Vitaly Shmatikov, *De-anonymizing social networks*, Symposium on Security and Privacy, IEEE, 2009, pp. 173–187.

[17] Thomas Paul et al., *The students portal of ilmenau: A holistic osns user behaviour model*, Tech. report, Palestine Polytechnic University, 2016.

[18] Joseph Poon and Thaddeus Dryja, *The bitcoin lightning network: scalable off-chain instant payments*, 2016.

[19] S. Ramasubramanian, H. Krishnamoorthy, and M. Krunz, *Disjoint multipath routing using colored trees*, ComNets (2007).

[20] Stefanie Roos, Martin Beck, and Thorsten Strufe, *Anonymous addresses for efficient and resilient routing in f2f overlays*, INFOCOM, 2016.

[21] Stefanie Roos et al., *Settling payments fast and private: Efficient decentralized routing for path-based transactions*, NDSS, 2018.

[22] Sahel Sahhaf et al., *Single failure resiliency in greedy routing*, Design of Reliable Communication Networks, 2013.

[23] Sahel Sahhaf et al., *Experimental validation of resilient tree-based greedy geometric routing*, Computer Networks **82** (2015), 156–171.

[24] Ankit Singla et al., *Scalable Routing on Flat Names*, CoNEXT, 2010.

[25] Yanbin Sun et al., *Secure data sharing framework via hierarchical greedy embedding in darknets*, MONET (2019).

[26] Andras Varga, *OMNeT++ Discrete Event Simulator*, https://omnetpp.org/, November 2018.

[27] Bimal Viswanath et al., *On the evolution of user interaction in facebook*, Workshop on Online social networks, 2009.