Susceptibility Analysis of Structured P2P Systems to Localized Eclipse Attacks

Daniel Germanus, Robert Langenberg, Abdelmajid Khelil, Neeraj Suri
CS Department, TU Darmstadt
{germanus,langenberg,khelil,suri}@cs.tu-darmstadt.de

***Abstract*—Peer-to-Peer (P2P) protocols are susceptible to Localized Eclipse Attacks (LEA), i.e., attacks where a victim peer's environment is masked by malicious peers which are then able to instigate progressively insidious security attacks. To obtain effective placement of malicious peers, LEAs significantly benefit from overlay topology-awareness. Hence, we propose heuristics for Chord, Pastry and Kademlia to assess the protocols' LEA susceptibility based on their topology characteristics and overlay routing mechanisms. As a result, our method can be used for P2P protocol parameter tuning in order to substantially mitigate LEAs. We present evaluations highlighting LEA's impact on contemporary P2P protocols. Our proposed heuristics are abstract in nature, making them applicable plus customizable for many other structured P2P protocols. We validate our model's accuracy through a simulation case study.**

***Index Terms*—Peer-to-Peer Security, Eclipse Attack, Overlay Topology Analysis**

I. Introduction

Peer-to-Peer (P2P) computing emerged from file sharing requirements to become a viable alternative for the traditional client/server computing in many diverse application areas such as critical information infrastructures, searches, video streaming, social networks, etc. The expanding usage of the P2P model underlies the high value of enhancing P2P resilience to disruptions and attacks. P2P networks are not only vulnerable to attacks from the environment but also to malicious peers. This is mainly due to the anonymity and typical autonomy of peers to join and leave the overlay network. Subsequently, a spectrum of P2P-specific attacks have been documented on varied P2P-based software systems. Examples include Index Poisoning [1], Sybil [2] and Eclipse [3] attacks.

P2P's progression towards critical applications imposes high resilience needs not only on the overlay itself but on individual peers as well. In such applications, an attacker might insidiously damage the system by carefully selecting a subset of peers that get eclipsed by malicious peers. This Eclipse Attack (EA) variant is called Localized Eclipse Attack (LEA) [4].

LEAs usually require a distinguished attacker who has knowledge about the application running on top of the P2P system. LEAs may result in censorship of individual files, incapacitated game players, or blackouts in a power grid. Thus, consequences of a successful LEA may reach from disgruntled file sharers to hazardous ramifications for human users.

Research partially funded by DFG GRK 1362 (TUD GKmM).

The concept of LEAs has been extensively discussed in literature [4], [5], [6], [7], [8], [9] and was evaluated in the context of the Kad network which is dedicated to file sharing purposes and based upon the Kademlia [10] protocol. Yet, a generic and multi-protocol applicable analytical evaluation of LEAs is still lacking. We aim to close this gap and provide an abstract heuristic that can serve as a template for P2P protocol-specific heuristics to approximate the degree of malicious message interception induced by LEAs. We provide heuristics for three dominant structured P2P protocols, i.e., Chord [11], Pastry [12] and Kademlia [10], and validate them using simulations. The presented heuristics also aid P2P protocol parameter tuning in order to mitigate LEA attacks by increasing attackability costs, and identify the intrinsic weakness of structured overlay routing schemes, i.e., the concept of predictable proximities and hop wise distance reductions.

A. Contributions

Our key contributions that help in assessing and mitigating LEA threats are:

(i) Defining an abstract heuristic that reflects overlay routing mechanisms of multiple P2P protocols. We provide a detailed overview how common P2P protocol concepts relate to that abstract heuristic (Section III).
(ii) Development of heuristics for Chord, Pastry and Kademlia to assess the level of malicious message interception under LEA conditions (Section IV).
(iii) Validation of the proposed heuristics in a simulation case study and considering different LEA strategies. The study shows heuristic accuracy of over 90% (Section VI).

B. Paper Structure

Section II introduces the technical preliminaries. In Section III we present an abstract heuristic for overlay proximities and utilize it to develop (a) protocol specific heuristics in Section IV and, (b) generalized LEA strategies in Section V. The protocol-specific instantiations and simulation case study appears in Section VI. For fuller context, the related work is presented in Section VII.

II. Preliminaries

This section presents technical foundations behind our work. First, we provide a generalized model of structured P2P protocols and outline protocol-independent common concepts.

Next, we present P2P protocol concepts relevant to plan and conduct EAs. Finally, we propose a brief EA taxonomy that helps in understanding EA variations, such as LEAs.

A. Common Notions & Concepts

The P2P overlay topology is modelled as a directed graph $D = (P, E)$, where each peer is part of the peer set P. Each peer maintains its routing state to enable interaction with its overlay network peers. If peer a's routing state points to peer b, there exists a directed edge $e = (a, b)$ in E. The set $E^-(a)$ denotes incoming edges to peer a and $E^+(a)$ denotes outgoing edges from peer a. Furthermore, $E(a) = E^-(a) \cup E^+(a)$. In structured P2P overlays, edges between peers are usually asymmetric though close-by peers display a high probability of symmetric edge relationships.

Over an attack, the peer set P is divided into a benign peer set $B \subset P$, a malicious peer set $M \subset P$ and a victim peer set $V \subset P$. We assume that $P = B \cup V \cup M$ and $N = |P|$.

Applicable to most structured P2P protocols we present four key concepts that serve as basic assumptions driving our abstract and protocol specific heuristics in Sections III and IV.

(i) *ID space distance function:* Computes the ID space distance across two overlay IDs. Peers are assigned IDs from the overlay ID space typically as integer numbers in the range of $[0, \ldots, 2^{128} - 1]$.

(ii) *Overlay routing:* For an arbitrary routing path from a source peer to a destination peer, the ID space distance between the current hop and the destination decreases on each subsequent routing hop towards the destination. Usually, routing mechanisms choose the next hop with the highest ID space distance reduction.

(iii) *Routing state granularity:* Peers store more routing state information of close by peers than of peers further away. In an overlay network with N peers, routing state information per peer is usually of size $c \cdot \log(N)$ with c being a protocol specific constant. This is a protocol class design aspect to leverage scalability.

(iv) *Maintenance protocol:* Continually keeps peers' routing state information updated and handles proper insertion (removal) of joining (leaving) peers.

B. EA Taxonomy

This brief taxonomy provides an overall picture of EAs and their variants. We first detail the assets in a P2P system. Next, we present various EA mechanics. Finally, we discuss various aspects of attacker knowledge that helps the attacker to reduce cost and increase the severity of the EA.

1) Assets: The key assets in a P2P system are data, keys, peers and the overlay network. We relate to a datum in terms of payload which is stored, replicated or transmitted by peers. A datum is usually assigned to at least one key, e.g., as a key/value tuple in a Distributed Hash Table (DHT).

2) Action on Interception: LEAs target overlay message interception. On interception, malicious peers can take various actions. The trivial action is to drop messages with specific characteristics, e.g., based on the sending or receiving peer's identity. Moreover, messages could be forged, replayed, altered, delayed, or simply passed through.

3) Direction: We differentiate between (i) in-attacks, i.e., messages destined towards a set of victim peers should be intercepted, or (ii) out-attacks in case messages originating from the victim peer set should be intercepted. Naturally, a combination of in- and out-attacks is possible.

4) Placement: The placement strategy of malicious peers depends on which assets are targeted. In case the overall overlay is targeted, randomly placing malicious peers throughout the overlay is a possible approach for an attack. Also, an attacker could consider the application's data model or P2P protocol details to place malicious peers in the proximity of specific peers which are required for proper overlay operation.

In case individual peers are targeted, the EA is considered a LEA. Often, placements close to the victim peer are beneficial, where closeness can be expressed in terms of the overlay's ID space distance or other distance metrics, e.g., latencies of the underlay network infrastructure.

The exact placement with a specific overlay ID is typically not possible. Depending on the overlay protocol and the application requirements, an external identifier is required as input parameter to compute the overlay ID. Examples for external identifiers are the IP address, a fixed random number which is created upon peer installation, or certificates.

5) Propagation: The propagation of malicious peers refers to the prevalence of routing state entries in benign peers which point to malicious peers. It depends on the specific P2P protocol's maintenance and lookup mechanisms to propagate routing state. Also, we differentiate between passive and active propagation. For the passive propagation case, malicious peers behave similar to benign peers. In the context of active propagation, malicious peers selfishly try to propagate their routing state.

6) Timeline: A LEA typically entails a start time and duration. In addition, a preparation phase is required to propagate the malicious peers' presence in the overlay.

7) Attacker Knowledge: Knowledge about the assets, network infrastructure and overlay topology characteristics (as introduced in Section II-A) helps attackers to increase the efficiency of an EA. Here, efficiency relates to increased EA severity and reduced cost for the attacker. A P2P overlay serves as the technical foundation for applications that use the overlay's self-organization, routing mechanism, or data storage and replication capabilities. Consequently, knowledge about the application's data model and accordingly which key or datum is critical for the application is a useful information for an attacker to determine e.g. which peers to eclipse.

III. Proximity Based Abstract Heuristic

The proposed abstract heuristic builds upon overlay message routing in structured P2P protocols utilizing the key concept of overlay proximity. This abstraction will also serve as the template for deriving protocol-specific heuristics in Section IV. The heuristic represents the basis for a systematic investigation of structured P2P protocol LEA susceptibility. The assumption

for LEAs is that a victim peer's proximity is populated by malicious peers which can intercept messages addressed to the victim and conduct insidious actions on it. To set the context for the heuristic, we first present some key notions of proximity and overlay routing.

A. Overlay Proximity & Overlay Routing

The overlay proximity of a peer p denominates a set of peers close to p. The amount of peers in the proximity set and the overlay ID space range that is occupied by the proximity differ depending on the P2P protocol and overlay network size. Chord [11] and Pastry [12] have proximity concepts embedded in their routing algorithms. Therefore, we group these two in the following subsection. After that, we discuss the proximity notion of Kademlia's [10] routing algorithm.

1) Chord & Pastry: Chord and Pastry consider the proximity as an integral part of their two-tier message routing algorithms. As a first step, the algorithm checks if the message's destination is closer to a peer in the proximity. If no appropriate peer was found in its proximity, then the routing table is considered for selecting the next hop. Chord terms this proximity as a successor list; in Pastry it is called a leaf set. Successor list and leaf set are fixed size data structures and separate from the routing table that is queried in the second step as described before. Chord and Pastry's routing mechanisms are usually recursive, i.e., the originator passes the message away and surrenders control to the next hop which has to continue the routing mechanism recursively until the message reaches the destination.

2) Kademlia: Kademlia's routing algorithm follows a one-tier approach and its routing state consists of a single data structure which is interpreted as a tree. Leaves of the tree are assigned to so called buckets, which are lists that hold routing state information of other peers. With increasing depth from the root of the routing state tree, buckets contain routing state information of closer peers. ID space distance is measured using a binary XOR metric, i.e., the distance of two overlay IDs a and b equals a XOR b. The result of the XOR metric computation is the Common Prefix Length (CPL) and denotes the closeness of two IDs. Therefore, we define the Kademlia proximity as the set of peers in the three non-empty buckets with highest depth from the tree root. Routing in Kademlia is iterative. The message forwarding and delivery mechanism is separated in a lookup and delivery part. The lookup process is parallelized for performance and fault tolerance reasons. The message will be delivered directly in case the lookup process was successful and the message originator has found the destination's IP address. Using this approach instead of recursive message forwarding, Kademlia peers do not surrender control to other and potentially unknown peers.

B. Overlay Proximity Types

So far, we considered proximity notions in terms of routing mechanisms. Now, we discuss how proximities span across the ID space, either symmetrically or asymmetricall. This notion is important to understand LEA strategies directed towards proximities which will be defined in Section V. The following definitions refer to the proximity of a peer v and its ID is denoted by $id(v)$.

We define λ as the average distance between any two peers in the given overlay ID space and the protocol specific amount of peers that constitute the proximity of a peer as κ.

Hence, the expected proximity width ε is defined as:

$$\varepsilon = \lambda \cdot \kappa \tag{1}$$

A symmetric proximity range is defined as:

$$[id(v) - \frac{1}{2}\varepsilon, id(v) + \frac{1}{2}\varepsilon] \tag{2}$$

Furthermore, an asymmetric proximity range with proximate peers having lower IDs than v is defined as:

$$[id(v) - \varepsilon, id(v)[\tag{3}$$

An attacker can deduce the ID space range of each victim peer's proximity based on the victim set V in order to prepare the desired LEA.

C. Definition of the Abstract Heuristic

On this background, the proposed abstract heuristic develops a probabilistic basis to distinguish if a message is delivered to its destination via peers from the destination proximity or from distant peers. We chose a probabilistic approach since deterministic solutions are not feasible due to (a) the lack of full formal specifications for many structured P2P protocols, and, (b) given the dynamic nature of P2P overlays as caused by user behavior and/or peer or infrastructure perturbations.

In overlay routing, messages are either forwarded or delivered. Delivery implies that the destination is known whereas forwarding implies that the message is passed to a peer that is usually closer to the destination than the current peer. Messages can be delivered either via proximate peers or further distant peers which are not part of the destination proximity. We call the link between two proximate peers a Short Distance Edge (SDE), and Long Distance Edge (LDE) between distant peers. We observed in experiments that overlay message delivery via an SDE is significantly more likely than via an LDE. Our abstract heuristic reflects this correlation.

The Overlay Distance Class (ODC) concept allows a consideration independent of routing paths and their lengths. This abstraction reflects overlay message routing in arbitrary structured overlays. In Figure 1, three ODCs are depicted. The ODC with index l denotes the class of peers which require exactly $l + 1$ overlay hops towards a destination peer (bold arrows). On behalf of the overlay routing mechanism, three exceptions exist (dashed arrows): (i) LDEs from ODC ≥ 1 pointing to the destination (edge (a, v)), (ii) LDEs that point to an ODC which is not neighbored (edge (b, k)), and (iii) SDEs from an ODC ≥ 1 (edge (j, v)). The height of the rectangles illustrates the size of classes. Peers k, m and j are located in the proximity of destination peer v. Unlabeled arrows across ODCs denote message forwarding, and arrows pointing to peer

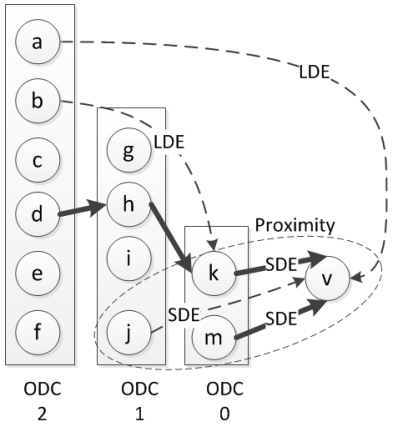

Fig. 1. Overlay Distance Classes (ODCs) for an overlay with not more than 3 hops and with v as destination peer

v denote message delivery. The delivery from proximate peers occurs along SDEs (edges (k, v), (m, v) and (j, v)) or LDEs of distant peers (edge (a, v)). Consequently, the proposed abstract heuristic is essentially a set of probabilities as in the following 5 steps. Each of these abstract 5 steps can be specifically instantiated for target protocols as will be shown in Section IV.

Step 1: $p_{SDE}(v, l)$ is the probability that a peer in ODC l has an SDE to v. This reflects the likelihood of message delivery via peers belonging to ODC l which are also close by the destination w.r.t. their overlay ID.

Step 2: $p_{LDE}(v, l)$ is the probability that a peer in ODC l has an LDE to v. This reflects the likelihood of message delivery via distant peers belonging to ODC l.

Step 3: $\overline{p}_{SDE|LDE}(v, l)$ is the probability that a peer in ODC l has neither an SDE nor an LDE to peer v. This reflects the likelihood of message forwarding by peers in ODC l to an intermediate peer on the overlay routing path towards the destination.

Thus, in a message loss free scenario, the following holds:

$$p_{SDE}(v, l) + p_{LDE}(v, l) + \overline{p}_{SDE|LDE}(v, l) = 1 \quad (4)$$

Moreover:

Step 4: $p^*_{SDE}(v, l)$ is the probability that a message sent from a peer in ODC l will be delivered via an SDE. This is an aggregated likelihood and refers to an overlay routing sequence with the message originator in ODC l and the delivery along an SDE.

A compact representation of $p^*_{SDE}(v, l)$ is given by Equation 5:

$$\begin{aligned} p^*_{SDE}(v, l) = \quad & p_{SDE}(v, l) \\ + \quad & \sum_{i=1}^{l} \left(p_{SDE}(v, l-i) \cdot \prod_{j=0}^{i-1} \overline{p}_{SDE|LDE}(v, l-j) \right) \end{aligned} \quad (5)$$

Step 5: $p_{SDE}(v)$ is the probability that any message sent from any peer will be delivered via an SDE. This is the final step of the abstract heuristic.

In order to compute the overall probability $p_{SDE}(v)$, the ODCs' $p^*_{SDE}(v, l)$ for $l = 0 \ldots l_{max}$ need to be weighted with respect to the ODC class size. The protocol-specific fraction of peers belonging to each ODC is given by the weight function $g(l)$. Thus, we modify Equation 5 to consider the ODC size distribution of the specific target P2P protocol:

$$p_{SDE}(v) = \sum_{l=0}^{l_{max}} g(l) \cdot p^*_{SDE}(v, l) \quad (6)$$

The parameter l_{max} specifies the ODC with highest distance to target peer v ($l_{max} = 2$ in Figure 1).

IV. P2P Protocol-Specific Heuristics

For the heuristic steps of p_{SDE}, p_{LDE} and $\overline{p}_{SDE|LDE}$ developed in Section III, we now provide specific formulae for Chord, Pastry and Kademlia. Based on this, it is possible to compute the probability of malicious message interception from a peer's proximity for varied LEA strategies.

Heuristics are based upon the P2P protocol specifications. The following three assumptions have been made: (i) peers in the proximity have propagated to other peers in their ID space region, (ii) intermediate hops always reduce the distance towards the destination, (iii) routing state in peers is well populated. These assumptions simplify the dynamics of the overlay network and explain the deviations of heuristics and simulations that will be presented in Section VI.

A. Heuristic for Chord

We assume an arbitrary peer p_i that belongs to ODC i. $N(i)$ denotes the expected number of peers between p_i and v in Chord's ID space, i.e., small values of i result in small values returned by $N(i)$ which we define as:

$$N(i) = \frac{N}{s(i)} \quad (7)$$

The distance of each subsequent hop during overlay message routing towards the message's destination peer quarters according to Chord's specification. This is reflected by dividing N by $s(i)$ which we define as:

$$s(i) = 2^{2(l_{max}-i+1)} \quad (8)$$

Basically, $s(i)$ increases by factor of 4 over each subsequent hop.

Furthermore, we define

$$p_{SDE}(v, i) = \begin{cases} 1 & \text{if } N(i) \leq \kappa \\ \frac{\kappa^2}{N(i)^2} & \text{otherwise} \end{cases} \quad (9)$$

This formula approximates the probability of peer p_i having a SDE towards v. This probability decreases for increasing distances between the two peers. Thus, equation 9 returns:

- 1 if $N(i)$ is smaller or equal to the number of SDEs (=κ)
- a quadratic decrease if $N(i)$ is larger than κ

To compute $\overline{p}_{SDE|LDE}(v,i)$, a definition of the probability function $p_{LDE}(v,i)$ is required:

$$p_{LDE}(v,i) = \frac{(\log_2(N) - 2 \cdot (l_{max} - i)) \cdot s(i)}{3 \cdot N} \quad (10)$$

The expected amount of LDEs stored in each peer's routing state is $\log_2(N)$. On each hop towards the destination the amount of LDEs possibly pointing at v decreases by 2 on average while the amount of peers decreases by factor $s(i)$.

On subsequent routing hops, the amount of peers decreases as well as the number of LDEs towards the destination. The relation between these two is required to estimate the probability of having an LDE to the destination. Therefore, the number of remaining LDEs is divided by the number of remaining peers. The division in Equation 10 by N results in the probability of an LDE pointing at destination v. The factor of $\frac{1}{3}$ is a calibration parameter that has been derived from simulation experiments.

Thus, for Chord $\overline{p}_{SDE|LDE}(v,i)$ is obtained as:

$$\overline{p}_{SDE|LDE}(v,i) = (1 - p_{LDE}(v,i)) \cdot (1 - p_{SDE}(v,i)) \quad (11)$$

$g(i)$ is a weight function for the $p_{SDE}(v)$. It specifies the number of peers for the i-th ODC. We present the corresponding values in Figure 2 and set:

$$g(i) = \frac{3}{s(i)} \quad (12)$$

The amount of peers in ODC i is one quarter the amount of peers in ODC $i+1$. Consequently, ODCs further away from v tend to hold exponentially more peers than closer ones. This guarantees a number of hops that is logarithmically dependent on N, as each message forwarding hop traverses at least one ODC according to our assumption.

With the previous definitions, all terms of Equation 6 are defined. The heuristic computations for other overlay network sizes are presented in Section VI.

B. Heuristic for Pastry

Equation 13 calculates the probability of an arbitrary peer's membership in ODC i. The distribution has been calculated for different overlay network sizes and is also depicted in Figure 3.

$$g(i) = \frac{2^b - 1}{2^{b \cdot (l_{max} - (i+1))}} \quad (13)$$

Similar to Chord, the ID space distance decreases on each hop. According to Pastry's specification, the distance to a destination peer v decreases by a factor of $s(i) = 2^{b \cdot (l_{max} - (i+1))}$.

Pastry uses a prefix based distance notion which is encoded with a positional numeral system typically of base 16, i.e., $b = 4$. All peers share at least a prefix of length 0, therefore $N(l_{max})$ yields the number of all peers in the overlay.

Peers with identical shared prefix lengths are in the same ODC. With increasing shared prefix length, the number of peers in an ODC decreases exponentially. $N(i) - N(i-1)$ denotes the number of peers in ODC i. For Pastry, $N(i)$ is defined as:

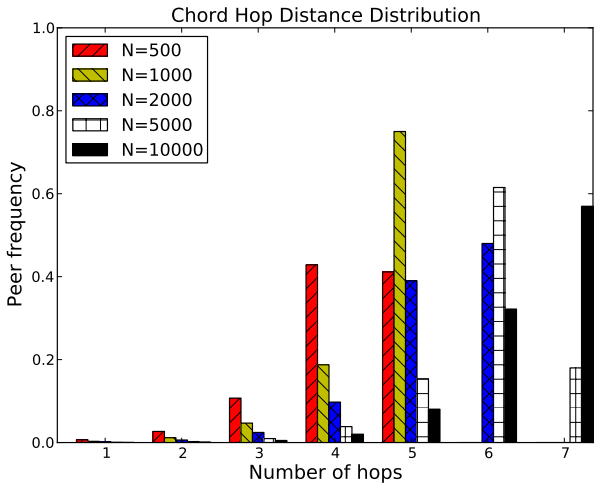

Fig. 2. Chord ODC size distribution

$$N(i) = \frac{N}{2^{b \cdot (l_{max} - (i+1))}} \quad (14)$$

To calculate the probability if an arbitrary peer of ODC i is located in v's proximity, the probability of an SDE towards v in the respective ODC needs to be known. Therefore, we define $T(i)$ as the expected number of peers which have both, an SDE towards v and an ODC i membership. The sum of probabilities of v belonging to ODC i such that not all of the SDEs to v are located in the same ODC i is multiplied by the number of SDE neighbors which are located outside of ODC i. The result is denoted as $T(i)$:

$$T(i) = \begin{cases} \frac{2}{N(i-1)} \cdot \sum_{j=1}^{\frac{\kappa}{2}} (\kappa - j) + \kappa \frac{N(i) - \kappa}{N(i)} \\ \quad \text{for } N(i) \geq \frac{\kappa}{2} + 1, \\ N(i) \quad \text{otherwise.} \end{cases} \quad (15)$$

If $N(i) \leq \frac{\kappa}{2} + 1$, two conclusions follow. First, $T(i) = N(i)$, i.e., the whole ODC i is within v's proximity. Secondly, $p_{SDE}(v,i) = 1$ because all peers in ODC i will have an SDE to v. $T(i)$ is used for the definition of $p_{SDE}(v,i)$. This definition includes three different cases to deal with varying amounts of SDEs for different ODCs.

We define $p_{SDE}(v,i)$ for Pastry as follows:

$$p_{SDE}(v,i) = \begin{cases} 1 \quad \text{if } N(i-1) \leq \frac{\kappa}{2} + 1 \\ q_{SDE}(v,i) \quad \text{otherwise} \end{cases}$$
$$q_{SDE}(v,i) = \begin{cases} \frac{\kappa - T(l_{max})}{N(i) - N(i-1)} \quad \text{for } i = l_{max} \\ \frac{T(i) - T(i+1)}{N(i) - N(i-1)} \quad \text{otherwise} \end{cases} \quad (16)$$

The two cases in $q_{SDE}(v,i)$ reflect the probability that a randomly chosen peer from ODC i is one of the $T(i)$ peers.

The probability of a peer having an LDE to v increases exponentially with decreasing distance to peer v, as exponentially less peers exist with this shared prefix length.

$$p_{LDE}(v,i) = \frac{2^{b(l_{max}-i+1)}}{N} \tag{17}$$

$\overline{p}_{SDE|LDE}(v,i)$ can be computed using the probability function of step 3 in Section III. The formula for the heuristic of the overall probability of a peer being reached via an SDE follows from Equation 6.

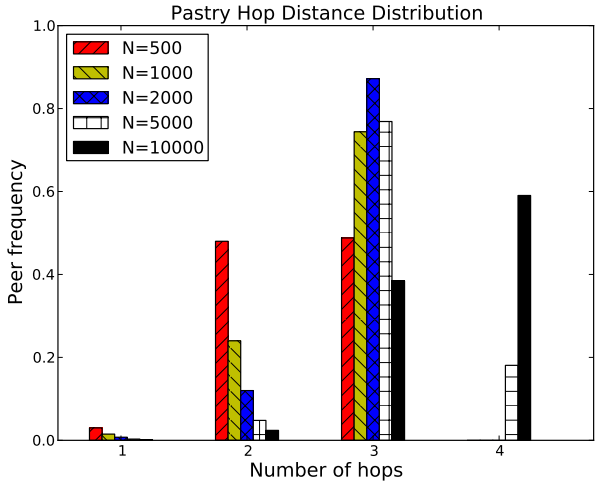

Fig. 3. Pastry ODC size distribution

C. Heuristic for Kademlia

The Kademlia protocol differs from previous protocols with respect to the following aspects. First, the routing mechanism does not differentiate between SDEs and LDEs. Despite having a proximity (see Section III-A2), on a conceptual level messages are routed via LDEs until they reach their destinations. Second, message routing is per default iterative and sends multiple parallel lookup requests until the destination IP address is resolved, the message itself is then delivered directly from the sender to the destination peer. Kademlia's ODC size distribution (i.e., values for $g(i)$) is shown in Figure 4.

Besides these aspects, the Kademlia routing mechanism is similar to Pastry's. Therefore, the Kademlia heuristic is related to the Pastry heuristic, with the differences presented below.

As Kademlia selects the most appropriate peers for the next hop from a peer set with a higher CPL than the current peer, order statistics are applied to determine the expectation $E\{Y_{max}\}$ for the highest CPL in the following way. We assume a Kademlia overlay with bucket size 8, 3 parallel routing paths and ID bit-length of 128. The peers' IDs in a bucket can be seen as 8 random instances of a bitstring X with dimension 128. For the coordinates of a string

$$X_i = (X_{i,1}, X_{i,2}, \ldots, X_{i,128}), \quad i \in \{1,2,\ldots,8\}$$

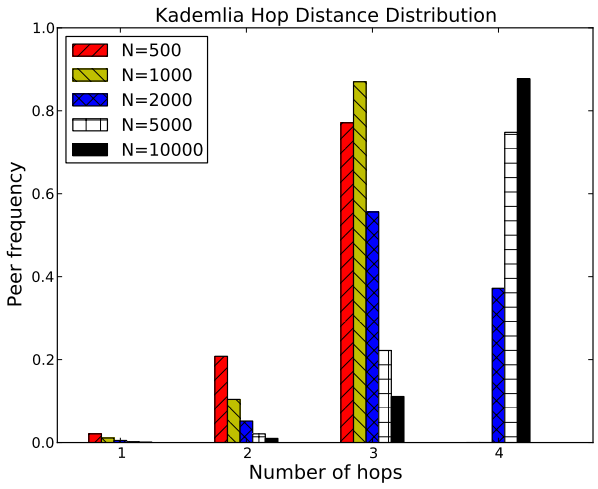

Fig. 4. Kademlia ODC size distribution

holds:

$$\mathbf{P}\left[X_{i,j}=0\right] = \frac{1}{2} = \mathbf{P}\left[X_{i,j}=1\right], \quad j \in \{1,2,\ldots,128\}.$$

We consider a random variable Y_i that denotes the CPL of a bitstring with the destination peer. Therefore, Y_i has values from $\{1,2,\ldots,128\}$ with a probability distribution:

$$\mathbf{P}\left[Y_i=k\right] = \begin{cases} \left(\frac{1}{2}\right)^{k+1} & \text{for } k \in \{1,2,\ldots,126\} \\ \left(\frac{1}{2}\right)^{128} & \text{for } k \in \{127,128\} \end{cases}$$

for all $i \in \{1,\ldots,8\}$. Let $\left(Y_{(1)},\ldots,Y_{(8)}\right)$ be the sample ordered by size, the so called order statistics. The expectation for the highest three (i.e., Y_8, Y_7, Y_6) needs to be found. For simplicity, the CPL of previous hops and one (from the bucket) is not considered in the example.

Equation 18 is based upon the Pastry heuristic, but references to b have been removed. $idLength$ represents the length of an ID, for Kademlia this is usually between 128 and 192 bit. $setSize$ stands for the number of IDs the highest CPL peers are chosen from, and max for the best peer to be considered, so $max = setSize$ yields the expectation for the peer with the maximum CPL.

$$\begin{aligned} &E\{Y_{max}\} = \\ &\textstyle\sum_{k=0}^{idLength} k \cdot \sum_{m=max}^{setSize} \binom{setSize}{m} (P[Y_{max} \le k]^m \\ &\cdot P[Y_{max} > k]^{setSize-m} - P[Y_{max} \le k-1]^m \\ &\cdot P[Y_{max} > k-1]^{setSize-m}) \end{aligned} \tag{18}$$

As the SDE concept is not anchored in Kademlia's routing algorithm, we experimentally ascertain that the closest 24 peers to a destination peer fulfill the characteristics of a

neighborhood comparable to Pastry's leaf set. We therefore define these 24 peers which are stored in the Kademlia buckets with the lowest indices as proximity. Therefore, $p_{SDE}(v,i)$ is defined as in Equation 16 with $\kappa = 24$.

We define Equation 19 to calculate the probability of an arbitrary peer's membership in ODC i.

$$g(i) = \frac{1}{2^{E\{Y_{max}\}\cdot i}} \tag{19}$$

The parallelized lookup process in Kademlia returns over the first iteration a peer set with potentially unequal CPLs.

The first iteration of $p_{LDE}(v,i)$ has to be calculated with z different values for b and with z equal to the degree of parallelism. This is because the expected CPL of the first parallel hops chosen from the sender's routing state is smaller than the CPL of the subsequent hops, as the set of peers stemming from the responses is larger. Therefore, we introduce b_0 and the formula changes accordingly to:

$$p_{LDE}(v,i) = \begin{cases} \frac{2^{b_0}}{N(i)} & \text{for } i = l_{max}, \\ \frac{2^{b(l_{max}-i+1)}}{N(l_{max})} & \text{otherwise} \end{cases} \tag{20}$$

Equation 5 needs to be calculated individually for the different values of b_0 as Kademlia takes three different paths to the target, and the average of these will represent the value.

V. Overlay Proximity LEA Strategies

This section briefly discusses two LEA strategies, namely Proximity Hijacking and Proximity Insertion, that target the overlay proximity of given victim peers. Our proposed heuristics approximate the Proximity Hijacking strategy. Thus, we deduce in the subsequent simulation case study the accuarcy of our heuristics from simulation runs according to the hijacking strategy.

A. Proximity Hijacking LEA Strategy

In the case of Proximity Hijacking no malicious peers are newly introduced into the overlay, but an attacker overtakes (i.e. hijacks) benign peers in the victim peers' proximities. An attacker could achieve this by exploiting security weaknesses to gain administrative control over the machines the peers are running on. Details on the various ways to achieve this clearly exceed the focus of this paper.

B. Proximity Insertion LEA Strategy

Using the Proximity Insertion strategy, newly joining malicious peers are inserted into the proximity of victim peers. IDs of the malicious peers are chosen to outrank the benign proximity, i.e., all malicious peers will be inserted within the range of the closest proximate peers (for symmetric proximities) or within the range of the victim and the closest proximate peer (for asymmetric proximities). Due to the self-organization mechanisms in structured P2P overlays, victim peers exchange after a given time span their benign proximities by the malicious peers. Simulations show, that the second strategy may result in a higher malicious message interception degree than the hijack strategy. We differentiate between the INSERT-low and INSERT-high Proximity Insertion strategies. INSERT-low implies that malicious peers are inserted very close to the victim peer, whereas INSERT-high relaxes this constraint a bit and requires the attacker to undercut the ID space range of the closest benign proximity peer(s) of the victim. Therefore, INSERT-high usually results in a wider range of malicious overlay IDs and INSERT-low accumulates malicious IDs nearby the victim.

VI. Performance Evaluation: Heuristics Validation & Discussion

In this section, we validate our three heuristics by evaluating a LEA simulation case study that shows an accuracy of 90% for the Proximity Hijacking strategy. The Proximity Insertion strategies reveal a higher deviation. Furthermore, we present simulation settings and performance metrics.

A. Simulation Settings and Performance Metrics

First, we give an overview on the parameters that have been chosen. Next, we present the configuration of the LEAs that have been conducted and present the measurement process. Then, we present the performance metrics to evaluate the model accuracy and the effects under malicious conditions.

1) Simulation Settings: The simulation study was conducted with Chord, Pastry and Kademlia implementations in OverSim [13]. Parameters of our simulation case studies are shown in Table I. Simulations are conducted for varying overlay network sizes between 500 and 10000 benign peers. For benign peers, we consider a uniform workload, i.e., each peer sends periodically a message to the prespecified victim peer v. Benign peers $b \in B$ and malicious peers $m \in M$ are subject to churn, and the victim peer v is present in the overlay for the whole simulation period.

2) Performance Metrics: Our simulations target to investigate two performance aspects, i.e., the accuracy of our heuristics and the degree of malicious message interception by varied LEA strategies. The model accuracy is compared to the experimental results of the hijack strategy. The attack severity is the ratio of messages that an attacker could intercept to the total number of messages that are addressed to v.

B. Attack Severity and Heuristics Accuracy vs. Overlay Size

We now investigate the impact of overlay size on the accuracy of our developed heuristics and on the severity of EA strategies. Hereby, we fix the number of malicious peers to $|M| = \kappa$, i.e., $\kappa = 8$ for Chord, $\kappa = 16$ for Pastry which reflects the default settings according to the protocols' specifications and $\kappa = 24$ for Kademlia according to our notion of Kademlia's proximity. We simulate the different strategies presented in Section V and present the EA severity evaluation in Figures 5 (Chord), 6 (Pastry) and 7 (Kademlia) for various overlay sizes. The p_{SDE} curve denotes the computation of our heuristics and the hijack curves denotes the Proximity Hijacking LEA strategy that has been presented in Section V. Deviations between p_{SDE} and the hijack curves are small, i.e., $\pm 5\%$, which yields an accuracy of our heuristics of 90%. The

Parameters	Values
Simulated time	up to 16 hours, depending on overlay size
Overlay sizes	500, 1000, 2000, 5000, 10000
% ID space (λ)	0.2%, 0.1%, 0.05%, 0.02%, 0.01%
Churn	1h lifetime with exponentially distributed probability
Chord proximity size (κ)	8
Chord average proximity width (ε)	1.6%, 0.8%, 0.4%, 0.16%, 0.08%
Chord high width inserted peer distance from v or one another	0.2%/κ, 0.1%/κ, 0.05%/κ, 0.02%/κ, 0.01%/κ
All overlays, low width inserted peer distance from v or one another	0.000001% for all sizes
Pastry proximity size (κ)	16
Pastry average proximity width (ε)	3.2%, 1.6%, 0.8%, 0.32%, 0.16%
Pastry high width inserted peer distance from v or one another	0.4%/κ, 0.2%/κ, 0.1%/κ, 0.04%/κ, 0.02%/κ
Kademlia proximity size (κ)	24
Kademlia average proximity width (ε)	5%, 2.5%, 1.25%, 0.625%, 0.3125%
Kademlia high width inserted peer distance from v or one another	0.4%/κ, 0.2%/κ, 0.1%/κ, 0.04%/κ, 0.02%/κ

TABLE I
SIMULATION PARAMETERS

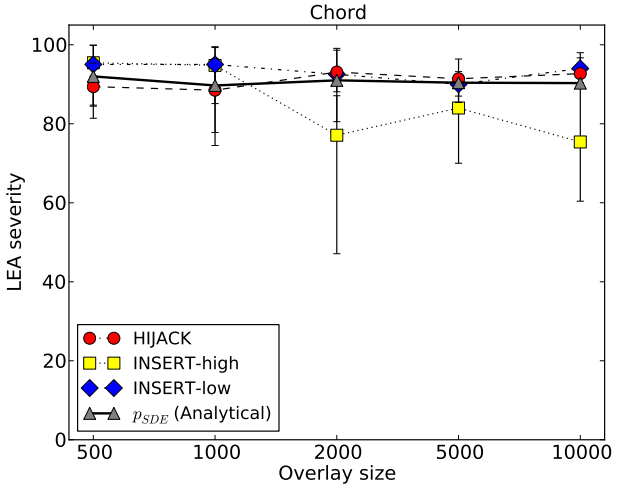

Fig. 5. LEA severity (Chord)

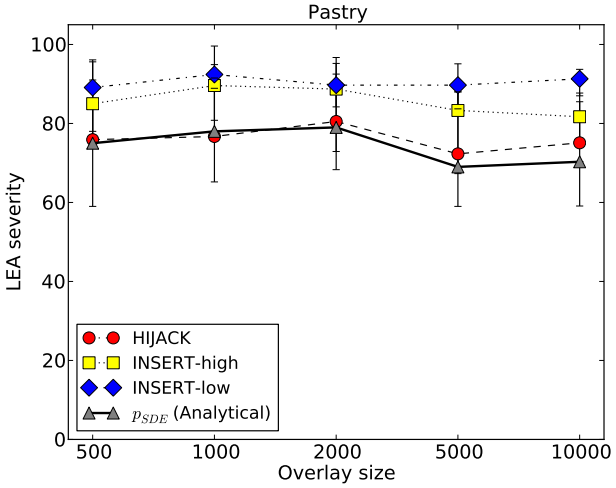

Fig. 6. LEA severity (Pastry)

p_{SDE} LEA severity is about 90% for Chord, up to 80% for Pastry, and up to 82% for Kademlia. The severity is equal to the degree of message interception, in case the victim peer's proximity has been hijacked or expelled through the newly inserted malicious peers. An important observation is that the LEA severity is almost independent of the overlay size.

Figures 5, 6 and 7 also show results for the Proximity Insertion LEA strategies. The severity of these two strategies is higher for Chord and Pastry compared to the Proximity Hijacking simulations. Kademlia displays lower susceptibility for the Proximity Insertion LEA strategy.

The difference between INSERT-low and INSERT-high is because of overlay maintenance effects that consider changes of LDEs that pointed towards v before the LEA was started.

C. Lessons Learned

We conducted a systematic analysis of structured P2P protocols and their susceptibility to LEAs. Our analysis allowed

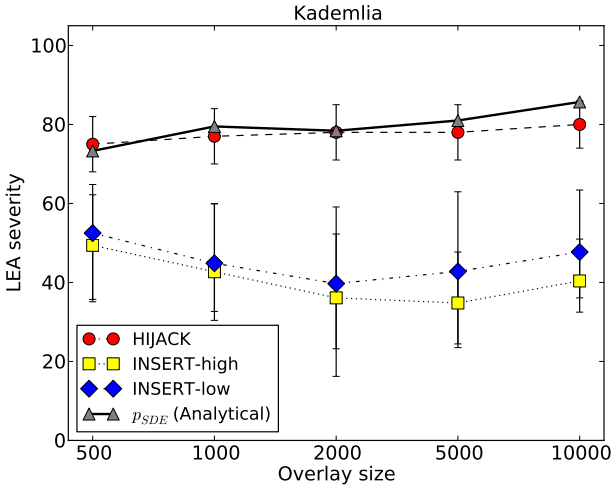

Fig. 7. LEA severity (Kademlia)

us to classify peers adjacencies according to their criticality to the incoming traffic. A key result of our work is the development of a probabilistic heuristic that serves as a template to define P2P protocol-specific heuristics to actually approximate potential malicious message interception degree from within victim peers' proximities. The heuristics show high accuracy compared to the experimental results obtained from our simulation study.

The heuristic validation has shown an accuracy of 90% compared to simulations. Both, the analytical and simulation case studies have shown that the last hop of a route significantly favors (with a probability higher than 70%) SDEs to deliver messages to their destination.

Though only representing an initial study, our three proposed LEA strategies can cause significant damage even through a small number of malicious peers and this almost independent of the overlay size. Besides covering SDEs by malicious peers, through the insertion of malicious peers (contrary to the hijacking strategy) the P2P maintenance protocols' operations result in removing some of the LDEs incident to the victim peer and thus increasing the LEA's severity for Chord and Pastry. By this way, the maintenance protocols of Chord and Pastry favor LEAs through an additional message control gain. This results in message interception of more than 90% for Chord and Pastry. Contrary, in Kademlia the proximity insertion strategies achieve a message interception of up to 50% which is clearly below the proximity hijack strategy's severity. One possible reason is the convergence of routing paths towards frequently addressed peers and the synthetic design of our simulation workload. We will address this in more detail in future work.

Overall, we identified four concepts of structured P2P protocols that are required to conduct the proposed LEAs. While these concepts leverage scalability, decentralization and self-organization, they also enable LEA susceptibility for multiple P2P protocols that entail these concepts.

D. Limitations and Future Work

A prerequisite for the presented LEA scenarios is the ability of the attackers to obtain the desired overlay IDs for malicious peers to perform a targeted insertion and hijacking activities. Depending on the protocol's mapping mechanism from an external identifier to the overlay ID, this might require a large resource pool for an attacker to choose from. However, the continuous increase in number and size of active botnets in the Internet weakens this assumption. In ongoing work, we are investigating metrics to quantify the required efforts to acquire certain amount of appropriate peers as a preparation of LEAs. Such metrics are necessary to accurately understand the efficiency and efficacy of LEAs.

While our underlying basic workload model is valid for some applications, it can also get outdated for the emerging P2P applications such as content streaming and social networks. Accordingly, we plan to extend the current analysis to scope various workload models and to consider the outgoing traffic of the victim peers.

Throughout the paper, we have emphasized the generalized applicability of our approach. We detailed our analysis for three case studies on Chord, Pastry and Kademlia and qualitatively discussed how to extend the results to other structured protocols. In future work, we aim at expanding our investigations of LEAs to both unstructured and hybrid P2P protocols.

VII. Related Work

A generic overview on various attack categories against structured P2P protocols is given in [14] and mainly focuses on Sybil, Eclipse, and routing & storage attacks.

Generic (or non-localized) EA discussions are given in [11], [15], [3], [1], [16], [17]. The remainder of this section presents for space limitation reasons only LEAs.

LEAs have been discussed in [4], [5], [6], [7], [8], [9]. Except for [4], all previously mentioned work discusses LEAs in the context of the Kad file sharing network. Kad is based on the Kademlia protocol [10], yet it is adapted to the file sharing use case in terms of overlay ID handling and replication mechanisms.

The focus on Kad limits LEA discussions to a specific file sharing application context. In our analysis, we introduce a proximity concept and substantiate the importance of the proximity based on heuristics that reflect the overlay routing algorithms of three different protocols. Thereby, we outline their differences but also the commonalities of various structured P2P protocols that allow for a unified abstract consideration of the overlay proximity notion.

In the following, we briefly present the existing works that discuss LEAs.

In [4], the possibility of LEAs are discussed on a conceptual level. The authors propose to mount the attack by occupying a specific region in the routing tables of a small set of victim peers. The authors claim that attacking peers need to have a low network latency to the victim peers and thereby assume a proximity neighbor selection (PNS) scheme in the P2P protocol. PNS refers to underlay network metrics that help to constitute a low latency proximity. Yet, for the studied LEAs in this work, we do not consider this extension.

In [5], two LEA variants are presented which are directed against the Kad file sharing network, i.e., localized in- and out-attacks. The data centric out-attack is conducted through peer placement in Kad's tolerance area of the file's ID that should be eclipsed. Peers with long-lasting overlay presence time have a higher chance to be part of other peers' routing tables. Based on this observation, they show that a LEA's effectiveness can increase over time. The out-attack poisons the victim peer's routing table by introducing many malicious peers close to the victim peer's ID.

In [6], the authors consider a Sybil attack as a preparatory step for launching a LEA. This LEA focusses on the interception of Kad lookup messages. 24 malicious peers are used to eclipse specific keywords. To achieve this, malicious peers are required to share a CPL of 96 bits with the victim peer. Intercepting Kad lookups is the same strategy as we propose

it in our Kademlia simulation study to validate our proposed Kademlia heuristic.

In [7], in- and out- LEAs are conducted. Thus, a Sybil attack is launched and malicious peers are placed randomly across the overlay. The authors introduce an attack mechanism called "back pointer hijacking" which poisons the routing tables of selected peers with the goal to evict the routing table entry of a victim peer. The mechanism exploits a security weakness in older Kad versions, recent versions of Kad prohibit this LEA variant. The LEA variants studied in our work do not consider malicious peers that propagate forged routing state information.

In [8], a LEA is conducted to place Sybil peers around a specific Kad ID. As a result, routing tables of benign peers with a common prefix length of 8 of the eclipsed ID get poisoned. Through experiments, it turned out that 8 malicious peers suffice to intercept all search requests for the eclipsed Kad ID. We assume that the different results in [8] and our study are due to the differences of Kad and Kademlia.

In [9], two LEA variants are conducted against the Kad network. Both aim at eclipsing a specific ID. The first attack prevents the ID's publication by using 10 malicious peers closely located to the ID to be eclipsed. "Close" means in this case with a common prefix length of 125 to the Kad ID of the data item. This is expectedly closer to any other peer, so the Sybil peers constitute a forged tolerance zone for the attacked Kad ID and subsequently the data corresponding to this Kad ID is replicated at the Sybils only. This LEA is a refinement of [8] and addresses a countermeasure introduced in more recent Kad protocol versions. This countermeasure delimits each peer's routing state to have not more than 10 peers from the same IP subnet or more than one peer with the same IP address in its routing table. The second attack presented in [9] prevents peers from finding replicated data by placing peers closer to the target Kad ID and propagating these peers in the area around the ID to be eclipsed.

Furthermore, without explicitly refering to LEAs, simple LEA mitigation techniques are presented in [18], [15], [1]. The authors propose to use a cryptographic hash function to compute the overlay ID, e.g., using the peer's IP address. As a result of the computation, the expected distribution of peer IDs throughout the ID space is uniform, and selection of specific IDs for a LEA requires a very high amount of attacker resources. The previous consideration is discussed in detail in [19]. The authors show how to obtain an overlay ID in a desired ID space region even with a considerable small set of external identifiers.

VIII. Conclusion

Our work has shown that the key feature of structured P2P protocols, i.e., maintaining a structured overlay topology, is at the same time a key security weakness that facilitates LEAs. Indeed, while a structured approach enables scalability and fault tolerance it also favors LEAs against specific peers. Though LEAs have been discussed in previous work, our work is the first to provide heuristics that approximate the potential severity of LEAs. These insights are crucial not only to conduct efficient and severe LEAs but also to address this weakness inherent in this class of protocols by developing requisite countermeasures.

References

[1] T. Condie et al., "Induced Churn as Shelter from Routing-Table Poisoning," in *Proc. NDSS*, 2006, pp. 115–120.

[2] J. Douceur, "The Sybil Attack," in *Proc. IPTPS*, 2002, pp. 251–260.

[3] A. Singh et al., "Defending against Eclipse Attacks on Overlay Networks," in *Proc. SIGOPS*, 2004, pp. 115–120.

[4] ——, "Eclipse Attacks on Overlay Networks: Threats and Defenses," in *Proc. INFOCOM*, 2006, pp. 1–12.

[5] T. Locher et al., "Poisoning the Kad network," *Distributed Computing*, pp. 195–206, 2010.

[6] T. Cholez et al., "Evaluation of Sybil Attacks Protection Schemes in Kad," *Scalability of Networks and Services*, vol. 5637, pp. 70–82, 2009.

[7] P. Wang et al., "Attacking the Kad Network," *SecureComm '08*, pp. 1–10, 2008.

[8] M. Steiner et al., "Exploiting KAD : Possible Uses and Misuses," *Computer Communication Review*, vol. 37, no. 5, pp. 65–69, 2007.

[9] M. Kohnen et al., "Conducting and Optimizing Eclipse Attacks in the Kad Peer-to-Peer Network," in *Lecture Notes in Computer Science*. Springer Berlin / Heidelberg, 2009, vol. 5550, pp. 104–116.

[10] P. Maymounkov and D. Mazières, "Kademlia: A Peer-to-Peer Information System Based on the XOR Metric," in *Proc. IPTPS*, 2002, pp. 53 – 65.

[11] I. Stoica et al., "Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications," in *Proc. SIGCOMM*, 2001, pp. 149 – 160.

[12] A. Rowstron and P. Druschel, "Pastry: Scalable, Decentralized Object Location, and Routing for Large-Scale Peer-to-Peer Systems," in *Middleware*, 2001, pp. 329–350.

[13] I. Baumgart et al., "OverSim: A Flexible Overlay Network Simulation Framework," in *Proc. GI at INFOCOM*, 2007, pp. 79 – 84.

[14] G. Urdaneta et al., "A Survey of DHT Security Techniques," *ACM Computing Surveys*, pp. 8:1–8:49, 2011.

[15] M. Castro et al., "Secure Routing for Structured Peer-to-Peer Overlay Networks," *SIGOPS Oper. Syst. Rev.*, pp. 299–314, 2002.

[16] K. Hildrum and J. Kubiatowicz, "Asymptotically Efficient Approaches to Fault-Tolerance in Peer-to-Peer Networks," in *Proc. DISC*, 2003, pp. 321–336.

[17] M. Srivatsa and L. Liu, "Vulnerabilities and Security Threats in Structured Overlay Networks: A Quantitative Analysis," in *Proc. ACSAC*, 2004, pp. 252–261.

[18] E. Sit and R. Morris, "Security Considerations for Peer-to-Peer Distributed Hash Tables," in *Proc. IPTPS*, 2002, pp. 261–269.

[19] D. Cerri et al., "ID Mapping Attacks in P2P Networks," in *Global Telecommunications Conference (GLOBECOM)*, 2005, pp. 6–12.