PASS: An Address Space Slicing Framework for P2P Eclipse Attack Mitigation

Daniel Germanus*†, Hatem Ismail*, Neeraj Suri*
* DEEDS Group, CS Department, TU Darmstadt, Germany
{germanus,hayman,suri}@cs.tu-darmstadt.de
† ENX Association, Frankfurt am Main, Germany
daniel.germanus@enx.com

***Abstract*—The decentralized design of Peer-to-Peer (P2P) protocols inherently provides for fault tolerance to non-malicious faults. However, the base P2P scalability and decentralization requirements often result in design choices that negatively impact their robustness to varied security threats. A prominent vulnerability are Eclipse attacks that aim at information hiding and consequently perturb a P2P overlay's reliable service delivery. Divergent lookups constitute an advocated mitigation technique but are size-limited to overlay networks with tens of thousands of peers. In this work, building upon divergent lookups, we propose a novel and scalable P2P address space slicing strategy (PASS) to efficiently mitigate attacks in overlays that host hundreds of thousands of peers. Moreover, we integrate and evaluate diversely designed lookup variants to assess their network overhead and mitigation rates. The proposed PASS approach shows mitigation rates reaching up to 100%.**

***Index Terms*—Peer-to-Peer Networks, Distributed Hash Table, Security, Localized Eclipse Attack, Mitigation, Lookup**

I. Introduction

Peer-to-peer (P2P) networks are increasingly used to facilitate collaborative computing across distributed peer nodes where a virtual overlay topology, independent of physical network topology, defines the inter-node interactions for peer communication at the application level for information aggregation/dispersal, retrievals, node discovery etc. While the overlay utilizes the physical network for the actual data transfer, the nature of the overlay as either *unstructured* (dynamic/random association across nodes e.g. Gnutella, Kazaa) or *structured* (utilizing defined search/linkage topologies e.g, BitTorrent, Kad, Kademlia) defines the type of P2P. Structured P2P protocols are often favored as the choice of a structured topology helps provide efficient searches, retrievals or associative computing. Such structured P2P protocols form the focus of our research.

Unfortunately, the structured overlay topology that provides efficiency also makes them susceptible to topology-based attacks notably Eclipse attacks (EA) [1], which represent a class of attacks where a group of attacker nodes surround a good node and 'eclipse' its ability to see the correct state of the system. While P2P protocols design choices outstandingly support resilience against *benign* faults, this is unfortunately not quite true in the presence of *malicious* attacks, such as EAs. In this work we focus on topology-aware localized EAs (taLEAs) [1], [2], [3], [4], in which adversaries target a small set of peers and require only a small amount of malicious resources compared to the overlay network size. To launch such a taLEA, an adversary may exploit a basic P2P protocol design weakness in the overlay lookup mechanisms. Lookups are required to resolve peers' network addresses; this is mandatory for direct communication between peers. The weakness stems from the deterministic behavior of lookups which reduce the distance towards the destination peer on each step. An adversary simply has to place malicious resources in the destination's proximity and subsequently is able to intercept a high percentage of lookup calls. Finally, lookup messages intercepting malicious peers e.g. lie about the correct address of the benign destination and detour lookup initiating peers to a malicious one. taLEAs exist in the real world and have been found in the open accessible KAD filesharing network [5].

The **contributions** in this work include a novel technique of *P2P address space slicing* (PASS) which, building upon *divergent lookup* [4] techniques significantly improves efficiency and scalability of P2P protocol resilience to EAs. Additionally, we propose a framework that integrates PASS with contemporary lookup variants to address a wide range of application requirements including timeliness, reliability, and scalability. The framework helps establish foundations for expanded intrusion detection and intruder removal functionality for a wide class of P2P protocols and attacks.

As validation for the proposed framework, our case studies address OverSim's [6] Kademlia [7] and R/Kademlia [8] implementations to make the case for PASS using the widespread iterative and recursive lookup variants. Besides that, we also provide experimental results for the taLEA's severity and show PASS's ability to mitigate between 90% - 100% of the encountered attack cases.

A. Paper Structure

Section II introduces the background on P2P protocol vulnerabilities, the taLEA and our mitigation approach followed by a discussion on related work. Subsequently, Section III

Research partially funded by TUD CASED/EC-SPRIDE and DFG GRK 1362.

presents the system model and the essential concepts used throughout the paper. The analytical insights that lead to the design requirements for the PASS approach are outlined in Section IV. The architecture and algorithmic realization of our framework are presented in Section V. The validation of our approach, and the results discussion, appears in Section VI.

II. Background & Related Work

We now outline the main problem dimensions to provide context for the subsequent sections. Alongside each dimension, we also highlight our related approach leading to a discussion on related work in the problem area.

A. taLEA Threat

This article's focus is on taLEAs which represent a proliferating variant of the localized Eclipse Attack [1]. The threat of a taLEA to the overlay arises as a subset of peers get hidden ("eclipsed") from the majority of the peer population. Moreover, a savvy adversary could forward requests – which were originally addressed to the hidden subset – to another malicious peer to make the rest of the overlay believe there is no attack running. This way, an adversary could affect the overlay services' availability, integrity or confidentiality. Further taLEA threats include censorship and white washing. A critical detail of taLEA is the high attack severity caused by a relatively small amount of of malicious peers compared to the overall peer population.

B. taLEA Vulnerability

taLEAs exploit weaknesses in a P2P protocol's lookup mechanism to conduct the attack. Lookups are necessary for resolving addresses of unknown peers prior to direct communication. Many lookup mechanisms are subject to a specific deterministic behavior that is exploited by the adversary.

C. taLEA Mitigation

Divergent lookups are due to their non-deterministic behavior a suitable mitigation technique [4]. Yet, their efficiency is limited and probably does not scale well with an increasing amount of peers. As an extension, we propose P2P address space slicing (PASS) utilizing divergent lookups. PASS subdivides the overlay address space into slices based on lookup success and efficiency expectations. Moreover, we propose an extensible framework to support the integration of various lookup mechanisms to address individual application requirements, diverse attacks against overlay routing and lookups, and furthermore to append downstream mechanisms, such as intrusion detection systems.

D. Related Work

The popularity of P2P protocols [9] also results in a variety of threats to it and we point the reader to [10] for a high-level overview on P2P dependability, security and anonymity. A conceptual overview for the, more used and more attacked, class of structured P2P protocols appears in [11]. The article provides insights and a classification on threats and countermeasures for different types of attacks. The systematic weakness of various structured P2P protocols is presented in [3] and an analytical and experimental study is presented to underline the potentially high taLEA impact for Chord [12], Pastry [13] and Kademlia [7].

The seminal work of secure routing [14] is a powerful mitigation technique for various EA variants. While effective, it also requires crypto primitives and a certificate authority. Consequently, this requires additional infrastructure and neglects requirements such as anonymity, openness, or resource frugality. Similarly, correctness issues in P2P routing can be found in [15]. The mitigation technique proposes a robust routing primitive which addresses generic EAs but is limited in handling localized EA's. These are introduced in [1] and this work primarily focuses on generic EAs and proposes degree bounding as a mitigation technique. The authors themselves state that the degree bounding approach is unsuitable for mitigating localized attacks.

Sybil attacks [16] discuss the case of attackers introducing a set of potentially maliciously colluding peers in order to negatively affect the service of benign peers in the overlay. These can be used as a preparatory step for localized EAs, as proposed in [2] against the KAD file sharing network whose implementation is based on Kademlia. Their mitigation approach demands a certificate authority to enable encrypted message exchange and authenticated overlay access. While [17] discusses the same attack scenario, the authors propose to use IP address restriction and a flooding protection scheme for mitigation. Also [18] discusses a similar localized EA variant, however their mitigation technique proposes usage of routing table constraints to prevent large quantities of resources exhibiting a certain underlay network locality as this applies to adversaries in some cases.

Backpointer hijacking to mount a localized EA against the popular KAD network is discussed in [19] and mitigated by applying routing table policies which prevent different network hosts from being mapped to the same overlay address. However the KAD spcific approach does not extend to other protocols. Progressing from localized EA to taLEA, an approach targeting the KAD file sharing network is presented in [20]. In this work, malicious peers populate KAD peers' tolerance zones, which is an overlay topology neighborhood, and the impact is measured over time, but no mitigation approach is proposed.

An address space crawler that detects non-uniform key distributions is the mitigation approach in [5]. Non-uniformity is an indicator for LEAs and the crawler shows a good detection rate. However, the approach has limited scalability and applicability for taLEA mitigation.

In [21] and [22], the authors propose overlay routing based on multiple and independent paths in order to increase the robustness. Due to the fact that these approaches rely on convergence across various paths or dimensions, the susceptibility to taLEAs remains, although with potentially higher cost.

Also, specialized lookup variants exist which mainly address fault-tolerange, e.g., [23], [24], [25]. For space limitations we do not go into a detailed discussion, yet we underline

their taLEA mitigation inability.

III. System Model

We now introduce the notations and the basic P2P overlay and protocol models behind the PASS framework.

A. P2P Overlay Model

The P2P overlay network is modeled as a directed graph $D = (P, E)$ with peers $p \in P$. Each peer maintains a routing table whose entries point to other peers $o \in P$. Whenever there is such a pointer, an edge $(p, o) \in E$ exists. We differentiate between incoming $E^-(p)$ and outgoing edges $E^+(p)$ of a peer p. Furthermore, we split the peer set P into malicious peers M, benign peers B, and victim peers V. $P = B \cup V \cup M$ and $B \cap M = \emptyset$, $B \cap V = \emptyset$, as well as $M \cap V = \emptyset$, and $N = |P|$.

B. P2P Protocol Model

A given P2P protocol is essentially characterized by five feature areas that we outline below.

1) Address space: This is the reference point of all resources managed in an overlay. The resources are the peers and the data items associated to peers; both may be mapped to the same address space. Contemporary P2P protocols use an integer scale range of $[0, 2^w - 1]$ with typically $w \in \{128, 160\}$ as an address space definition. Resources obtain an overlay key κ with length w which is mapped on the address space scale. Therefore, an external identifier E_{ID}, such as the peer's IP or MAC address, or the hash of a file for a data item, is passed as parameter to a hash function: $\kappa = h(E_{ID})$. The key κ therefore represents a unique identifier (moreover, at least with a low collision probability) and should be unmodified for the resource's lifetime.

2) Distance function: P2P protocols implement a distance function for address space traversal as it is required by routing, lookups, responsibility and replication mechanisms. Two overlay keys $\kappa_1 \neq \kappa_2$ yield a distance $dist(\kappa_1, \kappa_2) > 0$. A widely used distance metric is the common prefix length (CPL) which performs a XOR operation on both peers' key bitstrings. The result refers to the CPL distance.

3) Routing table: A routing table is a data structure managed individually by each peer. It contains tuples that relate a peer's overlay key to a tangible underlay network contact information, e.g., an IP address and port combination. Direct communication between peers imperatively demands a contact information. Usually, $c\ logN$ pointers are stored in a routing table and c is a protocol specific constant. Furthermore, the structure of routing tables relates to the address space and distance notion, i.e., peers store k contact information tuples about peers whose key is in a distance range of $[2^i, 2^{i+1})$ with $i = 0 \ldots w - 1$ and k constant. With decreasing distance between any two peers, the amount of common bits of their overlay keys increases. To resolve contact information of unknown peers, P2P protocols implement a lookup mechanism.

4) Lookup mechanism: In case a peer a has to send a message to another peer b but a has no contact information about b in its routing table, then a *lookup* call is performed to resolve b's contact information. To support scalability, lookup mechanisms in most structured P2P protocols are subject to a design best practice. Peer a chooses from its routing table peer(s) with the smallest distance to b and sends them the lookup message. This approach is iteratively or recursively repeated, until either b is resolved or a timeout occurs. We call lookup mechanisms that follow this design *convergent*. They are covered in more technical depth in Section V.

5) Proximity: The proximity of a peer p describes an address space region around this peer which is expectedly sparsely populated by other peers. The proximity size may vary, e.g., depending on key length and the number of peers in the overlay. We assume that a cryptographic hash function is used as it maps peers uniformly distributed to the address space [26]. Consequently, the average distance between peers is $\lambda = {}^{2^w}/_N$. Therefore, we consider the proximity of p with key $p.\kappa$ as the address space range $[p.\kappa - \lambda, p.\kappa + \lambda]$.

IV. Mitigation Fundamentals

Following the base concepts and outlying of the system model, we now set the fundamental elements that build up to the PASS technique. After explaining the taLEA susceptibility, we outline an address slicing example to further motivate the PASS approach.

A. taLEA Susceptibility

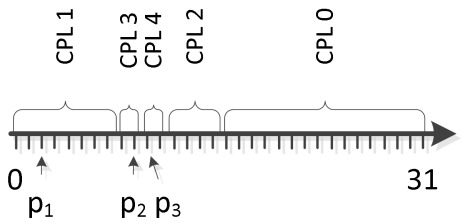
In a taLEA, malicious peers are placed closer to victim peers than any benign peer in the overlay, i.e., they populate the victim's proximity. As a consequence, malicious peers receive a significant amount of convergent lookup calls from benign peers [3] and are then able to launch further attacks such as returning bogus information or denial of service. Previous work [4] has shown that taLEAs are optimal in terms of the amount of intercepted lookup calls in Kademlia by utilizing only $|M| = 24$ malicious peers.

One way to prevent malicious peers from conducting a taLEA is to restrict lookup calls of benign peers to non-proximity address space regions of the destination peer. In that region, we expect no malicious peers or at least a significantly lower ratio of them. This leads to the hypothesis

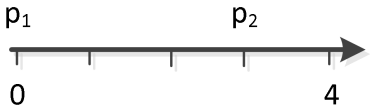
(**H**): taLEA-resilient lookups avoid the destination's proximity.

Divergent lookups represent a mitigation technique in accordance with hypothesis (H) and were validated beforehand in conjunction with a random walk non-proximity search strategy [4]. Unfortunately, random walks do not scale very well and reveal a high network overhead. To this end, we propose in this article the PASS technique which allows for efficient and scalable taLEA mitigation.

Because random walks are suboptimal in potentially very large peer populations, the PASS approach slices the non-proximity address space and then selects slices based on two



(a) Address space and CPL regions for peer p_3.



(b) CPL space from peer p_3's perspective.

Fig. 1. Example address space and CPL space.

CPL	Binary key pattern	Decimal key range for slice
0	1 _ _ _ _	[16, 31]
1	0 0 _ _ _	[0, 7]
2	0 1 1 _ _	[12, 15]
3	0 1 0 0 _	[8, 9]
4	0 1 0 1 _	[10, 11]

TABLE I
CPL SLICES EXAMPLE FOR PEER p_3 WITH KEY $\kappa_3 = 10_{10} = 01010_2$.

criteria: (i) slices should contain peers that know the peer being looked up with high probability, and (ii) peer population size of the selected slices must allow for exhaustive searching in the worst case.

B. PASS: Slicing Example

Now, we provide an example to illustrate address space slicing. We create a small address space, assign three peers to it and explain the notion of CPL (cf. Section III-B2) by showing the CPL space and the key amounts that could be mapped to the different slices.

We assume for this example a key length $w = 5$ which results in an address space range $[0, 31]$. Furthermore, we assign to three peers p_1, p_2, and p_3 the decimal keys $\kappa_1 = 2$, $\kappa_2 = 9$, and $\kappa_3 = 10$. Figure 1a shows the address space with the three peers and the CPL regions on the address space in terms of p_3. The key ranges for a given CPL slice depend on the keys' binary prefixes of length 0 through 4, as shown in Table I. In this Table, the binary key pattern is augmented with boxes which encapsulate the prefixes and underscore characters show mutable bits. CPLs 0 through 3 require a further fixed bit q next to the prefix because for $\neg\, q$ the pattern would be classified as the next higher CPL. The actual slice assignment from peer p_3's perspective is depicted in Figure 1b.

C. PASS Analysis

Building on the notion of slicing and CPL, we now develop the technical arguments supporting the proposed PASS technique.

Lemma 4.1: For two arbitrary peers, the likelihood of one peer having stored contact information about the other increases for decreasing distance.

We provide a proof under consideration of two assumptions:

(A1) The P2P protocol uses CPL-based routing, and

(A2) Peers are subject to a random application workload, i.e., they choose other peers according to a uniform distribution with a reasonable frequency.

Proof: We assume the existence of peers p, and q in an address space of size 2^w. p and q share a CPL of $w_{p,q} = w - \theta$ with w being the key length, and $\theta \in \{1, \ldots, w-1\}$. Clearly, the CPL is larger for small values of θ. Routing tables of peers consist of w lists with usually $k \in \{1, \ldots, 20\}$ entries storage capacity for contact information of peers at a distance of $[2^i, 2^{i+1})$ with $i = 0, \ldots, w-1$. Therefore, q is stored in list $i = \theta$. Hence, for a smaller CPL (larger θ) the range of potentially selectable keys for storage increases exponentially whereas the lists' storage space size k remains constant. Consequently, the likelihood of storing a peer with small values for θ is higher than for a peer with a large value for θ. ■

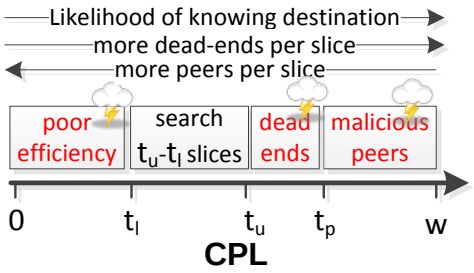


Fig. 2. Address space slicing illustration on a CPL scale.

D. PASS Design Rationale

Lemma 4.1 supports the claim that searching non-proximity slices closer towards the proximity threshold of the destination peer is more efficient.

We now provide insights on the different CPL regions, in order to motivate the PASS approach. Figure 2 depicts the CPL scale on behalf of an arbitrary destination peer. It shows three thresholds t_l, t_u, and t_p. Searching the whole range $[0, w]$ is suboptimal for the following reasons:

- The subrange $(t_p, w]$ is by taLEA assumptions populated with malicious peers and should not be searched to prevent an attack from being activated.
- The subrange $[0, t_l)$ contains a very large amount of peers which also have a low chance of knowing the destination peer (cf. Lemma 4.1). Searching this region would negatively affect the divergent lookup's efficiency.
- The subrange $(t_u, t_p]$ contains a relatively high amount of so called *dead ends*. These are peers, which know the destination neither directly nor through a transitive relation, i.e., they cannot find a path to a peer in the same slice which has contact information about the destination peer. Dead ends exist in all slices, but those close-by

the proximity threshold are at the same time sparsely populated; few peers and a high amount of dead ends would impair the divergent lookup's reliability.

As a consequence, PASS selects the range $[t_l, t_u]$ and performs divergent lookups in these slices. Clearly, a good choice for thresholds decides on both, reliability and efficiency. Naturally, the thresholds depend on P2P protocol parameters and the amount of peers. We provide experimental insights that support the threshold selection in Section VI-G.

In the next section, we outline our divergent lookup framework before detailing the divergent lookup algorithms and evaluation metrics. Subsequently, we validate our approach with a simulation case study in Section VI.

V. Resilient Lookup Framework

We now outline the proposed lookup framework. It utilizes the classical N-version-programming (NVP) design diversity paradigm [27] and enables the integration of different lookup variants to benefit from their individual advantages and mutually mitigate their disadvantages.

The overall architecture is shown in Figure 3 and consists of three different lookup variants, i.e., (i) convergent lookup, (ii) divergent lookup with PASS, and (iii) divergent lookups with random walk strategy. The convergent variant is susceptible to taLEAs but it provides low latency lookups with a low network overhead. In contrast, divergent lookups are resilient to taLEAs but yield a higher network overhead. Integrating (ii) and (iii) within one framework helps us to directly compare their reliability and overhead. In actual P2P systems, in addition to the convergent variant, a set of lookups resilient to different types of attacks is clearly preferable. Figure 3 shows that all three lookups search for κ's contact information tuples and hand over their results to the acceptance test which returns contact information and the test's decision. Assuming low-latency application requirements, the convergent lookup result could be returned to the application whereas the acceptance test is executed only after divergent lookup results are available, too. In case of failing the acceptance test, the framework could propose a backward error correction to the application. Furthermore, the test could be used to control further downstream mechanisms like intrusion detection or blacklisting of malicious peers.

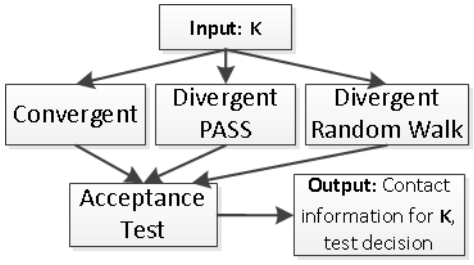


Fig. 3. Lookup framework example.

We now describe the different lookup variants using pseudocode. Therefore, we start with the convergent lookup and subsequently introduce divergent lookups with random walk and PASS strategies. Iterative divergent lookups deduplicate collected results and keep track of which peers have been queried to avoid duplicate lookup calls. For improved readability and space limitations, the deduplication mechanism is not reflected by the pseudocode algorithms.

Algorithm 1 Convergent Lookup (iterative)

```
1:  procedure LOOKUP(κ, rt, α, i_max)
2:      Q ← rt
3:      for i = 1 to i_max do
4:          C ← query ≤ α closest peers in Q for κ
5:          Q ← Q ∪ C
6:          if C contains c with c.κ == κ then
7:              return c.IPaddress
8:          end if
9:      end for
10:     return NOTFOUND
11: end procedure
```

A. Convergent Lookup

We provide pseudocode for the convergent lookup in Algorithm 1 and discuss its parameters and behavior. Four parameters are provided, the key κ of the peer to be looked up, the routing table rt of the lookup initiator, α denotes the degree of spatial redundancy, and i_{max} defines the maximum amount of lookup iterations in order to limit the network overhead and to keep a lookup from accidentally running forever. The convergent behavior can be observed in line 4 of the algorithm where at most α closest peers to the destination are picked from the queue Q and queried using the *query* primitive.

Usually, convergent lookups show good performance with $O(\log\ N)$ steps. One of their major drawbacks is the taLEA susceptibility.

Algorithm 2 Divergent Lookup (iterative) - Random Walk

```
1:  procedure LOOKUP(κ, rt, α, i_max, t_p)
2:      Q ← rt
3:      for i = 1 to i_max do
4:          Q ← Q \{sharing CPL > t_p with κ}
5:          C ← query ≤ α random peers in Q for κ
6:          Q ← Q ∪ C
7:          if C contains c with c.κ == κ then
8:              return c.IPaddress
9:          end if
10:     end for
11:     return NOTFOUND
12: end procedure
```

B. Divergent Lookup - Random Walk

We now detail the divergent lookup with random walk strategy (divRW), the respective pseudocode is presented in

Algorithm 2 which considers hypothesis (H) as a template for its implementation (cf. Section IV). The noteworthy differences to the convergent variant can be seen in lines 4 and 5 respectively. Firstly, in line 4, peers from κ's proximity, i.e., those sharing higher CPL than t_p with κ, are removed to avoid malicious peers in Q. Moreover, in line 5 peers get selected in a *random* fashion instead of selecting closest ones for the next iteration.

While divRW is resilient to taLEAs, the strategy's efficiency degrades with increasing overlay network size compared to convergent lookups.

Algorithm 3 Divergent Lookup (iterative) - PASS

```
1:  procedure LOOKUP(κ, rt, α, i_max, t_l, t_u)
2:      j ← 1
3:      Q ← rt \{sharing t_l > CPL > t_u with κ}
4:      while |Q| == 0 and j ≤ t_l do
5:          Q ← rt \{sharing (t_l − j) > CPL > t_u with κ}
6:          j ← j + 1
7:      end while
8:      if |Q| > 0 then
9:          for i = 1 to i_max do
10:             C ← query ≤ α random peers in Q for κ
11:             Q ← Q ∪ C
12:             Q ← Q \{sharing t_l > CPL > t_u with κ}
13:             if C contains c with c.κ == κ then
14:                 return c.IPaddress
15:             end if
16:         end for
17:     end if
18:     return NOTFOUND
19: end procedure
```

C. Divergent Lookup - PASS

An optimization of divRW is PASS for divergent lookups which we abbreviate as divPASS. PASS reduces the network overhead, enables divergent lookup scalability, and follows the hypothesis from Section IV. We only discuss the iterative lookup algorithm pseudocode due to space limitations, yet the recursive variant is validated in Section VI, too.

The algorithmic changes from divRW to divPASS include t_l and t_u as lower and upper thresholds for the CPL range to search for contact information that corresponds to κ. Because a lookup initiating peer could be of considerable CPL distance to κ, the condition of requiring the lower bound t_l is increasingly relaxed in case no suitable candidates can be found (see lines 4 through 7 in Algorithm 3). After an initial peer candidate set has been selected, divPASS searches in lines 8 through 17 in the given CPL bounds for κ.

VI. Evaluation

To complement the analytic basis behind PASS, we conducted extensive simulations to validate the reliability and efficiency of divergent lookups using PASS. Specifically, three distinct simulation series were conducted:

1) **taLEA case study** - to underline the significant taLEA impact on convergent lookups.
2) **PASS threshold selection** - to explain threshold selection and to project the applicability of PASS for varied overlay network sizes.
3) **divPASS vs. divRW** - to highlight the superiority of PASS over random walks.

Before discussing results of these three evaluations, we first introduce our simulation environment, models, metrics, and parameters.

A. Simulations Overview

Experiments are performed using the discrete event simulator OMNeT++ [28] and the OverSim [6] extension that provides P2P protocol implementations.

Each experiment is simulated for 8 hours simulation time and is repeated at least 20 times. The average metric measurements are provided along with 95% confidence intervals. The measurements are collected shortly before each experiment run's end to ensure that peers have stored entries their routing tables due to workload and churn effect exposure. Experiments using iterative and recursive lookups have been conducted by extending OverSim's Kademlia and R/Kademlia implementations correspondingly. Table II lists the most important simulation experiment parameters.

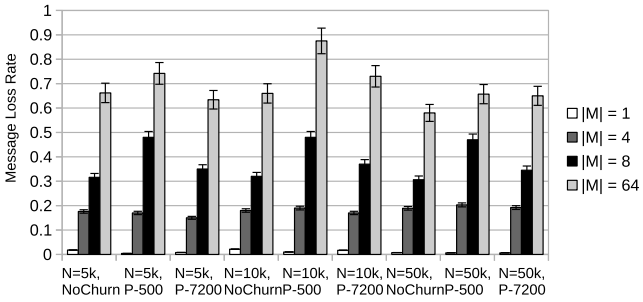
Parameter	Value	Parameter	Value
α (iterative)	10	α (recursive)	3
i_{max} (iterative)	50	TTL_{max} (recursive)	50
t_p (divRW)	80	w	128
t_l (divPASS)	4	t_u (divPASS)	6

TABLE II
SIMULATION PARAMETERS.

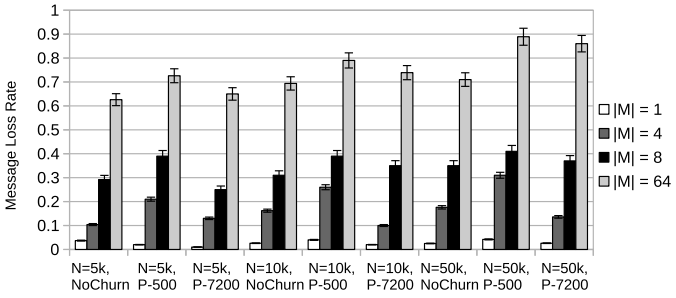
B. Simulation Churn Models

Churn denotes entering and leaving peers in the overlay. These dynamic effects are often imposed by user behavior, benign failures of peers, or failures of the underlying networking infrastructure. Churn models define an average lifetime of a peer, i.e., the amount of time units it remains on the overlay network after joining it. Moreover, an average dead time is also part of our churn models to denote the waiting period after a peer leaves before a new one is joining the overlay. Using this approach, the peer population fluctuates only slightly once the initial overlay construction is done. Peer lifetimes are modeled using the Pareto distribution [29]. Three churn models are used throughout the simulation case study:

*1) **NoChurn**:* Refers to a static peer population. Peers join at the specified rate at the beginning of the simulation and remain until the end. While victim peers $v \in V$ and malicious peers $m \in M$ are in our experiments subject to the *NoChurn* model, this does not always hold for the benign peer population $b \in B$. Benign peers can also be subject to the two Pareto models described at next.



(a) W1, average message loss



(b) W2, average message loss

Fig. 4. Convergent iterative lookup during taLEA

*2) **P-500**:* This churn model is subject to a Pareto distribution with average lifetime and dead time of 500 seconds. Given the experiment runtime, this results in approximately 50 full exchanges of the benign peer population.

*3) **P-7200**:* This is basically the same model as the previous one, but with an average lifetime and dead time of 7200 seconds which results in approximately 4 full exchanges of the benign peer population.

C. Simulation Workload Models

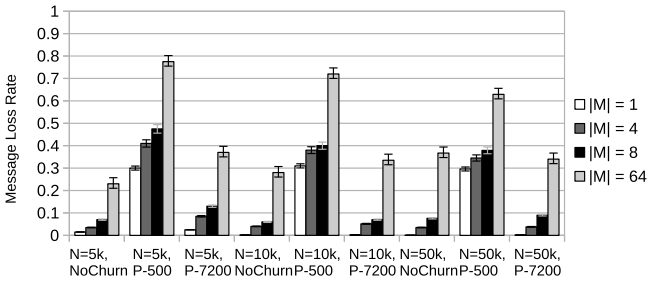
The simulation experiments are subject to two different application workload models. Peers are sending application messages on average every 10 seconds to a randomly chosen peer, the sending interval is subject to a uniform distribution with a standard deviation of 5 seconds. Thereby, peers trigger lookup calls in case the random destination peer is unknown for the originator. The two workload classes differ in how the destination peers are chosen.

*1) (**W1**) - Fully Distributed Application:* This workload class selects the destination peer uniformly distributed from $V \cup B$, as has been found in [30].

*2) (**W2**) - Service Overlay Network:* This workload class selects in 90% of the cases the destination peer uniformly distributed from V and in the remaining 10% of the cases from B.

D. Simulation taLEA Model

Throughout our taLEA experiments varying amounts of malicious peers $|M|$ are introduced into the overlay. Their lookup mechanism is modified in such a way that reply messages contain incorrect contact information of the destination peer, if the destination is a dedicated victim peer $v \in V$. This behavior is reflected in the lookup success metric which counts the aforementioned behavior as an unsuccessful lookup. Evaluation metrics are defined in the next subsection.



(a) W1, average message loss

Fig. 5. Convergent recursive lookup during taLEA

E. Evaluation Metrics

Three metrics steer our efficiency and reliability assessments for the different lookup variants:

The *lookup success rate* (**LSR**) specifies the average percentage of successful lookup calls of benign peers $b \in B$ which request contact information of victim peers $v \in V$. Moreover, we consider LSR as a taLEA resilience indicator in the divergent lookup validation process. A lookup fails once it receives contact information from a malicious peer $m \in M$ or when the maximum number of iterations i_{max} or the time-to-live TTL_{max} has exceeded and NOTFOUND is returned.

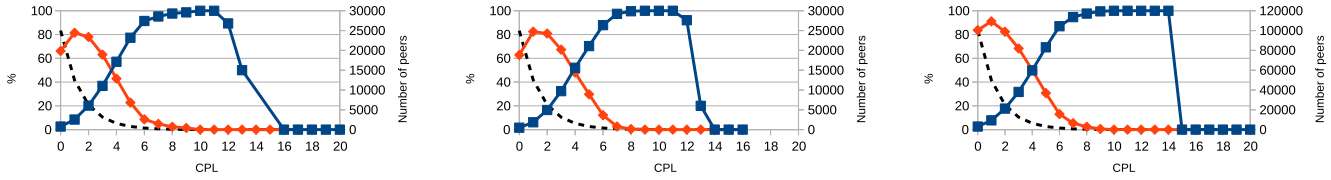
Moreover, *message complexity* (**MC**) allows to assess the network overhead of a given lookup method. It is the average number of messages sent of all successful lookup calls initiated by $b \in B$ and destined to $v \in V$.

Our third metric is *number of iterations* (**NoI**). It offers an estimate for divergent lookup latencies and provides the average amount of iterations for the first (out of up to α) successful lookup call from peers $b \in B$ destined to $v \in V$.

F. taLEA Case Study

We conduct taLEAs with varied amounts of malicious peers $|M|$ in order to assess the application message loss.

Furthermore, we assess the message overhead for the convergent lookups to compare them with the divergent variants. Experiment results for iterative convergent routing are presented in Figure 4 for $N = \{5000, 10000, 50000\}$ peers. Both workloads show a message loss of at most 89% for the P-500 churn model with $|M| = 64$, and close to 45% (W1) or 40% (W2) with $|M| = 8$. Message loss decreases for increasing peer lifetimes in both workload scenarios. MC measurements are in the range from 10.3 to 11.3 for W1 and 4.9 to 5.3 with the W2 workload (not depicted). Figure 5 presents message loss for the recursive convergent lookup in the W1 workload scenario. The average message loss for $|M| = 64$ is – when compared to the iterative variant – with about 61% to 11% lower and yields rates between 28% and



(a) $N = 50.000$, $w = 128$, $\rho \approx 1,92 \cdot 10^{-34}$ (b) $N = 50.000$, $w = 16$, $\rho \approx 0,762$ (c) $N = 200.000$, $w = 20$, $\rho \approx 0,190$

Fig. 6. Ratio of peers with non-proximity edges to destination peer (blue boxes), no-dead-end ratio (red diamonds), number of peers (dashed).

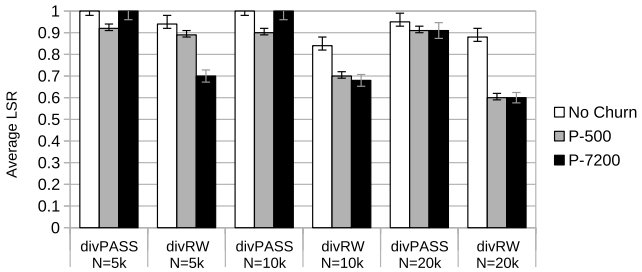
78%. MC measurements are in the range of 2.9 to 4.3 (not depicted).

Interpreting the Results: Peers in the NoChurn scenario are exposed for the longest time to the workload, therefore they have a higher chance of receiving a lookup call by one of the victim peers in which case they store or update correct victim contact information to provide this information to other benign peers in subsequently sent lookup reply messages if needed. Hence, peers with limited lifetime, i.e., P-500 or P-7200 churn model, have a lower chance to receive a lookup initiated by victim peers; more likely, peers request contact information for $v \in V$ and retrieve incorrect contact information. The provision of bogus contact information about $v \in V$ is therefore increasing with N and consequently message loss increases as well.

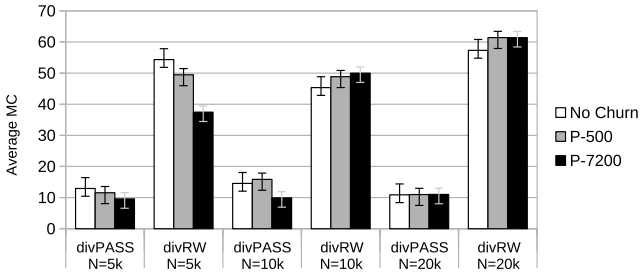
G. PASS Threshold Selection

Efficiency and reliability of divPASS depends on selecting adequate values for the lower and upper thresholds t_l and t_u. Insights about the selection process are provided through an experiment set whose results are shown in Figures 6a through 6c. These present two ratios for peers with non-proximity edges to the destination peers (blue squares) and no-dead-ends (red diamonds). The number of peers per slice (dashed line) is denoted on the right side y-axis. All x-axes show the CPL range $[0, 20]$ for visualization reasons. This set of measurements was conducted using the W1 workload and the NoChurn model for different values of N and w. This leads to different address space load factors $\rho = \frac{N}{2^w}$, which have been considered in this experiment series to compare dense to sparsely populated address spaces.

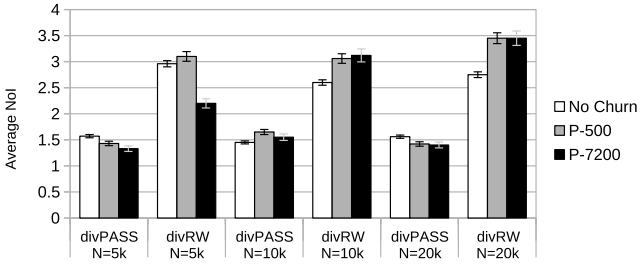
Interpreting the Results: Similar key characteristics of Figures 6a through 6c show an increase of peers with non-proximity edges ratio up to CPL 12 (or 14 for Figure 6c) whereas the no-dead-end ratio decreases. The upper threshold t_u should be chosen such that the chance of reaching a dead-end can be neglected. The selection of the lower threshold t_l must ensure that the amount of peers populated in the lookup slice range $[t_l, t_u]$ is not excessively large, such that α parallel lookups with search depth i_{max} (TTL_{max} respectively for the recursive variant) have a high probability of finding a non-proximity edge to the destination. Thus, for the set of experiments using $w = 128$ and $N = \{5000, 10000, 20000\}$ in the next subsection, we have selected $t_l = 4$ and $t_u = 6$ as divPASS parameters.



(a) Lookup success rate (LSR)



(b) Message complexity (MC)



(c) Number of iterations (NoI)

Fig. 7. Comparison of iterative divPASS vs. divRW, W1

We believe due to the measurement similarities that overlays with both, high load factors and longer key lengths, can also efficiently apply divPASS, i.e., overlays hosting millions of peers.

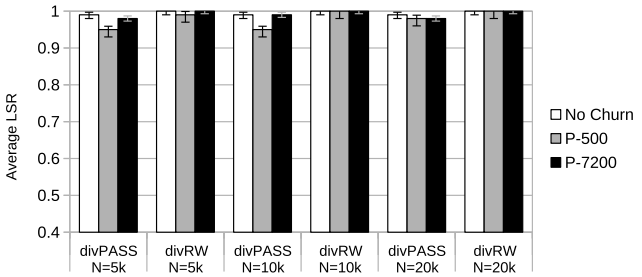
H. divPASS Reliability and Efficiency Case Study

In this subsection, divPASS is compared to divRW using our three metrics LSR, MC, and NoI (abbreviations defined in Section VI-E).

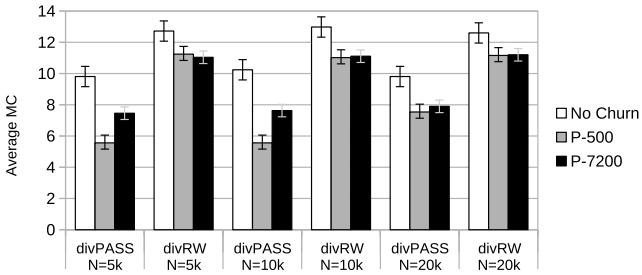
1) Iterative, W1: Figures 7a through 7c show experiments for the iterative divergent lookup comparison using workload W1. In terms of LSR, a pairwise comparison of divPASS to divRW for similar churn - N configurations always shows a better LSR measurement for divPASS; its measurements are in the range of 90% to 100% successful lookups, whereas divRW results are in the range of 60% to 90%. Figure 7b depicts MC, and shows for all divPASS configurations a significant lower overhead for the same divRW configuration. Clearly, the MC of divPASS outranks the one of divRW by factor 3 to 6 depending on the considered churn - N configuration. The iterative divPASS experiments have been conducted using $\alpha = 10$, yet the average MC measurements show 10 messages or less for some of the configurations. This happens, as PASS searches slices that reveal a high destination knowledge ratio but are sparsely populated which may result in a successful divergent lookup call after querying less than α peers. Figure 7c shows the NoI; like in the previous MC discussion, we find for NoI significantly better divPASS measurements, i.e., in the range between 1.3 to 1.6 iterations. The corresponding divRW NoI measurements are in the range between 2.2 and 3.4 iterations. Clearly, divPASS shows better results here which makes us believe that divPASS would provide better latencies than divRW for successful lookups in real networks.

2) Iterative, W2: Figures 8a through 8c show comparison experiments for the iterative divergent lookups using workload W2. Average LSR in Figure 8a shows equally good results for divPASS like in the W1 workload case, yet divRW has a higher LSR in W2 compared to W1. Overall, all LSR results are in the range between 98% to 100%. Average MC results are shown in Figure 8b. The maximum MC improvement of divPASS over divRW is about factor 2 in the W2 case. Although, the MC difference between divPASS and divRW is lower for W2 than for W1, it is important to consider the different MC scale ranges of W2 (Figure 8b) and W1 (Figure 7b). divPASS MC measurements are in the range of 5.8 to 10.2 and divRW in the range of 11 to 13. NoI measurements in Figure 8c show in the pairwise churn - N configuration comparison that divPASS measurements do not outrank those of divRW like it is the case for W1. Because in W2 more contact information among benign peers about $v \in V$ exists, divRW scores better as it can select without slice range restriction easily α peers during the first iteration with a high probability of a successful lookup. As divPASS searches a subset of slices, it may occur that during the first iteration less than α peers for the given slice range are found. If these peers from the first iteration cannot resolve $v \in V$, a second iteration is required. All measurements are close to each other in the range of 1 to 1.4 iterations.

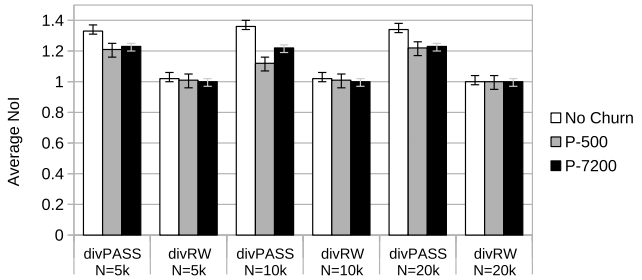
3) Recursive, W1: Figures 9a through 9c show comparison experiment results for the recursive divergent lookup variants using workload W1. LSR measurements in Figure 9a show success rates in the range of 95% to 100% and divRW shows slightly better LSR results than divPASS. On the other hand, MC for divPASS is around 10 messages while divRW is in the range of 18 to 39 messages, as can be taken from Figure 9b. NoI measurements are in the range of 1.4 to 2.5 iterations, as depicted in Figure 9c.



(a) Lookup success rate (LSR)



(b) Message complexity (MC)

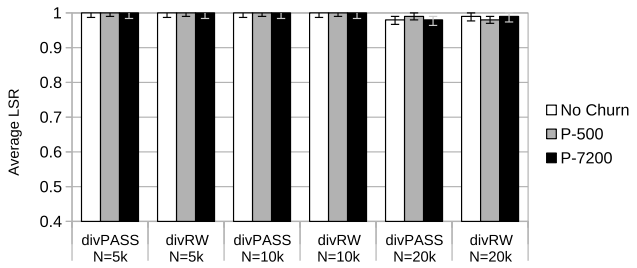


(c) Number of iterations (NoI)

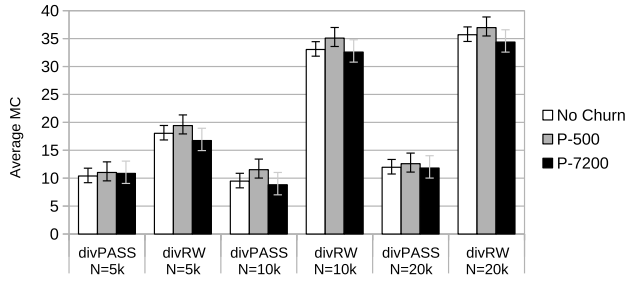
Fig. 8. Comparison of iterative divPASS vs. divRW, W2

I. Interpreting the Results

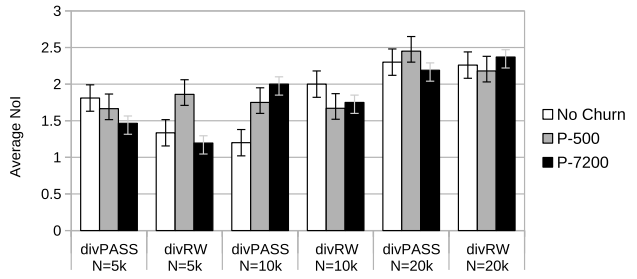
The first part of our case study shows that taLEA may cause message loss of 60% up to 89%. We propose PASS for divergent lookups to mitigate taLEAs and give insights about the ratio of peers knowing the destination as well as the no-dead-end ratio in overlay networks. Experiment data supports the claim that PASS is a suitable taLEA mitigation approach for very large overlay networks with hundreds of thousands to potentially millions of peers. The actual divPASS validation yields a worst case result of 90% LSR. Failing divPASS lookups can be ascribed to dead-ends or lack of suitable candidate peers, especially for churn models with short average peer lifetimes. The pairwise comparison of iterative divPASS to iterative divRW shows divPASS's superiority for the W1 workload model. The W2 workload model shows similar and good LSR results for both approaches. Furthermore, divPASS either significantly reduces MC or is at worst on par with divRW. Also, NoI for divPASS shows an improvement over



(a) Lookup success rate (LSR)



(b) Message complexity (MC)



(c) Number of iterations (NoI)

Fig. 9. Comparison of recursive divPASS vs. divRW, W1

divRW in case of workload W1. In case of W2, correct contact information of the victim peers is widely available in the overlay, which allows divRW to retrive it faster than divPASS which has to select an appropriate set of peers for the smaller search region of PASS.

VII. Conclusion & Future Work

We have presented a novel lookup framework for structured P2P protocols whose divergent lookups are resilient to taLEAs. Furthermore, the newly proposed PASS approach ensures for large overlays a moderate network overhead compared to divergent lookups using random walks. Our mitigation approach is decentralized, does not require additional infrastructure, and mitigates taLEAs in 90% to 100% of the cases. Consequently, the approach is suitable for a wide range of application requirements in terms of scalability, reliability, and timeliness.

Future Work

We consider the extension of our framework by an intrusion identifier mechanism, to remove malicious routing information from large portions of the benign peer population. Also, we are planning to implement our framework for a distributed testbed in order to conduct a large-scale timeliness study.

References

[1] A. Singh et al., "Eclipse Attacks on Overlay Networks: Threats and Defenses," *In Proc. INFOCOM*, pp. 1–12, 2006.

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

[3] D. Germanus et al., "Susceptibility Analysis of Structured P2P Systems to Localized Eclipse Attacks," *In Proc. SRDS*, pp. 11–20, 2012.

[4] D. Germanus et al., "Mitigating Eclipse Attacks in Peer-to-Peer Networks," *In Proc. CNS*, pp. 400–408, 2014.

[5] T. Cholez et al., "Detection and Mitigation of Localized Attacks in a widely Deployed P2P Network," *Peer-to-Peer Networking and Applications*, vol. 6, no. 2, pp. 155–174, 2013.

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

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

[8] B. Heep, "R/Kademlia: Recursive and Topology-aware Overlay Routing," *In Proc. ATNAC*, pp. 102–107, 2010.

[9] Lua et al., "A survey and comparison of peer-to-peer overlay network schemes," *Communications Surveys Tutorials, IEEE*, vol. 7, no. 2, pp. 72–93, 2005.

[10] F. DePaoli and L. Mariani, "Dependability in peer-to-peer systems," *Internet Computing, IEEE*, vol. 8, no. 4, pp. 54–61, 2004.

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

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

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

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

[15] D. Wallach, "A survey of peer-to-peer security issues," in *Software Security Theories and Systems*, ser. Lecture Notes in Computer Science, M. Okada, B. Pierce, A. Scedrov, H. Tokuda, and A. Yonezawa, Eds. Springer Berlin Heidelberg, 2003, vol. 2609, pp. 42–57.

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

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

[18] M. Kohnen et al., "Conducting and Optimizing Eclipse Attacks in the Kad Peer-to-Peer Network," *LNCS*, vol. 5550, pp. 104–116, 2009.

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

[20] T. Locher et al., "Poisoning the Kad network," *LNCS*, vol. 5935, pp. 195–206, 2010.

[21] I. Baumgart and S. Mies, "S/Kademlia: A practicable Approach towards Secure Key-based Routing," *In Proc. ICPADS*, pp. 1–8, 2007.

[22] E. Oh and J. Chen, "Parallel Routing in Hypercube Networks with Faulty Nodes," *In Proc. ICPADS*, pp. 338–345, 2001.

[23] Ricardo Villanueva et al., "Secure Routing Strategies in DHT-Based Systems," *LNCS*, vol. 6265, pp. 62–74, 2010.

[24] A. Nambiar and M. Wright, "Salsa: A Structured Approach to Large-scale Anonymity," *In Proc. CCS*, pp. 17–26, 2006.

[25] N. S. Evans and C. Grothoff, "R5n: Randomized Recursive Routing for Restricted-Route Networks," *In Proc. NSS*, pp. 316–321, 2011.

[26] Karl Aberer et al., "The Essence of P2P: A Reference Architecture for Overlay Networks," *In Proc. P2P'05*, pp. 11–20, 2005.

[27] A. Avizienis, "The N-Version Approach to Fault-Tolerant Software," *IEEE Transactions on Software Engineering*, vol. SE-11, no. 12, pp. 1491–1501, 1985.

[28] G. Pongor, "OMNeT: Objective Modular Network Testbed," in *In Proc. MASCOTS*, 1993.

[29] Zhongmei Yao et al., "Modeling Heterogeneous User Churn and Local Resilience of Unstructured P2P Networks," *In Proc. ICNP*, pp. 32–41, 2006.

[30] Krishna P. Gummadi et al., "Measurement, Modeling, and Analysis of a Peer-to-peer File-sharing Workload," in *SOSP '03*. New York, NY, USA: ACM, 2003, pp. 314–329.