A Composite Malicious Peer Eviction Mechanism for Super-P2P Systems

Hatem Ismail
DEEDS Group, TU Darmstadt
hayman@deeds.informatik.tu-darmstadt.de

Stefanie Roos
University of Waterloo
stefanie.roos@uwaterloo.ca

Neeraj Suri
DEEDS Group, TU Darmstadt
suri@cs.tu-darmstadt.de

Abstract—**Large-scale P2P applications (e.g., social networking, online gaming, video streaming) that host millions of users increasingly rely upon semi-structured super-P2P systems to provide efficient services in dynamic environments. Given the critical role of 'super peers' in such topologies, attackers target super peers due to the resultant high damage on P2P services.**

In this paper, we consider the prominent class of Outgoing Eclipse Attacks (OEA) where an attacker aims to block the communication by controlling all the outgoing connections of honest super peers. Our interest on OEA stems from the fact that our simulation studies reveal that OEAs can cause up to 90% of all service requests to fail.

Our attack mitigation relies upon a novel (a) monitoring and (b) malicious peer eviction scheme based on a composite proactive and reactive mechanism. Our proactive mechanism enforces an upper bound on the number of connections an attacker can establish, whereas our reactive mechanism expels malicious peers from the overlay using a distributed consensus protocol. We show that our protection mechanism is highly effective and exhibits a low false-positive rate. Our extensive simulation study validates the analytical results over a large range of parameters with observed detection accuracies of 99% and throughput enhancements of up to 100% while entailing an overhead of less than 5%.

Index Terms—**P2P, Eclipse Attacks, Auditing, Super-P2P.**

I. Introduction

Large-scale P2P systems for video streaming, VoIP, Massively Multiplayer Online Games (MMOG) and other popular applications host millions of users [1]–[3]. On this scale, despite their ease of maintenance, unstructured P2P systems are often inefficient as they rely on flooding or random walks to disseminate requests. Hence, the performance penalty grows rapidly with the number of nodes [4]. In contrast, structured P2P systems require considerable maintenance overhead to adapt to such scale and are vulnerable to a number of denial-of-service attacks [4], [5]. Therefore, system developers increasingly rely on semi-structured *super-P2P systems* to organize service provision in P2P systems and form the foundation of large-scale applications [6].

In a super-P2P system (or overlay), *super peers* disseminate messages on behalf of regular peers, which then only request or provide a service without relaying traffic. Typically, super peers exhibit an above-average performance with regard to indicators such as bandwidth or reliability, thus improving the service quality in contrast to random peers. In addition, the hierarchical nature of a super-P2P system increases their scalability and efficiency compared to unstructured overlays as peers flood messages only between super peers. Based on a simple hierarchical network structure, super-P2P systems entail minimal topology maintenance and load balancing issues while providing inherent protection against certain denial-of-service attacks [6], [7].

However, due to their distinguished nature, super peers present tempting targets for attackers. Disabling a large fraction of super peers, which corresponds to a low fraction of all peers, effectively undermines the service provision of the overlay. The class of Outgoing Eclipse Attacks (OEAs) are particularly threatening for super-P2P systems. When launching an OEA, an attacker infiltrates the routing tables of super peers such that all or most relayed messages of these peers end up at malicious peers. These malicious peers then perform a denial-of-service attack by either dropping the requests or faking replies. Indeed, a simulation study based on real-world super-P2P systems highlighted that 0.1% of malicious peers incur a performance degradation of up to 40% [8], indicating that denial-of-service (DoS) attacks might cause detrimental damage to the aforementioned large-scale applications that rely heavily on super-P2P systems.

Eclipse attacks and their countermeasures constitute an active area of research. However, the existing work is primarily concerned with eclipsing incoming messages in structured overlays. Therefore, most of the proposed countermeasures are not suitable in the context of super-P2P systems as they (i) rely on the routing scheme of structured overlays [9], [10], (ii) are limited to specific adversarial behavior such as localized or topology-aware attacks [11], (iii) are inefficient in terms of communication or computation overhead [12], or (iv) require central parties [13], [14].

In this paper, we propose an effective detection and peer eviction mechanism to mitigate OEAs in super-P2P systems. Our composite proactive and reactive mechanism mitigates the effect of routing table infiltration as well as the subsequent denial-of-service attack. Our proactive scheme aims to reduce the number of appearances of malicious peers in routing tables. For the proactive mechanism, we modify a previous auditing scheme proposed [15] that enforces an upper bound on the number of connections that peers can establish. We improve the existing protocol by reducing negative impacts on honest peers that could lead to the accidental eviction of these peers in the original scheme. we provide an in-depth theoretical analysis of our modified algorithms, which provides bounds on the likelihood to recognize malicious peers within a certain time frame. Complementing our anonymous auditing scheme, our novel reactive mechanism detects denial-of-service attacks. Peers assign trust values to their neighbors and collectively blacklist peers with low trust values using a distributed consensus protocol, resulting in a permanent eviction of these peers from the system.

The evaluation of the proposed mechanism is twofold. In the theoretical evaluation, we leverage a probabilistic model to ascertain that our proactive mechanism detects malicious infiltration with high probability while erroneous removals of honest peers are rare. Combining the resulting upper bound on the number of malicious connections with a combinatorial argument about the reactive mechanism, we provide tight bounds on the number of honest peers that an attacker can eclipse. Our extensive simulation study validates the theoretical bounds. Furthermore, our experimental results indicate that our composite detection mechanism significantly increases both lookup success ratios and throughput while only imposing overheads as low as 5%. We thus effectively mitigate a serious threat to super-P2P networks.

II. System Model

In this section, we present the system and adversary model. We first introduce the notation regarding the different parties in a super-P2P overlay, followed by an overview of neighbor selection and communication protocols. Afterwards, we describe our adversary model.

A. Overlay Model

We model a P2P overlay as a directed graph $D = (P, E)$ with the set P of peers and the set $E \subset P \times P$ connections between those peers. More precisely, each peer $p_i \in P$ maintains a *routing table* RT_i containing the contact information of other peers. We write $(p_i, p_j) \in E$ if RT_i contains the contact information of peer $p_j \in P$.

Peers exchange messages to serve requests and maintain the overlay. When a peer p_i sends a request to a destination peer p_d, peers forward the message to p_d using connections $e \in E$. A *lookup mechanism* determines which connections are chosen to forward the message.

In this paper, we focus on super-P2P networks; i.e., we divide the set of peers $P = S \cup R$ into two subsets of *super peers* S and *regular peers* R. We assume that super peers evolve from regular peers, where a regular peer is promoted to a super peer if it exhibits promising characteristics such as high bandwidth, CPU capabilities, reliability and storage limit.

Various super peers selection strategies have been proposed [16], though the selection strategies *per se* are not the focus here. A regular peer p_i obtains the contact information of one super peer sp_i, e.g., from a web-server or via an already known peer in the network. A super peer sp_i maintains connections to all regular peers p_i that are connected to it. We call the set of sp_i and its connected regular peers a *cluster*. In addition, sp_i maintains connections to super peers sp_j in order to serve requests from both regular and super peers.

As message delivery in super-P2P systems relies primarily on super peers, regular peers forward their *lookup requests* to super peers for dissemination. Consequently, super peers forward the lookup requests to other super peers until the destination address is resolved. Otherwise, the lookup fails.

B. Attack Model

We now describe the OEA model used for evaluating the impact of eclipsing super peers' outgoing messages. We start by stating the attacker's goal and motivation, followed by representative high-level attack strategies.

For P2P systems, we are primarily concerned with an internal attacker that infiltrates the system by inserting malicious peers. Hence, we divide the set of peers into benign and malicious peers B and M, i.e., $P = B \cup M$.

a) Attack Goals and Motivation: We consider an internal and active adversary. The adversary aims to execute a denial-of-service attack and minimize the number of successful message deliveries. For that purpose, an attacker executes an Outgoing Eclipse Attack (OEA) on a set of victims $V \subseteq B$ with the goal of capturing all their outgoing messages. OEA is a variant of EAs: malicious peers only target outgoing messages whereas typical EAs target ingoing messages to victims. Hence, we highlight the feasibility of launching OEA as intercepting messages is a well deployed adversarial strategy. As regular peers have little to no impact on the success of message deliveries[1], we focus on malicious super peers. The attacker might either target all super peers equally, referred to as passive OEA, or focus on a specific victim set, denoted as active OEA.

In a passive OEA, the attacker has no specific set of victim super peers, therefore $V = S \cap B$. Rather, the attacker targets capturing as many messages as possible regardless of the originating peer. The reasons for launching a passive OEA include, among others, (i) a desire to degrade the overall reliability and (ii) an incapability to specifically connect to certain targeted peers.

In contrast, in an active OEA, the attacker aims to eclipse a specific subset $V \subset S \cap B$ of super peers. Complementary to the scenario of passive OEAs, the reasons for launching an active OEA include (i) a desire to prevent certain users from receiving services and (ii) the limited attacker resources to perform an attack on all peers.

b) Attacker's Capabilities and Strategy: A decisive quantity for the strength of the attacker is the malicious fraction MI of super peers it can control, i.e., $MI = \frac{|M|}{|S|}$. We assume that malicious peers collude and are aware of all malicious peers in the network. In particular, malicious peers can observe, drop and replay received messages. Furthermore, they can send, and possibly forge, messages in the absence of a valid message authentication scheme.

Based on the above capabilities, the attacker can execute a two-fold strategy to maximize the impact of a denial-of-service attack. In the first step of the attack, malicious peers aim at receiving as many messages as possible in order to maximize their impact on the system. Being in as many routing tables as possible increases the probability to receive a message.

[1]Indeed, their only option is launching a DoS attack on their super peer, which can easily be detected.

Hence, malicious peers aim to maximize their presence in routing tables by leveraging the neighbor discovery algorithm of the overlay. Usually, they execute the algorithm at a higher frequency and accept more neighbors than intended.

Regarding the second attack strategy, when receiving a lookup request, malicious peers may either forge a reply, drop the request, or pretend that a time-out occurred and the request cannot be served. As the effect of an alleged timeout on the successful delivery is identical to the effect of dropping, we focus on the first two attack strategies.

III. Background & Related Work

The purpose of this section is twofold. First, we summarize the existing work on EAs detection mechanisms in various P2P overlays and highlight their unsuitability for super-P2P systems. Second, we introduce Singh et al's anonymous auditing protocol for enforcing degree constraints [15].

A. Attack Detection Mechanisms

There are various approaches for detecting EAs in structured P2P overlays that rely upon the existence of deterministic lookups and address-based resource allocation. For instance, Young et al. integrate a quorum into each step of the deterministic lookup [17]. It is unclear how to select the quorum members for super-P2P overlays. Moreover, an unstructured lookup involves the majority of all peers in the system, such that the additional communication overhead induced by the quorum drastically reduces the efficiency.

Furthermore, there exist malicious peer detection and eviction schemes for structured overlays based on quorums [18], [19]. However, the proposed mechanisms consider only directed attacks, i.e., attacks launched against a specific set of victims based on their addresses in the structured overlay. These approaches are not applicable for semi-structured overlays.

In contrast, *OceanStore* [14] and *Rosebud* [13] offer more general protocols for malicious peer eviction in large-scale storage systems. However, both require a centralized server to handle node eviction.

Although *Commensal Cuckoo* [20] and *HQ* [21] offer fault tolerance mechanisms, they either focus on very specific attacks or fail to evict malicious peers. Similarly, Scheideler et al. [22], [23] propose a mitigation approach against join-leave and Sybil attacks based on continuous node relocation. Though their mechanisms are efficient and provide high robustness, no reactive mechanism for detection or eviction is provided.

B. Anonymous Auditing of Node Degrees

In the following, we discuss the widely known anonymous auditing scheme introduced by Singh et al. [15], that aims to limit the number of routing tables a peer exists in. Next, we highlight the key elements presented in their work that are utilized in our proactive mechanism. In the process, we identify weaknesses in the original algorithm and motivate the necessity of a new auditing scheme.

In [15], the authors aim at limiting the incoming connections of peers, i.e., the number of routing tables a peer exists in. For this purpose, they propose an anonymous auditing scheme to ensure that peers adhere to an upper bound on the in-degree.

Let the *backpointer set bp* of peer u refers to the set of peers whose routing table contain u. A peer v, the *challenger*, anonymously requests the backpointer set of each peer u in its routing table. If, for peer v, a neighbor u's backpointer set exceeds the maximal allowed size or does not contain the challenger v, u does not adhere to the protocol and v removes u from its routing table. Requesting the backpointer set anonymously is the main challenge of the algorithm: peer v has to employ an *anonymizer* w to query the *responder* u so that u cannot deduce the identity of v. We discuss the selection of anonymizer peers after elaborating on how to overcome a small fraction of malicious anonymizers.

If the responder u is malicious, a malicious anonymizer w can reveal v's identity to u so that u can include v in its reply. In contrast, if u is honest, the use of authenticated messages prevents a malicious anonymizer w from forging a reply but does not prevent w from dropping the request.

In order to account for these strategies, v executes k requests using diverse anonymizers w. If any of these requests results in a backpointer set that is too large or does not contain v, v removes u. Otherwise, v keeps u only if at least l responses contain v in the backpointer set. Under the assumption that the fraction of malicious peers is small and the peers are selected approximately uniformly, the probability to accidentally remove an honest peer or keep a malicious one is low.

Singh et al. consider several methods for selecting the anonymizer w. Their preferred method is a secure lookup for the key $H(x_u)$ with x_u denoting the address of u. In this manner, all peers v rely on the same set of anonymizers when querying for u's backpointer set so that the anonymizer selection does not reveal information about the requesting peer. If sufficiently many peers in the set $H(x_u)$ are honest, u is unable to increase its in-degree beyond the permitted bound.

Our proactive mechanism builds upon Singh et al.'s work. However, we identify two key concerns that require careful modification of the original protocol:

1) A secure lookup for specific peers in an unstructured overlay is highly inefficient.
2) The fact that $H(x_u)$ is identical for all challengers v might lead to the expulsion of honest peers from the system and allows for individual malicious peers of arbitrary degree.

In order to clarify the second aspect, consider the scenario that more than l of the closest k nodes to $H(x_u)$ can be malicious with a certain probability. If u is malicious, more than l malicious anonymizers imply that u can convince all challengers that it is honest. Thus, u could have an arbitrary high in-degree. On the other hand, if u is honest, more than $k-l$ malicious anonymizers imply that all challengers will drop their connection to u, ultimately exiling the honest peer u. In contrast, if each challenger v uses a distinct set of anonymizers for the responder u, only a small fraction of challengers with malicious anonymizers removes u. Hence, u can still participate in the system.

In summary, the work by Singh et al. provides some key ideas on how to monitor the number of connections a node can establish. Nevertheless, the algorithm in its current is unsuitable for our scenario. As a consequence, we propose a novel two-fold mechanism that specifically overcomes the aforementioned issues. Our mechanism provides (a) a method for selecting dynamic and distinct anonymizer sets and (b) a distributed consensus protocol that expels malicious peers, which is an aspect that is not considered in Singh et al. 's work.

IV. Detection Mechanisms

In this section, we present our proposed combined proactive and reactive mechanisms for detecting malicious behavior. The goal of the proactive mechanism is to prevent malicious peers from gaining a disproportional influence. Complementary, the reactive mechanism detects and expels malicious peers. Table I enumerates variables used throughout the paper.

TABLE I
Acronyms

Var.	Description	Var.	Description
Q	quorum	$SN(v)$	super neighbors of v
k	num(anonymizers)	l	necessary correct replies
θ	degree bound	bp	backpointer list
$tv_v(u)$	v's trust in u	t_{out}	reply timeout
τ	min quorum interval	$L_{r,v}$	lowest trust neighbors
δ_1	increase in trust	δ_2	decrease in trust

A. Proactive Detection

An attacker usually attempts to maximize its influence by infiltrating as many routing tables as possible. In order to mitigate the impact of the attacker, we aim to reduce the fraction of routing tables malicious peers can infiltrate.

Our mechanism keeps the key ideas of the anonymous auditing scheme by Singh et al. presented in Section III-B but i) replaces the secure deterministic lookup with a gossiping protocol, ii) bases its decision upon a sliding window of replies from individually chosen anonymizers, and (iii) does not require a structured overlay for selecting anonymizers. We now discuss these modifications in detail.

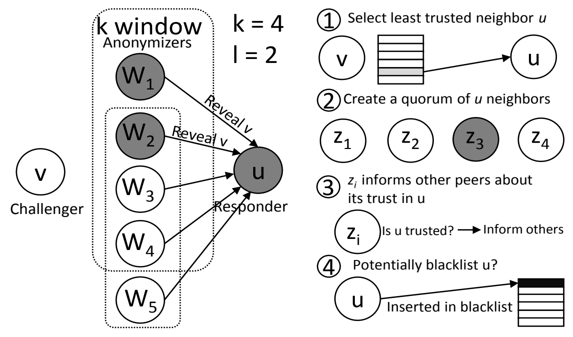
First, we integrate a gossiping protocol in our system, such as Cyclon [24]. Here, super peers periodically spread their contact information through the network[2]. In this manner, all peers eventually obtain the contact information of all other peers. Note that the overlay consisting of super peers usually is of a smaller size and higher stability. Hence, spreading the contact information of all peers is indeed feasible.

Second, we propose a new protocol for selecting diverse anonymizers using the information disseminated by the above gossiping protocol. As each challenger v has the contact information of all or most other peers, v periodically selects an anonymizer uniformly at random from all known peers. v then bases its decision on whether to remove a neighbor u from its routing table upon the last k queries; i.e., the decision is based on a sliding window of the most recent queries. In other words, v removes u from its routing table if:

- a received backpointer set is too large,
- a received backpointer set does not contain v, or
- v received fewer than l valid replies from the last k queries for u's backpointer set.

Otherwise, v keeps u for the time being but continues to periodically requests its backpointer set, see Figure 1a.

Malicious peers are mostly unable to bias the selection of anonymizers for such a gossiping protocol. Unless the sub-graph induced by the benign peers B is disconnected, dropping the contact information of honest peers is of little consequence as the blocked information reaches the peer via a different path. Similarly, spreading their own information at a higher frequency, malicious peers might bias newly joined peers initially. However, peers only store each peer's contact information once. As soon as a peer has received the contact information of all benign peers, the bias disappears. Thus, the peer selection is indeed close to uniform and the probability to select a malicious anonymizer peer is approximately equal to the fraction MI of malicious peers.


k window
Anonymizers
k = 4
l = 2
W1
W2
W3
W4
W5
Reveal v
Reveal v
v
Challenger
u
Responder
1 Select least trusted neighbor u
v
u
2 Create a quorum of u neighbors
z1
z2
z3
z4
3 zi informs other peers about its trust in u
zi
Is u trusted?
Inform others
4 Potentially blacklist u?
u
Inserted in blacklist

(a) Proactive mechanism (b) Reactive mechanism

Fig. 1. Detection mechanisms: a) Our proactive mechanism prevents malicious peers from occupying too many routing tables (malicious peers are marked with dark grey). Peers v anonymously request the backpointer set of their neighbors and remove those that i) reply with too large sets, ii) reply with sets that do not contain v, or iii) fail to reply too often (here: 3 out of the last 4 times). b) Our reactive mechanism aims to expel malicious peers executing a denial-of-service attack. Peers construct a quorum on the trustworthiness of a peer exhibiting a bad performance.

We now explain how our modification prevent the unintentional exiling of honest peers. The continuous changing of anonymizer nodes implies that eventually one set of anonymizer nodes is dominated by malicious peers. As a consequence, an honest peer v will remove an honest peer u from its routing table as malicious peers create the impression that peer u is unwilling to respond and, hence, is likely malicious. However, v does not expel u from the system but only from its own routing table, so the removal merely requires v to select a new neighbor. Such additional edge churn is acceptable if its frequency and the associated overhead is low.

Thus, as detailed in [15], the probability that more than l of k randomly chosen anonymizer peers are malicious corresponds to the probability that a binomially distributed random variable X with parameters k and f attains values of at least $l+1$, i.e.,

$$P(X \geq l) \sum_{i=l+1}^{k} \binom{k}{i} f^i (1-f)^{k-i} \tag{1}$$

For values of $f \leq 0.2$ and $l = \lceil k/2 \rceil$, the probability to accidentally accept a malicious peer is very low, as we show

[2]For brevity, we write 'peers' for the rest of the section rather than 'super peers'. However, only super peers participate in the proposed mechanisms, as malicious regular peers do not have significant impact on the system.

in Section VI.

B. Reactive Detection

Our reactive mechanism expels malicious peers based on a two-step process. First, a super peer v keeps track of the reliability of its neighbors. If a neighbor u does not perform adequately, v marks u as suspicious and initiates a quorum in order to decide upon u's continued presence in the system. The quorum consists of all peers with outgoing connections to u, i.e., peers in u's backpointer set, as the misbehavior of u primary affects these peers as they forward requests to u. Note that v uses the most recent backpointer set it received while executing the proactive mechanism from Section IV-A.

Second, the quorum then collectively decides to either allow the peer to remain in the system or not. This means that a malicious peer thus might remain in the system if either more than 1/3 of its in-coming connections are from malicious peers or it only selectively misbehaves in order to fool the majority of peers into believing that it is benign. Next, we specify i) how the peer v decides to initiate a quorum and ii) how to reach an agreement on the expulsion of a peer.

In order to determine a neighbor's reliability, v needs to be capable of determining if a request forwarded to the neighbor u was adequately processed by u. Usually, v cannot detect the misbehavior of u with absolute certainty but can determine if the requested service has been provided, e.g., if the correct file was retrieved. However, the lack of a positive response does not necessarily imply a misbehavior of u. Rather, a failed request can indicate a misbehavior by any other peer contacted via u to provide the desired service. Furthermore, an inability to provide the desired service, e.g., requesting a non-existent file, results in failure without the presence of misbehavior. Accordingly, individual failures to provide the desired services should not directly result in marking the corresponding peer as suspicious. Rather, we propose an iterative adjustment of the neighbors trust-values [25].

Peers assign their neighbors trust-values $tv_v(u)$, which are increased by δ_1 upon the successful completion of a request forwarded to the neighbor. On the other hand, if such a request fails, the trust value is decreased by δ_2. Note that peers periodically check their contacts' liveliness to differentiate between offline and non-cooperative peers. Periodically, peers v select their neighbor with the lowest trust value that has not been subject to a quorum for at least time τ. v initiates a quorum to decide if the peer should be expelled.

After v has decided to establish a quorum to decide upon removing a neighbor u, v retrieves the contact information of the peers in u's backpointer set from the most recent backpointer set received during the execution of the proactive mechanism from Section IV-A. As depicted in Figure 1b, each member w of this quorum ranks the trust values of their super peer neighbors $SN(w)$ and considers $L_{w,r}$, the set of the $\lfloor rSN(w) \rfloor$ neighbors with the lowest trust value for $r \in [0,1]$. If $u \in L_{w,r}$, then w agrees with the expulsion of u, otherwise not. Finally, w sends its decision to all other peers participating in the quorum. After a time-out, each quorum member considers the received votes. If more than half of them declare that u is malicious, all quorum members remove u from their routing table. In this manner, all honest peers in the backpointer set remove u from their routing tables.

If u is indeed expelled, the above protocol drastically reduces u's connectivity. Ideally, u does not have any remaining neighbors and cannot join again. However, malicious peers can remain connected to u and continue to advertise u as a potential neighbor to honest peers. Furthermore, the returned backpointer set might only be a subset of u's actual set.

Due to the in-degree bound, the number of additional honest neighbors should be small and restricted to peers that recently added u to their routing tables, so that those peers did not yet receive a backpointer without their entry. We thus suggest keeping a blacklist of expelled peers. In order to ensure that all peers in the list have indeed been removed, peers blacklisting others have to provide i) the backpointer set used for the quorum signed by the expelled peer u, and ii) the signed responses of all members regarding the expulsion of u. Peers can spread new entries to the blacklist via gossiping, so that eventually all honest peers remove blacklisted peers. For consistency, newly joining peers or peers changing status (offline to online) receive the latest blacklist prior to adding new contacts.

However, malicious peers can abuse the quorum mechanism to discredit honest peers. If the majority of the backpointers of an honest peer u points to malicious peers, these malicious peers can collectively decide to expel u. The remaining honest peers in the quorum remove u from their routing tables and effectively expel u from the network. Because the blacklisting algorithm requires the signatures of all quorum members, blacklisting is only possible if the peer was indeed expelled. So, the blacklisting does not allow for additional abuse. However, we show that the probability of malicious peers dominating the quorum of an honest peer is low in the following section.

V. Analysis

In this section, we analyze how effective the algorithms from Section IV are with regard to detecting malicious peers and erroneously discrediting honest peers. In the following, we present our analysis for discrediting honest peers. The analysis for malicious peers follows analogously and is provided in the extended version[3] in Section V-A and V-B.

A. Effectiveness of the Degree Bound

We evaluate the probability of accidentally removing an honest peer from a routing table. As the set of anonymizer peers changes over time and malicious peers are largely unable to bias or predict the selection, any peer v with a malicious neighbor eventually obtains a set of anonymizer peers with less than l malicious peers. In contrast, v also eventually obtains a set of more than $k-l$ malicious peers to check upon an honest neighbor u, thus erroneously removing u from its routing table. The parameters k and l should be chosen such that the duration of connections between honest peers is long, while malicious peers are removed nearly immediately. Hence, the rare removals of honest peers should barely affect the performance and allow the removed peers to obtain new neighbors easily. On the other

[3]https://1drv.ms/b/s!AtAeXR7J3ixLjU-HI4D0u2k6S-94

hand, the frequent removals of malicious peers from routing tables should force them to maintain the prescribed limit θ.

We model the response behavior as a Markov chain. Each state of the Markov chain $(X_i)_{i \in \mathbb{N}_0}$ corresponds to a set of anonymizer nodes. The absorbing states of the Markov chain correspond to the removal of a peer u from v's routing table. Then, we can derive the probability that a peer v removes its neighbor u as the probability that the Markov chain has reached an absorbing state.

Formally, the state space $S = \{0,1\}^k$ describes the distribution of malicious peers within the last k anonymizer nodes. In other words, let a vector $s = (s_1, \dots, s_k) \in S$ represents k anonymizer nodes with $s_i = 1$ indicating that the $k-i+1$-th recent anonymizer node is malicious. As M is the set of malicious peers, with $MI = \frac{|M|}{|S|}$ denoting the fraction of malicious super peers, we have $P(s_i = 1) = MI$ for all i and consequently $P(s_i = 0) = 1 - MI$. The initial probability of a state s is the probability to obtain a certain list of malicious and honest anonymizer nodes from the backpointer set:

$$P(X_0 = s) = \prod_{i=1}^{k} MI^{s_i}(1-MI)^{1-s_i} \tag{2}$$

The transition between states is governed by two rules. The set of absorbing states A_X, i.e., states s such that the transition probability $P(X_1 = t | X_0 = s) = 0$ for all states $t \neq s$, corresponds to all states s with $\sum_{i=1}^{k} s_i \geq k - l + 1$. In other words, absorbing states correspond to more than $k - l$ malicious anonymizer nodes, resulting in the removal of u from v's routing table. Transitions from a non-absorbing state to any state correspond to removing the first entry in the vector, i.e., the k-th recent anonymizer node, shifting all other entries, and adding a new entry. The new entry is either 1 or 0 with probability MI or $1 - MI$, respectively. Hence, for all non-absorbing states $s \in S$, we have

$$P(X_1 = t | X_0 = s) = \begin{cases} MI, & t_k = 1, t_i = s_{i+1}, i = 1..k-1 \\ 1 - MI, & t_k = 0, t_i = s_{i+1}, i = 1..k-1 \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

After enumerating all elements in S, we can construct an initialization vector I and a transition matrix T based on Eqs. 2 and 3. The probability distribution of the random variable X_i corresponds to $T^i I$. We obtain the probability of reaching an absorbing state at step i as $p_i = \sum_{s \in A} P(X_i = s)$. Expressed in terms of our original question, an honest peer u remains in the routing table after $k + i + 1$ anonymizer nodes have been contacted with probability $1 - p_i$. We derive the probability for various values of k and l in Section VI.

B. Effectiveness of the Quorum

In Section IV-B, we proposed a method to collectively expel malicious peers from the system rather than only removing them from a set of routing tables. Here, we analyze to which extent malicious peers can counteract this method by i) ensuring that quorums against malicious peers fail, and ii) discrediting honest peers to expel them from the system. Based on the results, we show that passive eclipse attacks are unlikely to have any impact. However, active eclipse attacks might result in eclipsing a small set of peers but only after remaining in the system for an extended time period.

Throughout this section, D denotes the average routing table size, which is equal to the average in-degree. In a well-performing system, D should be close to θ for maximal connectivity and performance.

Now, we derive the probability that malicious peers can successfully discredit honest peers. Let E denote the event that malicious peers dominate the backpointer set of size $|bp|$ of an honest peer and hence, can expel the peer from the system. The malicious peers have to control $\lfloor |bp|/2 \rfloor + 1$ of the pointers. As pointed out in Section V-A, malicious peers maintaining more than θ in- or outgoing connections are likely to be detected soon. Thus, the routing table size of a malicious peer is essentially bounded by θ, disregarding short-time connections that are quickly dissolved when the lack of adherence to the degree bound is revealed. The total number of routing table entries of all malicious peers is bounded by $\theta|M|$, while the number of edges in the system is $D|B \cup M|$. Hence, the fraction of edges to malicious peers is at most $\theta|M|/(D|B \cup M|) = \frac{\theta}{D}MI$.

Now, we consider how the fraction of malicious edges corresponds to the probability of discrediting an honest peer. If malicious peers establish random connections rather than targeting individual peers, each incoming connection of an honest peer v is to a malicious peer with probability $\frac{\theta}{D}MI$. Thus, the number of connections to malicious peers in an honest peer u's bp (approximately under the assumption $\theta << n$, where n is the network size) is binomially distributed with parameters $|bp|$ and $\frac{\theta}{D}MI$, such that

$$P(E) = \sum_{i=\lfloor |bp|/2 \rfloor + 1}^{|bp|} \binom{|bp|}{i} \left(\frac{\theta}{D}MI\right)^{i} \left(1 - \frac{\theta}{D}MI\right)^{|bp|-i} \tag{4}$$

In contrast, if malicious peers target a set V of peers, they should eventually control a considerable fraction of incoming connections of these peers. Assuming that each targeted peer has an average in-degree of D, malicious peers need to control at least $0.5D|V|$ of the edges to peers in V to discredit them. Thus, they can execute the attack on sets of size up to $|V| = \frac{2\theta}{D}|M|$. However, super peers are usually very stable, such that the connections between super peers are likely to change slowly and thus, establishing targeted connections should take considerable time. In particular, targeted peers might leave the system before the attack is successful. We evaluate the duration of targeted attacks in Section VI.

VI. Evaluation

We now show the overall effectiveness of our detection mechanism in two case studies. First, we demonstrate the importance of an effective detection mechanism by quantifying the impact of OEAs on super-P2P systems. Second, we show how different parameters affect the effectiveness of our detection mechanism and its communication overhead.

A. Simulation Framework

We leverage the widely used discrete event based OMNeT++ simulator [26] with the OverSim framework [27]. In order to realize communication between super peers, we make use of OverSim's GIA module, short for Gianduia, for unstructured overlays [28]. GIA communication is restricted to super peers: regular peers attach themselves to one random super peer and restrict their communication to this peer only. The super peer assignment leverages existing techniques [29].

Upon promotion to super peers, these peers receive a list of existing super peers, e.g., via gossiping and choose potential neighbors from this list. We assume all connections to be bidirectional, i.e., a node v has u in its routing table if and only if u has v in its routing table.

As discussed in Section II, both super and regular peers initiate lookup requests to retrieve content stored in the system. Regular peers contact their super peer and accordingly, super peers use flooding to discover the requested content.

The detection mechanism is implemented based on the description in Section IV. For the proactive mechanism, we consider two attack strategies. If a malicious responder u with more than θ neighbors encounters an honest anonymizer, u either replies with a random (but constant over one simulation run) backpointer set containing θ of its actual backpointers (denoted ***guess-reply***) or refuses to reply (denoted ***no-reply***). We assess the impact of these adversary behaviors below.

For the reactive mechanism, we consider two denial-of-service attacks. Attackers can either drop received lookups (denoted ***drop***) or return a fake reply (denoted ***FR***). Faking replies are usually more effective as only one reply is forwarded to the originator of the request. Hence, super peers may forward a fake reply despite finding a route to the correct target. However, if the content is authenticated, peers detect fake replies and discard them. Thus, it is also essential to consider the impact of *drop reply*.

B. Simulation Set-up and Metrics

In our simulation study, we consider overlays of 3,000, 6,000, and 10,000 peers in total to validate the performance of our approach when the overlay size scales. We set the ratio of super peers to 10%, which is common for such overlays [7], [30]. The maximal routing table size of honest peers corresponds to 25% of the total number of super peers and the maximal depth for flooding is 15. The fraction of malicious peers MI varies between 10% and 20%.

Each peer initiates a lookup every 4 seconds on average. Peers choose the target of the lookup randomly. However, for active OEAs (a targeted specific victim set), we limit the computed statistics to lookups addressed to peers within this set. Our simulation time is $1000s$ with 10 runs of each parameter combination. Statistics collection is every $15s$. The proactive detection starts at $t = 100s$ and is invoked against a peer every $10s$, and the reactive mechanism starts at $t = 500s$ to allow peers to acquire trust values and is invoked every $70s$.

In our second case study, we evaluate various parameter settings for both the proactive and the reactive mechanisms. For the proactive mechanism, we consider window sizes $k \in \{6, 12, 16\}$ and required correct replies $l \in \{3, 4, 6, 8\}$. For the reactive mechanism, we choose the interval $\tau = 100s, 150s$ between subsequent quorum initiations to the same peer and the factor r to determine the list of the lowest trust value peers $r = \{0.07, 0.1, 0.15, 0.2\}$. Initially, peers assign each of their neighbor a trust value of 0 and the trust increment δ_1 and decrement δ_2 are both 1.

In this manner, we cover a wide range of overall parameter combinations. In our discussion, we focus on the most prominent results of the parameter study. We focus on the following common metrics for characterizing the performance of lookup and detection mechanism:

We compute the following metrics to characterize the performance of our algorithms:

- **Lookup Success Ratio** (LSR), i.e., the fraction of successful lookups.
- **Detection Overhead** (DO), the ratio of detection-related messages in contrast to all messages sent in the overlay.
- **Malicious Ratio per RT** (MRT), the average ratio of malicious peers in an honest peer's routing table.
- **Expulsion Ratio per RT** (ERT), the average fraction of correctly expelled malicious peers.
- **Honest Keeping Ratio per RT** (HRT), the fraction of honest peers correctly retained in a routing table.

C. Case Study 1: OEA impact

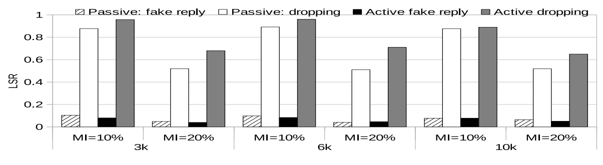
The main goal of this study is to highlight the severity of OEAs on super-P2P overlays based on the aforementioned metrics and parameters.

We start by considering a super-P2P system without any attacks. Without attacks, we achieve an LSR of at least 99%, more precisely 99.8% for 3,000 peers, 99.2% for 6,000 peers, and 99.0% for 10,000 peers. The slight chance of failure is due to the limited depth of the flooding, i.e., the rare cases where all peers storing a specific content are out-of-reach.

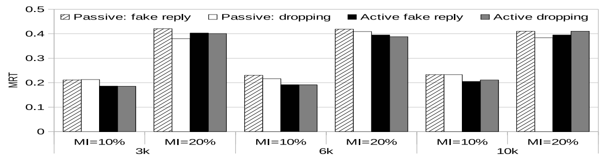
In contrast, attacks result in a low LSR, as illustrated in Figure 2a. In particular, *FR* results in a very low LSR with an average success ratio between 8%-11% for $MI = 10\%$ and 4%-7% for $MI = 20\%$. The low LSR is due to the instant termination of the lookup upon receiving a fake reply. The dropping behavior shows a higher average LSR of 80%-96% and 51%-89% for $MI = 10\%$ and $MI = 20\%$, respectively.

The reason for this higher LSR average is that, unlike in *FR*, the dropping behavior does not instantly results in a failed lookup. Hence, depending on the flooding value used, honest peers still may receive and accordingly respond with the destination's contact information, which results in a successful lookup. Nevertheless, the reduction in successfully served requests is considerable, which highlights the impact of OEA attacks on super-P2P systems.

We consider the MRT metric to show that malicious peers indeed make up a disproportional fraction of routing table entries for both passive and active OEAs. Indeed, for $MI = 10\%$, we observe an MRT between 18.5%-23.5% rather than the expected 10%, whereas $MI = 20\%$ increases MRT further to 38%-41.5%, as depicted in Figure 2b. Thus, malicious peers indeed manage to infiltrate routing tables.

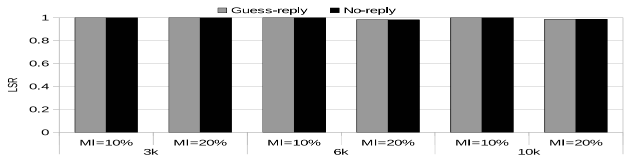


(a) Lookup success rate (`LSR`)

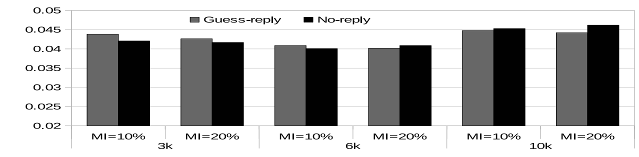


(b) Malicious ratio per RT (`MRT`)

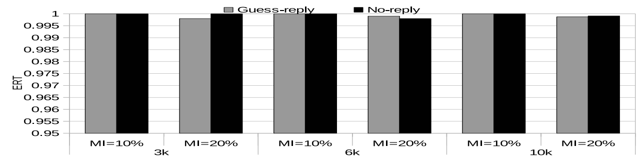
Fig. 2. OEA impact on Super-P2P networks



(a) Lookup success rate (`LSR`)



(b) Detection Overhead (`DO`)



(c) Expulsion ratio per RT (`ERT`)

Fig. 3. Detection mechanism performance

D. Case Study 2: Detection assessment

In this study, we evaluate the reliability, accuracy and effectiveness of the proposed two-fold detection mechanism. We start by considering the composite proactive and reactive mechanism for various network sizes. Afterwards, we consider the proactive and reactive mechanism individually. For the first, we vary the number of correct replies l whereas we consider the impact of the fraction of routing table entries r that are considered untrustworthy on the latter. We compare the simulation results with our analytical results from Section V.

a) Composite Mechanism: Figure 3 exemplifies the effectiveness of our detection mechanism for $k = 12$, $l = 6$, $\tau = 100$ and $r = 0.1$. We vary the network size between 3,000, 6,000, and 10,000 peers. Malicious peers make up either $MI = 10\%$ or $MI = 20\%$ of the set of super peers.

Our detection scheme is indeed highly effective. The composite detection mechanism increases the lookup success ratio to at least 98% for all considered parameters, as depicted in Figure 3a. In comparison to the often enormous failure rate experienced in case study 1, we now can achieve success ratios close to the no-attack scenario.

We only incur a detection overhead of 4%-5% regardless of the network size, as illustrated in Figure 3b. The reason for the low overhead is that the mechanism only requires contacting a low fraction of peers in comparison to the flooding-based lookup mechanism. Furthermore, the mechanism is only applied at a low frequency, i.e., every $10s$ for the proactive and every $70s$ for the reactive mechanism. More frequent detection attempts did not considerably improve the performance.

Figure 3c depicts the successful expulsion rate ERT, which provides a deeper understanding on how the detection mechanism works. Regardless of the attack strategy and infiltration rate, the expulsion ration ranges from 99.98%-100%.

Malicious peers are expelled locally via the proactive mechanism and globally via the reactive mechanism. Indeed, the result agrees with our analytical model, which states that the likelihood to expel malicious peers at some point is essentially 1. Similarly, both the simulation and the theoretical results (Equation 4) indicate that the chance to evict an honest peer is negligible. We consider the slightly higher chance of removing an honest node from a routing table in the following.

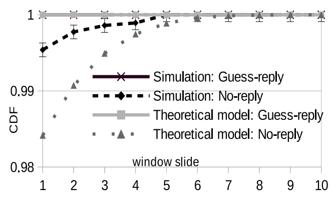
b) Proactive Mechanism: We compare the theoretical and simulation results for the proactive scheme and elaborate on the impact of the parameter l. Throughout this evaluation of the proactive mechanism, we set $MI = 0.1$.

In general, our theoretical results for removing malicious peers from routing tables match closely with the simulation. The agreement between theory and simulation holds for both attack strategies: no-reply and guess-reply. For guess-reply, we restrict our results to the peers not contained in the guessed reply, as peers that are always contained in the reply are inherently unable to detect that the peer is malicious. Indeed, as peers are allowed to maintain θ neighbors, keeping these θ peers does not violate the protocol.

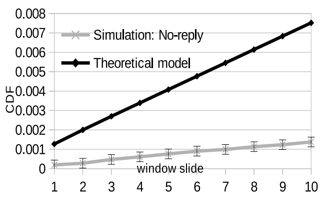
Figure 4a illustrates the closeness of the results for both guess-reply and no-reply with $k = 6$ and $l = 3$. We consider the cumulative distribution function of the fraction of removed malicious peers in terms of the number of windows (i.e., the number of queried anonymizers minus (k-1)). If malicious peers send a random reply when asked by an honest anonymizer, peers that are not contained in the reply recognize the misbehavior almost immediately. Hence, honest peers almost always remove malicious peers in the first window if the malicious peer uses guess-reply as a strategy, resulting in a removal rate of 1 in Figure 4a.

In contrast, if the malicious peer chooses not to reply, it might initially not be detected if there are at least l malicious anonymizers. However, the likelihood to continuously choose such l anonymizers decreases exponentially over the number of windows. Hence, malicious peers applying no-reply are detected rapidly as well, as depicted in Figure 4a.

We consider the honest removals in Figure 4b. The theoretical model presents an upper bound on the cumulative removal rates exhibited in the simulation. Overall, the chance of removing an honest peer from a routing table is several orders of magnitude lower than for a malicious peer. Hence, our proactive mechanism rarely disadvantages honest peers.

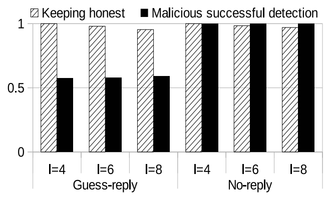


(a) Malicious-removals CDF

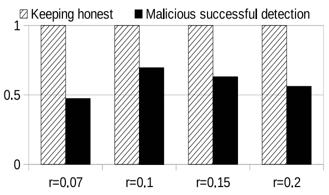


(b) Honest-removal CDF

Fig. 4. Comparing theoretical and simulation results $(k, l) = (6, 3)$



(a) Proactive: varying l



(b) reactive: varying r

Fig. 5. Evaluating parameters for detection mechanism

Now, we evaluate the effect of the number of correct replies l. For that purpose, we fix $k = 12$ and choose $l \in \{4, 6, 8\}$. As shown in Figure 5a, for both malicious behaviors types (guess-reply and no-reply), the honest keeping ratio HRT exhibits a slight decrease when l increases. The reason is the increased number of required honest anonymizers, which are increasingly unlikely to have over an extended period of time. In contrast, even at low values of l, the malicious detection rate is not equally affected as it is already at its maximum.

c) Reactive Mechanism: For the reactive mechanism, we vary $r \in \{0.07, 0.1, 0.15, 0.2\}$. We set $\tau = 100s$ and $MI = 10\%$ for this study. The effect of varying r is shown in Figure 5b. Choosing a very small value $r = 0.07$ decreases the probability of successfully expelling malicious peers. The lack of successful detection is due to that a specific malicious peer is unlikely to rank lowest in the routing table of the majority of its neighbors, as these might have multiple malicious peers. However, at $r = 0.1$, the ratio of successfully detected malicious peers increases to 70% as a result of including more malicious peers in $L_{w,r}$.

Note that at higher values $r = 0.15, r = 0.2$, honest peers are also included in $L_{w,r}$ by some of their neighbors. Nevertheless, the overall eviction rate of honest peers only increases slightly because honest peers are unlikely to be declared untrustworthy by all peers. To that end, we highlight the effectiveness of our detection mechanism for a wide range of parameters.

VII. Conclusion & Future Work

In this work[4], we highlight the severity of launching OEAs against super-P2P semi-structured overlays. Consequently, we propose a two-fold detection mechanism that constitutes of a proactive and reactive mechanisms.

Through an analytical and simulation-based evaluation, the proposed mechanism restores the overlay's reliability up to 100%, achieves 99% successful malicious detections, and entails a low network overhead (4%-5%). As a future work, we aim at combining the proposed algorithms with resilient online streaming P2P networks.

[4]Research supported in part by EC H2020 CIPSEC GA #700378 and BMBF TUD-CRISP

References

[1] Amad, M. et al. A Priority Based Lookup Model for VoIP Applications in Unstructured P2P Networks. In *Proc. IPAC*, pages 35:1–35:5, 2015.
[2] J. S. Gilmore et. al. A Survey of State Persistency in Peer-to-Peer Massively Multiplayer Online Games. In *IEEE Transactions on TPDS*, volume 23, pages 818–834, 2012.
[3] Wang, L. et al. Measuring Large-scale Distributed Systems: Case of Bittorrent Mainline DHT. In *Proc. P2P Computing)*, pages 1–10, 2013.
[4] Lua, E. et al. A Survey and Comparison of Peer-to-Peer Overlay Network Schemes. In *IEEE Commun. Surveys Tuts*, volume 7, pages 72–93, 2005.
[5] Cholez, T. et al. Detection and Mitigation of Localized Attacks in a widely Deployed P2P Network. *Peer-to-Peer Networking and Applications*, pages 155–174, 2013.
[6] Teng, H. et al. A Self-similar Super-peer Overlay Construction Scheme for Super Large-scale P2P Applications. In *Journal of Information Systems Frontiers*, volume 16, pages 45–58, 2014.
[7] Beverly Y. et al. Designing a Super-peer Network. In *Proc. Data Engineering*, pages 49–60, March 2003.
[8] Lopez, F. et al. Evaluating P2P Networks against Eclipse Attacks. In *Proc. Proceedia Technology*, volume 3, pages 61–68, 2012.
[9] Fantacci, R. et al. Avoiding Eclipse Attacks on Kad/Kademlia: An Identity based Approach. In *Proc. ICC*, pages 1–5, 2009.
[10] Baumgart, I. et al. S/Kademlia: A Practicable Approach towards Secure Key-based Routing. In *Proc. ICPADS*, pages 1–8, 2007.
[11] Li, Y. et al. Stochastic Analysis of A Randomized Detection Algorithm for Pollution Attack in P2P Live Streaming Systems. *Performance Evaluation*, 67:12:73–12:88, 2010.
[12] Dimitriou, T. et al. SuperTrust-A Secure and Efficient Framework for Handling Trust in Super Peer Networks. In *Proc. ICDCN*, pages 350–362, 2008.
[13] Rodrigues, R. et al. Rosebud: A Scalable Byzantine Fault Tolerant Storage Architecture. *MIT LCS TR/932*, 2003.
[14] Kubiatowicz, J. et al. Oceanstore: An Architecture for Global-scale Persistent Storage. *Proc. ACM Sigplan*, 35:190–201, 2000.
[15] Singh, Atul and others. Eclipse Attacks on Overlay Networks: Threats and Defenses. In *Proceedings of INFOCOM*, pages 1–12, 2006.
[16] Lo, V. et al. Scalable Supernode Selection in Peer-to-Peer Overlay Networks. In *Proc. Hot-P2P*, pages 18–25, 2005.
[17] Young, M. et al. Practical Robust Communication in DHTs Tolerating a Byzantine Adversary. In *Proc. ICDCS*, pages 2009–31, 2010.
[18] Ismail, H. et al. Detecting and Mitigating P2P Eclipse Attacks. In *Proc. ICPADS*, pages 224–231, 2015.
[19] Ismail, H. et al. Malicious Peers Eviction for P2P Overlays. In *Proc. CNS*, pages 216–224, 2016.
[20] Sen, S. et al. Commensal Cuckoo: Secure Group Partitioning for Large-scale Services. In *Proc. ACM OSR*, volume 46, pages 33–39, 2012.
[21] Cowling, J. et al. HQ Replication: A Hybrid Quorum Protocol for Byzantine Fault Tolerance. In *Proc. OSDI*, pages 177–190, 2006.
[22] Scheideler, C. How to Spread Adversarial Nodes?: Rotate! In *Proc. ACM STOC*, pages 704–713, 2005.
[23] Scheideler, C. et al. A Distributed and Oblivious Heap. In *Journal of ICALP*, pages 571–582, 2009.
[24] Voulgaris, S. et al. Cyclon: Inexpensive Membership Management for Unstructured P2P Overlays. In *Journal of Network and Systems Management*, volume 13, pages 197–217, 2005.
[25] Zhan, G. et al. Design and Implementation of TARF: A trust-aware Routing Framework for WSNs. In *IEEE TDSC*, volume 9, pages 184–197, 2012.
[26] G. Pongor. OMNeT: Objective Modular Network Testbed. In *Proc. MASCOTS*, pages 323–326, 1993.
[27] Baumgart, I. et al. OverSim: A Flexible Overlay Network Simulation Framework. In *Proc. INFOCOM*, pages 79–84, 2007.
[28] Y. et al. Chawathe. Making Gnutella-like P2P Systems Scalable. In *Proc. SIGCOMM*, pages 407–418, 2003.
[29] Min, S. et al. Optimal Super-peer Selection for Large-scale P2P System. In *Proc. ICHI)*, pages 588–593, 2006.
[30] Baset, S. et al. An Analysis of The Skype P2P Internet Telephony Protocol. In *Proc. INFOCOM*, pages 1–11, 2006.
[31] Hiller, M. et al. An Approach for Analysing the Propagation of Data Errors in Software. In *Proc. DSN*, pages 161–170, 2001.