ID Repetition in Kad

Jie Yu[1*], Chengfang Fang[2], Jia Xu[2], Ee-Chien Chang[2], Zhoujun Li[3]
[1]Department of Computer Science, National University of Defense Technology, China
[2]Department of Computer Science, National University of Singapore, Singapore
[3]School of Computer Science and Engineering, Beihang University, China
Email: yj@nudt.edu.cn, {c.fang,xujia,changec}@comp.nus.edu.sg, lizj@buaa.edu.cn

***Abstract*—ID uniqueness is essential in DHT-based systems as peer lookup and resource searching rely on ID-matching. Many previous works and measurements on Kad do not take into account that IDs among peers may not be unique. We observe that a significant portion of peers, 19.5% of the peers in routing tables and 4.5% of the active peers (those who respond to Kad protocol), do not have unique IDs. These repetitions would mislead the measurements of Kad network. We further observe that there are a large number of peers that frequently change their UDP ports, and there are a few IDs that repeat for a large number of times and all peers with these IDs do not respond to Kad protocol. We analyze the effects of ID repetitions under simplified settings and find that ID repetition degrades Kad's performance on publishing and searching, but has insignificant effect on lookup process. These measurement and analysis are useful in determining the sources of repetitions and are also useful in finding suitable parameters for publishing and searching.**

***Keywords*-Peer-to-Peer, Kad, ID, Repetition, Measurement**

I. Introduction

In DHT-based Peer-to-Peer (P2P) systems, each node or object (e.g. a keyword or file) is assigned with an identity (ID), which plays a crucial role during the lookup, publishing and searching processes. In some implementations of DHT, mechanisms are incorporated to ensure that the identifiers assigned to peers are unique. Many research works [1], [3], [4], [5], [18] have been devoted to this, including centralized certification and distributed certification. However, these certifications either require certain private information of users, such as cellular phone number [18] or email account [1], which are not acceptable for many P2P anonymous users, or need to combine with network layer information [5], e.g. IP address, sacrificing node mobility. For example, in Pastry, each identifier is derived from the IP address or the public key [14]. On the other hand, some implementations choose not to enforce ID uniqueness. One example is Kad, which is an implementation of Kademlia protocol [13] and has millions simultaneous users as to date. Each peer in Kad is free to choose an ID of 128-bit string as its identifier. The intention of such design decision is probably to reduce startup time and to support node mobility [24]. However, without enforcing uniqueness, multiple peers can share the same ID. This leads to the interesting question of whether such sharing and repetition of ID appears in actual DHT-based systems.

Indeed, we observe that, in Kad, there is a significant amount of repetition — 19.5% of the peers in routing tables and 4.5% of the *active peers* (i.e., peers who respond to the BOOTSTRAP requests.) do not have unique IDs. There are a few possible causes of the repetition, including intentional attacks on selected keywords, bugs in client software, crawlers, sensors or botnets that are setup to gather data, etc. There are many studies and analysis performed on Kad, including peer behaviors and distribution [17], [20], publishing and searching [19], lookup [22], security [18], [24], routing table [25], etc. Most of these works do not consider the repetition of the 128-bit ID. Without taking such repetition into account, measurements on Kad might be distorted, and parameters chosen for certain algorithms, for example publishing and searching processes, may not perform as expected.

To measure and study the effects of the ID repetition, we first gather ID related information of peers in Kad using a crawler for several weeks. Our observations and analysis can be summarized as below:

1) Among the peers who appeared in all routing tables, a low fraction (about 50%) of them respond to Kad messages. We believe that such low fraction of active peers is mainly due to long lifetime of contacts in routing tables (a peer left while its outdated contact is still in some routing tables), and UDP port aliasing (a peer frequently switches its UDP ports), instead of IP address aliasing or peers located behind NAT or firewall.
2) There is a significant amount of repetition — 19.5% of the peers appeared in routing tables and 4.5% of the active peers do not have unique IDs. Interestingly, the repetitions in routing tables follow a Zipf-like distribution. Such ID repetition would impact some measurements in previous works, such as lifetime or geographic distribution.
3) There are some *silent* groups of peers in the routing tables. Each group contains a large number (more than 100) of peers which all share the same ID, and none of them are active peers. Possibly, it is due to some modified clients or malicious tools connected to Kad that do not follow the standard Kad protocol.
4) We analyzed the effect of ID repetition on lookup and searching under some simplified and reasonable assumptions. The analysis shows that lookup performance will not be degraded while the searching performance of certain targeted keywords will be degraded significantly, and these two results are verified empirically. The anal-

*Work done during internship in National University of Singapore.

ysis also shows that by having slightly more peers to store the keywords, the lookup process can tolerate much more ID repetitions. Such analysis is useful in understanding Kad and provides a guide in choosing appropriate parameters for the relevant algorithms.

Organization. The rest of this paper is organized as follows. Section II presents the related work and Section III gives an overview of Kad. Section IV presents the measurement of Kad and Section V investigates the causes of low active rate of peers in routing tables and the causes of ID repetition. Section VI formally analyzes the effect of ID repetition on routing and searching and Section VII discusses the mitigation to this repetition. Conclusions are given in Section VIII.

II. Related Work

Many works have focused on the performance and security of Kad. K. Kutzner and T. Fuhrmann [10] measured the *IP address aliasing* in Overnet because of the use of dynamic IP addresses. During a two-week measurement, they found some IDs appeared to associate with more than 100 different addresses. M. Steiner *et al*. [17] observed *Kad ID aliasing* where a client changes its ID after one or several sessions. They found that most of this phenomenon appeared in China. We investigated this issues and believe that this is mostly due to the new versions of two popular download tools in China: FlashGet and Thunder. Both tools have integrated Kad but do not save ID information locally and randomly generate it in every session. To the best of our knowledge, there is no detailed investigation of ID repetition in Kad in the literature. Perhaps the most related work is the observation given by M. Steiner *et al*. [17], [21], that the distribution of ID over the 256 8-bit zones is not uniform as expected in Kad. They suggested that this could be due to software bugs.

Partially due to the lack of a certification service to verify peers' identities [1], Kad is vulnerable to a large spectrum of attacks, such as sybil attack [6], eclipse attack [16], routing table and index poisoning [12], [26], DDoS attack [24], [26], etc. Sybil attack [6] introduces multiple malicious peers into a distributed system, aiming to take control of the whole overlay network. M. Steiner *et al*. [18] discussed that sybil attack in Kad can be exploited to spy on publishing and searching traffic, eclipse contents or perform DDoS attacks. Eclipse attack [16] aims to separate a set of victim nodes from the rest of the overlay network. ID repetition discussed in this paper can be considered as a spacial case of eclipse attack, while it just tries to isolate the searching of keywords or files. P. Wang *et al*. [24] proposed to hijack routing table of clients using spoofed contacts to perform DoS attack to Kad network. To counter with vulnerabilities in Kademlia protocol, L. M. Aiello *et al*. [1] proposed *Likir*, a framework that is built on top of Kademlia and includes an identity based scheme and a secure communication protocol. It may provide an effective defense against above attacks, but is not easy to implement.

III. Overview of Kad

Kad is the first DHT implemented in real applications and it has millions simultaneous users as to date. Recently, Kad has been suggested to be the underlying infrastructure of other large-scale applications to increase the scalability and decrease the deployment cost, such as Second Life [23]. In this section, we give a brief overview on Kad and the related processes studied in this paper. Detailed information of Kad can be found in [17], [20]. IDs can be viewed as addresses in an overlay network that extends the functionality of the underlying network infrastructure [10]. It is recommended that each peer generates a 160-bit ID in the original design of Kademlia protocol [13]. However, in the implementation of Kad [7], each peer just has a 128-bit ID. Nevertheless, it is still a huge space and it is very unlikely that two randomly chosen IDs are identical.

The distance between two IDs is defined by the "XOR metric", which is a weighted Hamming distance. Given two IDs, k_1 and k_2, the binary representation of their distance can be computed by k_1 XOR k_2. For example, the distance between 0010 and 0100 is $0110_2 = 6$. This "XOR metric" plays a crucial role in the lookup, publishing and searching processes.

Each peer keeps a routing table which contains a list of contacts, and each contact mainly consists of 3 components: an ID, the respective address and the contact type. The contact type is an integer in $\{0, 1, 2, 3, 4\}$, and represents the level of availability. Value 0 means the best availability and value 4 means the worst. A contact with type value 4 will probably be eliminated from routing table at the next occasion. The contact type is typically assigned and updated as follows. When a new contact is added into the routing table of a peer, its type value will be initialized to 3. Afterwards, if the peer receives an "alive" sign (e.g. HELLO response) from the contact within 2 minutes, it will update the type value to 2, otherwise update it to 4. When a peer receives a message from a contact which has already stayed for more than two hours (half an hour, respectively) in its routing table, it will update the type value of this contact to 0 (1, respectively). The lifetime of a contact (that is, how long this contact could stay in a routing table) with type value 2, 1 or 0 is 1 hour, 1.5 hours or 2 hours respectively. J. Yu *et al*. [25] gave a detailed description and measurement on routing tables of Kad.

Given a *target*, say k, *lookup* is carried out to locate the peer in Kad network whose ID is closest to k. Kad employs an iterative process to lookup the target by querying routing tables in neighboring peers. Stutzbach *et al*. [22] gave a detailed description and performance analysis on the lookup process.

To *publish* a keyword, a 128-bit string k is derived from the keyword and treated as an ID. Information of the keyword is to be published and stored in γ peers whose IDs are closest to k, where γ is a parameter and is usually set to 10. To find these peers, typically, the iterative process of lookup is carried out with k as the target. Among the list of peers visited during lookup, the keyword is to be published to γ closest peers who

are willing to keep the information. To *search* for a keyword, similar process is carried out to determine γ closest peers. The search is successful if at least one of these γ peers is located.

IV. Measurement

A. Data Gathering

To obtain a snapshot of Kad, we deployed a crawler similar to *Blizzard* [17], [20]. The main difference is that we use BOOTSTRAP request instead of ROUTING request to traverse Kad network. Using BOOTSTRAP could be more effective, since a single BOOTSTRAP response contains 20 contacts while one ROUTING response contains at most 11 contacts [19]. We record two sets of peers: one set contains the peers who appear in routing tables; the other set contains the *active peers*, i.e., the peers who respond to the BOOTSTRAP requests. These two sets are analyzed separately to investigate ID repetitions in routing tables and among active peers. Note that most previous works [17], [20], [21], [22], [25] on measurement of Kad are done on the first set (i.e. all peers in routing table). To distinguish different peers, we keep track of the IP address and UDP port for each peer found. We also record TCP port and Kad version of each peer for a more in-depth analysis. The snapshots were obtained from 10 Feb 2009 to 15 Apr 2009, everyday around 18:00 GMT (daytime in East Asia) and 06:00 GMT (nighttime in East Asia). For each snapshot, we look for IDs, each of which maps to multiple ⟨IP address, UDP port⟩ tuples, and classify the corresponding peers as peers with repeated ID.

Each snapshot we collected contains about 2.5 to 3.8 million peers in routing tables and about 1.2 to 1.8 million active peers. This is slightly less than the numbers measured by M. Steiner *et al*. from March 2007 to May 2008, which are 3 to 4.5 million and 1.5 to 2 million respectively [17], [20]. The size of Kad during 06:00 GMT, corresponding to nighttime in East Asia, is about 20% to 30% larger than the size during the corresponding daytime. This is consistent with the measurements done by M. Steiner *et al*. [17], [20].

Note that there are only about half of peers in routing tables responding to the BOOTSTRAP requests. This low proportion may be due to the following reasons:

- $C1$: The routing table of each peer keeps every contact for a period of time. When a node left, its information may still remain in routing tables until its lifetime expires. When we send a message to such node, there would be no response.
- $C2$: Peers located behind NAT or firewalls can not receive any request messages directly and hence are unable to respond.
- $C3$: It takes 25 to 40 minutes for our crawler to collect information of all peers. During this period, some peers may change their IP addresses and thus can not receive request messages. This is possible as the Internet service providers may be running DHCP [10], [20] and it is known as *IP address aliasing*.
- $C4$: Peers can selectively or completely ignore Kad messages. These peers could be "selfish" ones who make use of Kad network but contribute little or none, or "bad" ones who exploit Kad for other purposes.

(a) Repetition among peers in routing tables

Number of repetitions	Mar 24 Tue	Mar 25 Wed	Mar 26 Thu	Average fraction
1	2689181	2696062	2691355	81.0%
2	335060	336282	341612	10.1%
3	91179	89637	92409	2.7%
4	46772	44968	45164	1.4%
5	26705	25955	26365	0.8%
6	15942	15720	16158	0.5%
7	12117	10815	10850	0.4%
8	9216	8664	9096	0.3%
9	7947	7173	7713	0.2%
10	6450	6560	6910	0.2%
$11-20$	31979	29990	29656	0.9%
$21-50$	15827	14718	15408	0.5%
$51-100$	4310	4565	4384	0.1%
$101-1000$	14088	12939	14338	0.4%
>1000	13887	14524	22358	1.2%

(b) Repetition among active peers

Number of repetitions	Mar 24 Tue	Mar 25 Wed	Mar 26 Thu	Average fraction
1	1497743	1463648	1473345	95.5%
2	26850	25786	28588	1.7%
3	5064	4869	4833	0.3%
4	4180	4208	4268	0.3%
5	3770	3825	3815	0.3%
6	3312	3042	3558	0.2%
7	2646	2296	2632	0.2%
8	2568	2032	1976	0.1%
9	1683	1332	1593	0.1%
10	1330	900	1220	0.1%
$11-20$	4052	2678	2769	0.2%
$21-50$	1472	1356	1305	0.1%
$51-100$	432	653	867	0.1%
$101-1000$	6429	6096	6273	0.4%
>1000	7591	6929	7464	0.4%

TABLE I
Distribution of peers by number of repetitions

M. Steiner *et al*. [17] suggested that this low active rate of peers in routing tables is mainly due to $C2$. However, in Section V-A we will show that $C2$ contributes a little to the fraction of unresponsive peers, while $C1$ and $C4$ are the main reasons. Furthermore, we observe a special case of $C4$, which we call *UDP port aliasing*.

B. Distribution of Repetitions

The fraction of peers without unique ID is significant, about 19.0% among peers in routing tables and 4.5% among active peers in every snapshot we obtained. In Section V, we will analyze why these two numbers are so different from each other. If there are m peers having k as their ID, let us say that *the number of repetitions* of k is m. Table I shows the distribution of peers by the number of repetitions of their IDs among peers in routing tables and active peers respectively. The first row shows the number of peers with unique ID, and the second row shows the number of peers whose IDs repeated twice, and so on. The percentages in the last column are the average over 3 days.

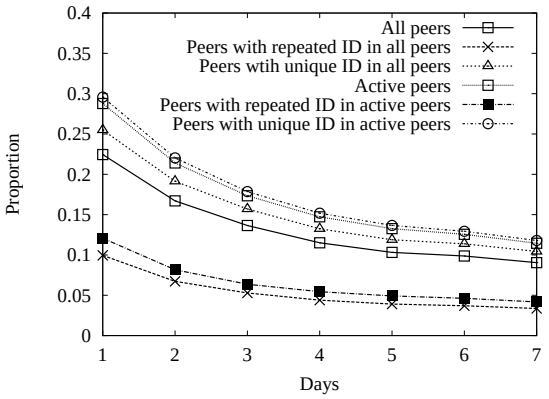
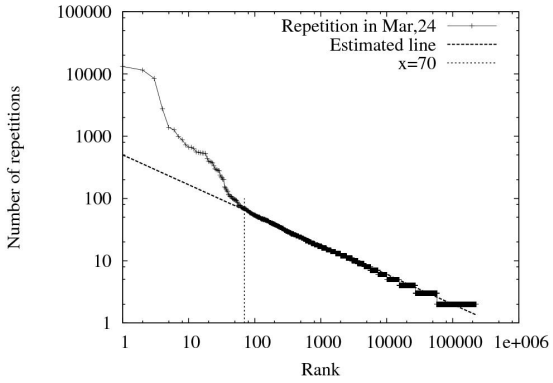


Fig. 1. Proportions of peers that also appeared in the reference day. Here "all peers" refers to peers in routing tables.

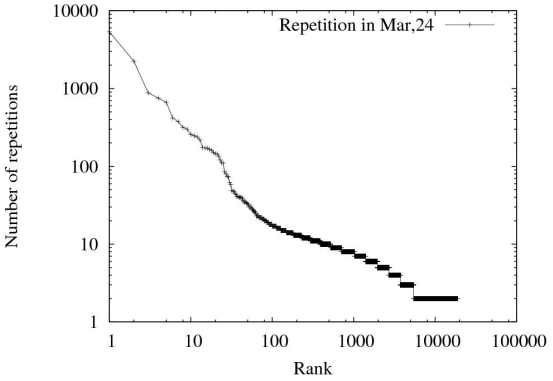
Note that in Table I the distributions are similar over a few days. We next look at the lifetime of individual peer. We first choose a snapshot (Mar 24, 2009) as reference. For each of the next following seven days, the fraction of peers that also appear in the reference snapshot are recorded. Fig.1 shows the fractions among peers with repeated ID and unique ID for both sets of peers in routing tables and active peers respectively. Clearly, there is a gap between the two curves for each set. Previous works on lifetime [17], [20] have not taken ID repetition into consideration. It would be interesting to further investigate the lifetime of different types of peers.

The number of repetitions of individual ID, sorted in decreasing order, are shown in Fig.2 in the log-log scale. It is interesting that Fig.2(a) shows a Zipf-like distribution (i.e. the probability of a randomly chosen ID having rank i is proportional to $\frac{1}{i^a}$ where a is a constant) of repetitions among peers in routing tables, for ranking 70 onwards. Note that the graph is in log-log scale, and the fraction of peers having ID with rank at most 70 is actually small, less than 1.7%. Hence, majority of the peers follows the Zipf-like distribution, with $a \approx 0.48$. Shamma *et al*. [15] observed the Google document frequency of the terms formed a Zipf-like distribution and M. Steiner *et al*. found that the distribution of keywords stored in Kad also follows Zipf-like distribution [19]. However, from Fig.2(b), the repetitions among active peers do not follow a Zipf-like distribution strictly. It is not clear why these two distributions are different and why the first distribution fits nicely to the Zipf-distribution.

Table II lists the details of the first few IDs with the largest number of repetitions among peers in routing tables and among active peers respectively. We investigate the groups of peers with the same ID. Here we use the file (config/ip-to-country.csv) provided in eMule client [7] to resolve IP addresses to countries. Note that for a single ID, the corresponding peers could spread over multiple countries. For some IDs, the corresponding peers are mainly concentrated in a single country, for example, the first and third rows in Table II(a). On the other hand, some IDs are widely spread, for example, the last two rows in Table II(a). We do not



(a) Repetition among peers in routing tables



(b) Repetition among active peers

Fig. 2. Number of repetitions per ID in log-log scale. Rank 1 means the largest, rank 2 means the second largest, and so on.

observe any interesting patterns in the geographic distributions of various IDs. Note that none of the 13,175 peers in the group listed in the first row of Table II(a) are active. We call such group a *silent group* and will further investigate it in Section V-C.

Fig.3 shows the geographic distribution of peers in the top 8 countries. The distribution of peers in each subset (i.e. all peers, peers with repeated ID in all peers, peers with unique ID in all peers, and so on. See Fig.3.) is different from others to a certain degree. Previous works on geographic distribution of Kad [17], [20] have not taken this into consideration. Especially, we find that among active peers, the percentage of peers in Spain is about 16.5%, while it is 35.4% among peers with repeated IDs, twice more than that of the global percentage.

C. Behaviors of Different Peers

To compare the behaviors of active peers with and without unique ID, we randomly select 1000 peers with ID repeated at least 3 times and 1000 peers with unique ID from the set of active peers, and then send routing requests of a randomly chosen ID and publishing requests of a random keyword. For those who agreed to store the keyword, we later send a request to retrieve the keyword. The result is shown in Table III.

(a) Most popular IDs in routing tables

Rank	ID (in hex format)	Number of Repetitions	Number of Countries	Country with Largest Number of Peers	Rank among Active Peers	Number of Repetitions in Active Peers
1	09262ce48db41838ce94c80cdaab3fab	13175	27	CHN(96%)	–	0
2	00000000000000000000000000000000	11557	87	CHN(39%)	1	5345
3	ab3d5a03c4892c603dd9beda87eda8d8	8492	48	CHN(95%)	2	2246
4	02ac8fc8a3e4caba1b1b520a623d5732	2751	23	CHN(95%)	4	752
5	5ad0327057fd75e85bf687011af12c3c	1379	5	ISR(98%)	3	879
6	e188d20e843abb978cd6eb24c591b846	1265	10	ISR(96%)	921	7
7	80f9ade2d68dc455efca6364cb9e9a31	984	17	FRA(95%)	5	664
8	dc173f913d2b41156fcb22373c4dcb74	875	2	CHN(99%)	13	230
9	efbef3a7f4406f876ee4dde0077ce8d0	726	12	KOR(96%)	11	247
10	9c2d76dd27f26ff1fe4f2a6544d53582	667	10	KOR(94%)	12	238
11	f111cbe0f1f0efcc2ceb37b27b0a6fb5	662	13	ARG(49%)	6	419
12	ad445207f1062287ef54341e0c110d7a	629	18	ITA(68%)	7	376

(b) Most popular IDs among active peers

Rank	ID (in hex format)	Number of Repetitions	Number of Countries	Country with Largest Number of Peers	Rank among All Peers	Number of Repetitions in All Peers
1	00000000000000000000000000000000	5345	66	CHN(29%)	2	11557
2	ab3d5a03c4892c603dd9beda87eda8d8	2246	26	CHN(96%)	3	8492
3	5ad0327057fd75e85bf687011af12c3c	879	5	ISR(98%)	5	1379
4	02ac8fc8a3e4caba1b1b520a623d5732	752	14	CHN(96%)	4	2751
5	80f9ade2d68dc455efca6364cb9e9a31	664	14	FRA(94%)	7	984
6	f111cbe0f1f0efcc2ceb37b27b0a6fb5	419	11	ARG(52%)	11	662
7	ad445207f1062287ef54341e0c110d7a	376	14	ITA(88%)	12	629
8	75b7002f8f0c5f0e8124db7e2b79ae0e	317	7	ESP(94%)	15	541
9	25ee26b837958af0307a086a51420368	301	28	ITA(28%)	13	559
10	672b3949cc995b6c47865eb796436380	257	11	ESP(86%)	14	551
11	efbef3a7f4406f876ee4dde0077ce8d0	247	8	KOR(96%)	9	726
12	9c2d76dd27f26ff1fe4f2a6544d53582	238	7	KOR(91%)	10	667

TABLE II
12 MOST POPULAR IDS IN ROUTING TABLES AND AMONG ACTIVE PEERS. HERE "ALL PEERS" REFERS TO PEERS IN ROUTING TABLES.

TABLE III
FRACTION OF SUCCESSFUL OPERATIONS

	Routing	Publishing	Retrieving	Searching By eMule
Unique	25.1%	89.8%	99.1%	56.7%
Repeated	27.2%	83.1%	97.8%	13.2%

The reason that we receive fewer routing responses is that peers will not respond if they do not have relevant routing information. From Table III, we can see that most active peers with repeated ID carry out Kad protocol as honestly as the peers with unique ID. This is a useful observation which leads to an assumption we made in Section VI: the majority of the peers with repeated ID would carry out routing, publishing and searching honestly.

We also search for these published keywords using eMule [7] (version 0.49b), which is a standard Kad client software. Different from the process described in the previous paragraph, whereby a keyword is retrieved from a known peer, an eMule client, given a keyword, first carries out the lookup process to locate the relevant peers, and then carries the retrieval process. As shown in the last column of Table III, the success rate of hitting keywords stored in peers with repeated ID is significantly lower than that in peers with unique ID. In Section VI-B we will formulate the success rate of searching in the situation with repeated IDs.

V. ANALYSIS OF REPETITION

In this section, we try to answer two questions:

- Why up to about 50% of peers in routing tables do not respond to the BOOTSTRAP request messages?
- Why the fraction of repetitions among peers in routing tables (i.e. 19.5%) is much larger than that among active peers (i.e. 4.5%)?

The first question has been overlooked in previous works[1]. However, it is important to study it because understanding it helps to design more effective management policies of Kad's routing tables, so as to increase the active rate of contacts. It also helps in solving the second question. The analysis to the second question provides insightful understanding to different kinds of repetitions.

A. Analysis of the low active rate of peers in routing tables

As discussed in Subsection IV-A, there are four possible causes, i.e., $C1$, $C2$, $C3$ and $C4$, that induce the low rate of active peers in routing tables. In the next few subsections, we will estimate the fraction of peers affected by each of the four causes.

1) Estimate the fraction of departure peers ($C1$): To measure the fraction of contacts with different types (i.e. types range from 0 to 4) in routing table, we add a module into the

[1]To the best of our knowledge, only M. Steiner *et al*. [17] briefly mentioned this question but they did not give a detailed investigation.

TABLE IV
FRACTION OF EACH TYPE OF CONTACTS IN A ROUTING TABLE

Type 0	Type 1	Type 2	Type 3	Type 4
82.3%	7.7%	7.1%	2.3%	0.6%

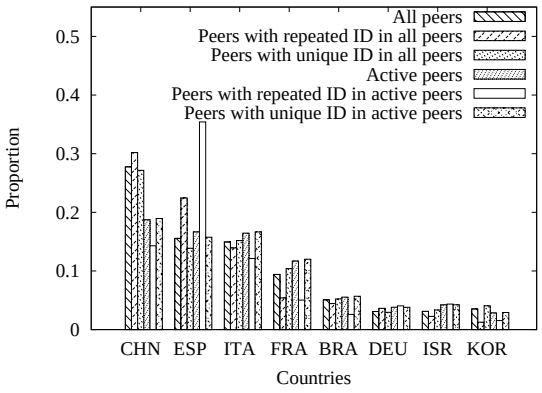


Fig. 3. Histogram of geographic distribution of different peers. Here "all peers" refers to peers in routing tables.

eMule software to record the routing table every 10 minutes. From a two-week monitoring, the average fraction of each type is shown in Table IV. It shows that most contacts in routing tables are of type 0, which means that the lifetime of most contacts in routing table are 2 hours. M. Steiner *et al*. [20] monitored a subset of live peers in Kad and obtained an approximated CDF (cumulative distribution function) of session times. We can then estimate the fraction of departure peers as:

$$\frac{1}{2}\sum_{i=0}^{4}[\text{fraction of type } i \ \times \ \text{CDF(lifetime of type } i)]$$

As a result, the fraction of departure peers is estimated as 24.2%.

2) Estimate the fraction of blocked peers ($C2$): One of the design goals of contact type is to distinguish peers located behind NAT or firewall, since these peers do not participate in routing or storing published information. Normally, these peers have contact types with values not smaller than 3. So, we can estimate an upper bounded of the fraction of these blocked peers in routing tables by the fraction of peers with contact type values 3 or 4, which add up to 2.9%. Note that since it was reported that 17% to 25% of Internet access is through NAT-enabled gateways [2], Kad has successfully prevented most of peers located behind NAT from "polluting" routing tables.

3) Estimate the fraction of peers with IP address aliasing ($C3$): In this situation, each peer appears as sharing the same ID with some others since its ID, as well as UDP port, TCP port and version do not change with IP address. Firstly, we argue that this fraction should be insignificantly small: (i) the "lease time" of the DHCP is typically longer than an hour, for instance, the default value for Windows Server 2003 is 8 days and most ISPs use the value of 24 hours, while it takes only 25 to 40 minutes for our crawler to collect information of all peers. (ii) even if the IP address of a peer is changed, there is a delay of a few minutes for this information to propagate to other routing tables.

Secondly, to support our argument, we estimate this fraction by investigating the distributions of peers in routing tables and active peers. For the convenience of analysis, we denote the set of peers with repeated ID in routing tables (or among active peers) as R_1 (or R_2), and the set of peers with the same ID, UDP port, TCP port and version but different IP addresses among peers in routing tables (or among active peers) as S_1 (or S_2). Similarly, we define P_1, P_2, P_3 and $\widetilde{P_2}, \widetilde{P_3}$ as follows:

- P_1: The set of peers in routing tables that change IP address before crawling[2], but keep the same ID, UDP port, TCP port and version.
- P_2: The set of peers in routing tables that change IP address during crawling, but keep the same ID, UDP port, TCP port and version.
- P_3: The set of peers in routing tables that share the same ID, UDP port, TCP port and version with some other peers.
- $\widetilde{P_2}$: The set of active peers that change IP address during crawling, but keep the same ID, UDP port, TCP port and version.
- $\widetilde{P_3}$: The set of active peers that share the same ID, UDP port, TCP port and version with some other peers.

Clearly, P_1, P_2 and P_3 is a partition of set S_1, and $\widetilde{P_2}$ and $\widetilde{P_3}$ is a partition of set S_2. We assume

$$\frac{|P_2 \cup P_3|}{|R_1|} = \frac{|\widetilde{P_2} \cup \widetilde{P_3}|}{|R_2|}.$$

Since we have counted $|S_1|$, $|S_2|$, $|R_1|$, and $|R_2|$ from the snapshot, hence,

$$\frac{|P_1|}{|R_1|} = \frac{|S_1|}{|R_1|} - \frac{|S_2|}{|R_2|} = 28.7\% - 23.5\% = 5.2\%.$$

Thus, the fraction of P_1 among peers in routing tables is $5.2\% \times 19.5\% = 1.0\%$. Since the lifetimes of most contacts are 2 hours while it takes less than 40 minutes by our crawler, we can safely conclude that the fraction of P_2 among peers in routing tables should be no larger than 1.0%. Overall, we can estimate that the fraction of peers with IP address aliasing, i.e. $P_1 \cup P_2$, is smaller than 2.0%.

4) Estimate the fraction of intentional unresponsive peers ($C4$): Inactive peers due to the causes of $C1$, $C2$ and $C3$ can be treated as peers who do not intend to remain inactive, but can not respond due to the ISP or are wrongly classified due to outdated information in the routing tables. We are not aware of other major causes and treat the remaining inactive peers not belong to $C1$, $C2$ and $C3$ as peers

[2]They are still in routing tables since their corresponding contacts have not expired.

who are programmed not to respond to the BOOTSTRAP messages. Hence, we estimate the fraction of such peers to be $50\%-(24.2\%+2.9\% + 2.0\%) = 20.9\%$. These peers might selectively or completely ignore Kad messages.

In sum, the low active rate of peers in routing tables is mainly due to the long lifetime of contacts in routing tables and intentional silence of some peers in response to Kad messages, instead of IP address aliasing or peers located behind NAT or firewall.

B. UDP port aliasing

A Peer in Kad communicate with others through UDP port, which should remain the same across all sessions. However, we observe that among peers in routing tables, there are some groups, such that all peers in each group have the same ID, IP address, TCP port and version, but different UDP ports. We call this phenomenon as *UDP port aliasing*, in contrast to IP address aliasing [10] and ID aliasing [17]. Such phenomenon is due to the frequent UDP port switching by some peers. One possible motivation of such behavior is that those peers try to avoid being connected by other peers in Kad. The fraction of peers with UDP port aliasing among peers in routing tables is 9.2%. Most of these peers contribute to low active rate discussed in the previous section.

Note that UDP port aliasing is not due to different peers located behind the same NAT and thus show the same public IP address in Kad. The reasons are: (i) different peers behind the same NAT still have different IDs; (ii) we have verified that at most one peer in each group responds to our requests at some point in time; and (iii) we have estimated that the fraction of blocked peers in routing table is about 2.9%, which is much smaller than 9.2%.

C. Silent groups

There are many groups of peers where all peers in a group share the same ID and they all do not respond to the BOOTSTRAP request message. For further investigation, routing and publishing messages are sent to them and yet there are still no response. These peers appear to be "silent" in Kad network. The fraction of these peers accounts to 8.7% among peers in routing tables. These silent groups can be divided into the following two types: (i) groups with peers that do not intent to remain silent and the repetition is due to "artifacts" of network protocols. This includes the multiple peers recorded due to IP address aliasing, which contribute to 1.0 % as discussed in Section V-A3. We are not aware of other mechanisms and thus take IP address aliasing as the main cause for groups in this type. (ii) groups with peers that intentionally do not respond to standard Kad requests. The second type of groups can be further divided into two subtypes: groups with UDP port aliasing and groups with peers across multiple ISPs. We further calculate that the first subtype contributes to 3.8% and thus the second subtype contributes the rest 3.9%.

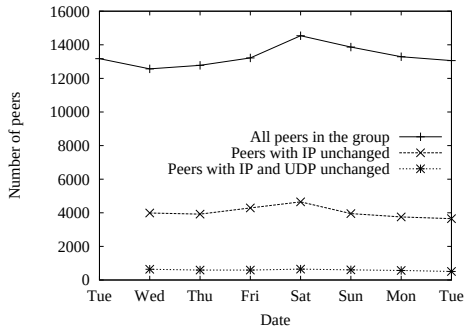


Fig. 4. The number of peers of the biggest silent group over a week.

We focus on the biggest silent group with ID corresponding to the first row in Table II(a). Since 96% of these peers are from China, we further investigate its distribution and find that they are widely distributed over most provinces of China. We can see from Fig.4 that its group size varies between 12,000 and 15,000, and the size at weekend is about 15% larger than that at weekday. We first randomly pick a snapshot of this group as reference. For each of the next seven days, the number of peers that share the same IP addresses or the same $\langle$IP addresses, UDP port$\rangle$ tuples and also appear in the reference snapshot are recorded. Fig.4 shows that the number of peers sharing the same IP addresses is about 4,000 while the number of peers sharing the same $\langle$IP addresses, UDP port$\rangle$ is only about 600, which means that these peers are possibly still involved in UDP port aliasing.

We suspect that most of these silent groups corresponds to some modified clients or malicious tools connected to Kad, which do not follow the standard Kad protocol. They just try to take advantage of Kad network and do not want to accept connections from normal peers. It is possible that they publish their peer information into routing tables of Kad, so that other members in the same silent group can locate them easily. Storm Worm [8], [9] is a well-known peer-to-peer botnet worm. It propagates via spam and communicates using overnet protocol, which is also based on the Kademlia protocol and is very similar to Kad. Since overnet has been dead for more than two years, it is possible that the attackers have switched or are trying to switch from overnet to Kad since it just requires a little modification on the original Storm Worm. One of our future works is to investigate their relationship.

D. Genuine Repetition

As analyzed above, peers can change their IP addresses or UDP ports dynamically before their information registered at routing tables expires. These peers are treated as multiple peers sharing the same ID by our crawler. This repetition is "false repetition" and it contributes to less than 11.2% among peers in routing tables. The others are different peers possibly across multiple ISPs while sharing the same ID, which is called *genuine repetition*. It contributes to more than 8.3% among peers in routing tables. Note that this fraction is still larger than 4.5% measured in active peers. It is because (i) some

silent groups never respond to our BOOTSTRAP requests and (ii) the total number of active peers is only half of all peers in routing tables. This genuine repetition may be due to intentional attacks on selected keywords, bugs in client software, crawlers, sensors or botnets that are setup to gather data, etc.

Note that although peers with "false repetition" do obey the ID assignment in Kad, they would still make lookup and searching less effective since Kad will ignore the other peers with the same ID, once it has found one. However, we can avoid this by changing Kad protocol slightly, such that it validates the aliveness of each contact before using it. In next section we will analyze the effect of genuine ID repetition on lookup and searching.

VI. Effect of ID Repetition

In this section, we investigate the effect of genuine repetition, since it disobeys the uniqueness assumption in ID assignment, and it is difficult to identify all peers with the same ID using standard Kad protocol without global information. We assume that peers with repeated ID carry out Kad's protocol honestly. Thus, its information will appear in other routing tables, and it will respond to lookup, searching and publishing processes. This assumption is supported by the observation made in Section IV-C. Here we first investigate the effect on lookup, followed by the effect on publishing/searching. Let n be the number of peers who are online at a particular time, and m the number of distinct IDs. For each ID i, let $f(i)$ be the number of repetitions.

A. Lookup

D. Stutzbach *et al* [22] gave the expected number of *hops*[3] required during lookup as follows:

$$1 + \frac{\log_2 n - 7.33}{6.58} \tag{1}$$

The above is for $\alpha = 3$ parallel lookup and is derived from the structure of *buckets* employed by the lookup processes. Some hidden parameters like the average *empty slots* and *stale contact per bucket* are obtained empirically. However, when there are significant fraction peers with repeated IDs, the expected number of hops in (1) may be overestimated.

When there are repetitions and the ID nearest to the target is shared by a few peers, we consider the lookup successful if one of these peers is found. Recall that all peers carried out the lookup process honestly. Thus, we can treated the group of peers sharing a same ID as a single "fat" peer. Intuitively, a "fat" peer acts like a normal peer during lookup, thus, the expected number of hops still follow the form of (1) but with number of peers reduced to m, the number of distinct IDs:

$$1 + \frac{\log_2 m - 7.33}{6.58} \tag{2}$$

[3]Note that the number of hops is not proportional to the time taken or number of peers visited. Nevertheless, it serves as a good indicator of the performance of lookup.

The expected number of hops in (2) does not consider the following scenarios: (i) information of a "fat" peer appears more often in routing tables comparing to a normal single peer, and (ii) "fat" peer lives longer in Kad. These certainly will affect the performance of lookup, and are worthy of further investigation.

The total number of distinct IDs in snapshots we gathered is about 86% of the number of peers, which is between 2.5 to 3.8 million. Applying (2), the expected number of hops is between 3.08 and 3.21. To verify this, we modify the eMule client to perform lookup of 10 randomly chosen targets every 10 minutes. This experiment is run for a week and the average number of hops is 3.14, which is within estimation. Actually, replacing n by $m = 0.86n$ in a logarithm has a very small effect. In sum, ID repetition does not degrade the lookup performance.

B. Publishing and Searching

Now we investigate the process of publishing and searching of a keyword, whose hashed value is k. To publish the keyword, its information is to be stored in γ peers whose IDs are closet to k, and to search for the keyword, information is to be retrieved from γ peers whose IDs are closest to k. There are two groups of peers of interests, the group P_{p} of γ peers involved in the publishing and another group P_{s} of peers involved in the searching. The search fails if there is no common peers in these two groups, i.e., $P_{\mathrm{p}} \cap P_{\mathrm{s}} = \emptyset$. Typically, in Kad, γ is chosen to be 10. To measure the number of peers hit in publishing/searching in present Kad, we modify the eMule client to publish a key to peers with unique ID and then search for these keys every hour for 24 hours[4] starting from 19:00 GMT, Mar 24, 2009. For each search, the number of peers holding the key is recorded, and 100 experiments are conducted in parallel. Fig.4 shows the average number of hit peers. Since as long as there is a single hit, the search is successful. Hence, the search/publishing is rather robust on average. However, as shown in Section IV-C, the success rate of hitting keywords stored in peers with repeated ID is significantly lower than that in peers with unique ID.

Let us now consider the scenario where IDs are repeated. When a keyword k is successfully published once to the ID i, only one peer with ID i will store this keyword. Assuming that the group of peers sharing an ID behave like a single peer during lookup, and only the ID i is visited during a search, the chance of successful retrieval is $\frac{1}{f(i)}$, where $f(i)$ is the number of repetitions for ID i as defined in the beginning of this section. In general, let $i_1, i_2, \ldots, i_d$ be the IDs of the peers holding the keyword and let $p(i_j)$ be the probability that i_j is visited during a search, then the probability that the search is successful is:

$$1 - \prod_{j=1}^{d} \left(1 - p(i_j) \cdot \frac{1}{f(i_j)}\right) \tag{3}$$

[4]The experiment was conducted for 24 hours since peers typically clear the stored keywords every 24 hours.

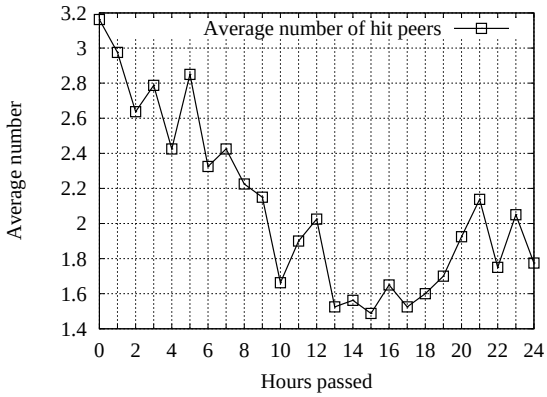


Fig. 5. The average number of hit peers by the eMule client over time, where each keyword is published to 10 peers with unique ID.

TABLE V
SUCCESS PROBABILITY WHEN 10 PEERS ARE HOLDING THE KEYWORD

Peers known by adversary	4	6	8	10
Attacked by 20 peers	77.1%	69.3%	60.5%	49.8%
Attacked by 40 peers	75.6%	65.1%	51.2%	33.5%

Note that the above can be approximated by $\sum_{j=1}^{d} \frac{p(i_j)}{f(i_j)}$ when $p(i_j)/f(i_j)$ is small for each j.

Consider a situation where an adversary wants to attack a keyword k using ID repetition and reduce the probability that the keyword is being retrieved[5]. The adversary has resources to generate N peers. Furthermore, he knows all the peers that store the keywords, say peers with ID $i_1, i_2, \ldots, i_d$, and information on the probability each of these peers will be visited during a typical search, that is, $p(i_1), \ldots, p(i_d)$. One way to reduce the probability of successful search is by generating peers with ID $i_1, \ldots, i_d$. An interesting question is how he should distribute the N peers to these d IDs, in other words, how to choose the numbers of repetitions $f(i_j)$ for each j so as to minimize the probability of (3), subjected to the constraints (i) $\sum_{j=1}^{d} f(i_j) = N + d$ and (ii) $f(i_j) \geq 1$ for each j. Note that each $f(i_j)$ must be an integer, and thus the optimization problem is not easy to solve. By relaxing the requirement to include real value for $f(i_j)$, it can be shown[6] that the probability is minimized by choosing:

$$f(i_j) = \frac{(N+d)\sqrt{p(i_j)}}{\sum_{\ell=1}^{d} \sqrt{p(i_\ell)}} \tag{4}$$

In the case where all $p(i_j)$ for $1 \leq j \leq d$ are the same, the peers should be equally distributed among the IDs to minimize the probability.

In practise, the adversary may not find all the peers that store the keyword. The success probability will increase when there are more peers storing the keyword that are not found by the adversary. Table V gives an illustration of this where there are a total of 10 peers holding the keyword, all peers have a 20% probability to be visited during a search, and the adversary can generate 20 or 40 peers and always choose the best strategy. From table V, we can see that when the adversary's information is limited, the additional resource for generating more peers does not help much.

[5]This is possible since an adversary can control a certain number of peers and serve for content providers (e.g. music or movies providers) by "isolating" particular keywords.

[6]Using the first order differential of (3).

TABLE VI
SUCCESS PROBABILITY WHEN 20 PEERS ARE HOLDING THE KEYWORD

Peers known by adversary	8	12	16	20
Attacked by 80 peers	94.1%	87.8%	76.1%	55.8%
Attacked by 160 peers	93.6%	85.8%	69.5%	36.2%

To summarize, ID repetition will reduces the performance of Kad by increasing possibility of a failure in searching for a published key. The probability of such failure depends on the number of repetitions, the number of IDs holding this key, and the probability that those IDs are visited during a search.

VII. MITIGATION

As mentioned in the introduction, it is possible to enforce ID uniqueness, for example, by requiring that each ID is derived from the IP address and UDP port using a cryptographic-secure hash function. However, employing this mechanism will lose node mobility since a new ID must be generated after the IP address is changed. Furthermore, it is tedious to support nodes located behind NAT, since these nodes need to obtain the translated IP address and port number to compute the ID.

A straightforward improvement is to treat the tuple ⟨ID, IP address, UDP port⟩ as the real identifier during publishing and searching. However, we have conducted a simulation with the ID repetition distribution according to the snapshot in Mar 24, 2009 and find that doing this actually *reduces* the probability of successful search.

One effective way to mitigate the ID repetition is to recruit more peers to store the keywords during publishing. Table VI shows the probability of successful search for a key held by 20 peers with 20% probability to be visited during a search and attacked by an adversary with 80 peers and 160 peers. Comparing to Table V, although the adversary has quadruple resources while the keyword is published only in twice more peers, the success probability increases. In general, from the approximation form of (3), to maintain the same probability, the adversary need to generate peers to increase both $f(i_j)$ and d. In other words, the resource required by the adversary is "quadratic" in the resource required by the publisher.

VIII. CONCLUSION AND FUTURE WORK

Kad does not enforce ID uniqueness and we observed that a significant fraction of peers have their IDs shared with others. Fortunately, the design of Kad is robust enough that even with the presence of ID repetition, there is no significant degradation in lookup performance. Although the performance of searching and publishing is severely degraded for targeted

keywords, our analysis shows that by publishing the keyword to slightly more peers, much more repetitions are required to degrade the searching performance. Nevertheless, if the mobility of peers is not a requirement, it would be desirable to prevent the problem by enforcing the ID uniqueness in the design. Although our studies classify the peers based on their behaviors, it is still not clear what are the sources of repetitions. It is interesting to further investigate and identify the sources. It is also interesting to find out whether such repetition exists in other DHT-based systems.

REFERENCES

[1] L. M. Aiello, M. Milanesio, G. Ruffo, and R.Schifanella. Tempering Kademlia with a Robust Identity Based System. In *Proceedings of P2P'08*, 2008.

[2] G. Armitage. Inferring the extent of network address port translation at public/private internet boundaries. *Tech. Rep. CAIA TR 020712A*, 2002.

[3] K. Butler, S. Ryu, P. Traynor, and P. McDaniel. Leveraging Identity-based Cryptography for Node ID Assignment in Structured P2P Systems. *IEEE Transactions on Parallel and Distributed Systems*, 01 Dec. 2008.

[4] M. Castro, P. Druschel, A. Ganesh, A. Rowstron, and D. S. Wallach. Secure routing for structured peer-to-peer overlay networks. In *Proceedings of OSDI'02*, 2002.

[5] J. Dinger and H. Hartenstein. Defending the Sybil Attack in P2P Networks: Taxonomy, Challenges, and a Proposal for Self-Registration. In *Proceedings of ARES'06*, 2006.

[6] J. R. Douceur. The Sybil attack. In *Proceedings of IPTPS'02*, 2002.

[7] Emule, http://sourceforge.net/projects/emule/

[8] T. Holz, M. Steiner, F. Dahl, E. W. Biersack, and F. Freiling. Measurements and Mitigation of Peer-to-Peer-based Botnets: A Case Study on Storm Worm. *First Usenix Workshop on Large-scale Exploits and Emergent Threats* (LEET), 2008.

[9] C. Kanich, C. Kreibich, K. Levchenko, B. Enright, G. M. Voelker, V, Paxson, and S. Savage. Spamalytics: An Empirical Analysis of Spam Marketing Conversion. In *Proceedings of ACM CCS'08*, 2008.

[10] K. Kutzner and T. Fuhrmann. Measuring large overlay networks - the overnet example. In *Proceedings of KiVS'05*, 2005.

[11] S. S. Lam and H. Liu. Failure Recovery for Structured P2P Networks: Protocol Design and Performance Evaluation. in *Proceedings of SIGMETRICS'04*, 2004.

[12] J. Liang, N. Naoumov, and K. W. Ross. The Index Poisoning Attack in P2P File Sharing Systems. In *Proceedings of INFOCOM'06*, 2006.

[13] P. Maymounkov and D. Mazires. Kademlia: A Peer-to-Peer Information System Based on the XOR Metric. In *Proceedings of IPTPS'02*, 2002.

[14] S.Rhea, D. Geels, T. Roscoe, and J. Kubiatowicz. Handling churn in a DHT. In *USENIX Annual Tech. Conf.*, 2004.

[15] D. Shamma, S. Owsley, K. Hammond, S. Bradshaw, and J. Budzik. Network Arts: Exposing cultural reality. In *Proceedings of WWW'04*, 2004.

[16] A. Singh, T. Ngan, P. Druschel, and D. Wallach. Eclipse attacks on overlays: Threats and defenses. In *Proceedings of INFOCOM'06*, 2006.

[17] M. Steiner, T. En-Najjary, and E. W. Biersack. A global view of KAD. In *Proceedings of IMC'07*, 2007.

[18] M. Steiner, E. W. Biersack, and T. En-Najjary. Exploiting KAD: Possible Uses and Misuses. *Computer Communication Review*, 37(5), 2007.

[19] M. Steiner, W. Effelsberg, T. En-Najjary, E. Biersack. Load Reduction in the KAD Peer-to-Peer System. In *Proceedings of DBISP2P'07*, 2007.

[20] M. Steiner, D. Carra, and E. W. Biersack. Long Term Study of Peer Behavior in the KAD DHT. Accepted for publication in *IEEE/ACM Transactions on Networking*, 2008.

[21] M. Steiner, E. W. Biersack, and T. En-Najjary. Actively Monitoring Peers in KAD. In *Proceedings of IPTPS'07*, 2007.

[22] D.Stutzbach and R. Rejaie. Improving lookup performance over a widely-deployed DHT. In *Proceedings of INFOCOM'06*, 2006.

[23] M. Varvello, C. Diot, and E. Biersack. P2P Second Life: experimental validation using Kad. In *Proceedings of INFOCOM'09*, 2009.

[24] P. Wang, J. Tyra, E. Chan-Tin, T. Malchow, D. F. Kune, N. Hopper, and Y. Kim. Attacking the Kad Network. In *Proceedings of SecureComm'08*, 2008.

[25] J. Yu and Z. Li. Active Measurement of Routing Table in Kad. In *Proceedings of DAS-P2P'09*, 2009.

[26] J. Yu, Z. Li, and X. Chen. Misusing Kademlia protocol to perform DDoS attacks. In *Proceedings of ISPA'08*, 2008.