EURECOM
Sophia Antipolis

Institut Eurécom
Department of Corporate Communications
2229, route des Crètes
B.P. 193
06904 Sophia-Antipolis
FRANCE

Research Report RR-07-205
Analyzing Peer Behavior in KAD

October 15th, 2007

Moritz Steiner, Taoufik En-Najjary, and Ernst W. Biersack

Tel : (+33) 4 93 00 81 00
Fax : (+33) 4 93 00 82 00
Email : {steiner,ennajjar,erbi}@eurecom.fr

1

Institut Eurecom, Sophia–Antipolis, France

Abstract

Distributed hash tables (DHTs) have been actively studied in literature and many different proposals have been made on how to organize peers in a DHT. However, very few DHTs have been implemented in real systems and deployed on a large scale. One exception is KAD, a DHT based on Kademlia, which is part of eDonkey2000, a peer-to-peer file sharing system with several million simultaneous users. We have been crawling KAD continuously for about six months and obtained information about geographical distribution of peers, session times, peer availability, and peer lifetime. We also evaluated to what extent information about past peer uptime can be used to predict the remaining uptime of the peer.

Peers are identified by the so called KAD ID, which was up to now assumed to remain the same across sessions. However, we observed that this is not the case: There is a large number of peers, in particular in China, that change their KAD ID, sometimes as frequently as after each session. This change of KAD IDs makes it difficult to characterize *end-user* availability or membership turnover. By tracking end-users with static IP addresses, we could measure the rate of change of KAD ID per end-user.

1 Introduction

Peer-to-peer systems have seen a tremendous growth in the last few years and peer-to-peer traffic makes a major fraction of the total traffic seen in the Internet. The dominating application for peer-to-peer is file sharing. Some of the most popular peer-to-peer systems for file sharing have been Napster, FastTrack, BitTorrent, and eDonkey, each one counting a million or more users at their peak time. Since these systems are mainly used by home-users and since the content shared is typically copyright-protected, the users of these systems often stay only connected as long as it takes for them to download the content they are interested in. As a result, the user population of these peer-to-peer systems is highly dynamic with peers joining and leaving all the time.

In this paper, we focus on a single peer-to-peer system, namely KAD, which is the publishing and search network of eDonkey. We want to characterize KAD in terms of metrics such as arrival/departure process of peers, session and inter-session lengths, availability, and lifetime.

To obtain the relevant raw data needed we decided to "crawl" KAD. Each crawl gives a snapshot of the peers active at that instant. The three major challenges in crawling are

- Time necessary to carry out a single crawl, which should be as small as possible to get a consistent view of the system.

- Frequency of the crawls, i.e. the time elapsed between two consecutive crawls should be small (no more than a few minutes) in order to achieve a high resolution for metrics such as session length.

- Duration of the crawl, which should be in the order of many months, to be able to correctly capture the longest session and inter-session lengths.

To meet all these goals, we built our own crawler that will be described in section 4.

While peer-to-peer systems have been explored previously using a crawler, the duration of these crawls was limited to a few days at best. We were able to crawl KAD for almost six months at a frequency of one crawl every five minutes, which allowed us to obtain a number of original results. We observed that:

- session lengths have a "long tail", with sessions lasting as long as 78 days.

- the distribution of the session lengths is best characterized by a Weibull distribution,with shape parameter $k < 1$. One property of Weibull distributed session lengths is that a peer that has so far been up for t units of time will – in expectation – remain up for a duration that is in the order of $O(t^{1-k})$. We can exploit this fact to use the past uptime in order to predict the remaining uptime.

- for many peers, the amount of time a peer is connected per day, called daily availability, varies a lot from one day to the next. This makes it difficult to predict daily availability

- the lifetime of a significant fraction of the peers observed can be as short as a single session. We could explain in part this surprising behavior by the fact that peers change their KAD ID, which is contrary to the assumption that KAD IDs are persistent.

- when classifying peers according to their geographic origin, the peers from China make about 25% of all peers seen at any point of time and Europe is the continent where KAD is most popular. We also saw a big difference between peers in China and Europe with respect to some of the key metrics such as session length or daily availability.

These results provide valuable impact to improve the design and performance of the KAD system.

The remainder of the paper is organized as follows. Section 2 presents related work followed by a section describing KAD. Section 4 presents the measurement methodology followed by two sections that contain the results. In the last Section 7 we present our conclusion and an outlook on future work.

2 Related Work

Overnet was the first widely deployed peer-to-peer application that used a DHT, namely Kademlia. The implementation of Overnet is proprietary and its operation was discontinued in September 2006 after legal actions from the media industry.

Overnet has been the subject of several studies such as [2, 7, 13] and up to 265,000 concurrent users have been seen online. The study most relevant to our work is the one by Bhagwan et al. [2]. A set of 2,400 peers was contacted every 20 minutes during two weeks. This study pointed out the *IP aliasing problem* that is due to the fact that many peers periodically change their IP address. So, in order to properly compute session times and other peer-specific metrics one needs to use the global identifier of the peer-to-peer system. This study also indicates, that for systems where peers leave permanently, the mean peer availability decreases as the observation period considered increases.

KAD is the first widely deployed open-source peer-to-peer system relying on a DHT. Two studies on KAD have been published by Stutzbach. The first explains the implementation of Kademlia in eMule [19] and the other [20] compares the behavior of peers in three different peer-to-peer systems, one being KAD. The results obtained for KAD are based on crawling a subset of the KAD ID space. We call a continuous subset of the total KAD ID space that contains all KAD peers whose KAD IDs agree in the high order k bits a **k-bit zone**. [20] crawled a 10-bit zone in 3-4 minutes and a 12-bit zone in approximately 1 minute. A total of 4 different zones were crawled during 2 days each. The short duration of the crawls implies the maximum values for some metrics such as session up-times or inter-session times that can be observed are naturally limited to 2 days. The paper by Stutzbach [20] is the most relevant with respect to our work and we refer to the results reported by Stutzbach at several occasions. As we will see, some of our conclusions do not agree with the ones made by Stutzbach, which is in part due to the fact that a crawl duration of two days is too short to correctly sample KAD with respect to some of the key metrics such as session durations.

3   Background on KAD

KAD is a Kademlia-based [9] peer-to-peer DHT routing protocol implemented by several peer-to-peer applications such as Overnet [14], eMule [5], and aMule [1]. The two open–source projects eMule and aMule do have the largest number of simultaneously connected users since these clients connect to the eDonkey network, which is a very popular peer-to-peer system for file sharing. Recent versions of these clients implement the KAD protocol.

Similar to other DHTs like Chord [18], Can [16], or Pastry [17], each KAD node has a global identifier, referred to as KAD ID, which is 128 bit long and is randomly generated using a cryptographic hash function. The KAD ID is generated when the client application is started for the first time and is then permanently stored. The KAD ID stays unchanged on subsequent join and leaves of the peer, until the user deletes the application or its preferences file[1]. Therefore, using the KAD ID, a particular peer can be tracked even after a change of its IP address.

---

[1]As we will see later, not all peers in KAD behave this way.

3.1 Routing

Routing in KAD is based on prefix matching: Node a forwards a query, destined to a node b, to the node in his routing table that has the smallest XOR-distance. The XOR-distance $d(a, b)$ between nodes a and b is $d(a, b) = a \oplus b$. It is calculated bitwise on the KAD IDs of the two nodes, e.g. the distance between $a = 1011$ and $b = 0111$ is $d(a, b) = 1011 \oplus 0111 = 1100$. The fact that this distance metric is symmetric is an advantage compared to other systems, e.g. Chord, since in KAD if a is close to b, then b is also close to a.

The entries in the routing tables are called *contacts* and are organized as an unbalanced *routing tree*: A peer P stores only a few contacts to peers that are far away in the overlay and increasingly more contacts to peers as we get closer P. For details of the implementation see [19]. For a given distance P knows not only one peer but a *bucket* of peers called contacts. Each bucket can contain up to ten contacts, in order to cope with peer churn without the need to periodically check if the contacts are still online.

For routing, a message is simply forwarded to one of the peers from the bucket with the longest common prefix to the target. Routing to a specific KAD ID is done in an iterative way, which means that each peer on the way to the destination returns the next hop to the sending node. While iterative routing experiences a slightly higher delay than recursive routing, it offers increased robustness against message loss and it greatly simplifies crawling the KAD network.

3.2 Publishing

A **key** in a peer-to-peer system is an identifier used to retrieve information. KAD distinguishes between two different keys:

- A **source key** that identifies the content of a file and is computed by hashing the *content* of a file.

- A **keyword key** that classifies the content of a file and is computed by hashing the tokens of the *name* of a file.

In KAD keys are not published just on a single peer that is numerically closest to that key, but on 10 different peers whose KAD ID agrees at least in the first 8-bits with the key. This zone around a key is called the tolerance zone.

Figure 1 shows an example of the publishing process. A peer wants to publish a file with the name `the_matrix`. This filename will result in two keywords, "the" and "matrix". All relevant references to the real file are generated, such as the source key and the the keywords with the attached metadata. Next, the keywords "the" and "matrix" are published, pointing to the source. Finally, the source is published, pointing to the publishing peer.

Keys are periodically republished: **source keys** every 5 hours and, **keyword keys** every 24 hours. Analogously, a peer on which a source key or keyword key
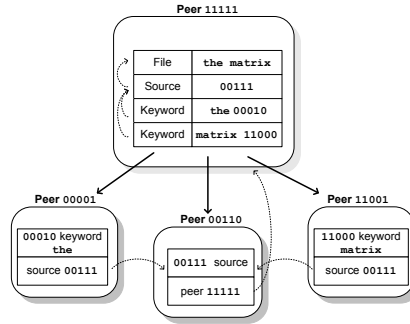
Figure 1: Sketch of the 2-level publishing scheme

was published will delete the information after 5 and 24 hours, respectively. This way re-publishing is done exactly the same way as publishing.

4   Measurement Methodology

We have developed Blizzard, our own crawler for KAD, with the aim to crawl KAD frequently and over a duration of several months. Our crawler logs for each peer P the time of the crawl, the IP address of P, and the KAD ID of P.

In a large peer-to-peer system such as KAD, peers are constantly joining and leaving, which makes it difficult to get a consistent view of the system. Therefore, the overall duration of a single crawl should be as short as possible. To speed up the crawl, previous crawlers often were distributed and ran simultaneously on multiple machines. We noticed that in a distributed crawl a lot of CPU time is used up for the synchronization between the different machines. The main idea of Blizzard is to use *only one machine* and keep all relevant information in main memory. After the crawl is completed, the results are written to disk. The implementation of Blizzard is straightforward: Blizzard runs on a local machine knowing several hundred contacts to start with. It uses a simple breadth first search and iterative queries: It first queries peers among its contacts in order to get to know more peers and so on.

There are various pitfalls when crawling a peer-to-peer system, such as incomplete data due to crawler crashes, loss of network connectivity, or random failures due to temporary network instability. To address these problems, we run simultaneously two instances of Blizzard, one at the University of Mannheim, Germany, connected to the German research network, and a second one at Institut Eurécom, France, connected to the French academic network. Doing two parallel crawls turned out to be very useful: at some point, due to network problems, one instance of the crawler was seeing fewer peers than the other one. Also, occasionally one of the two crawlers crashed. We take the raw data of the two parallel crawls and merge them in such a way that, for a given round of the crawl, a peer is considered

6

up when he has been seen by at least one crawler.

The speed of Blizzard allows us to crawl the entire KAD system (entire KAD ID space), which was never done before. Such a **full crawl** of KAD takes about 8 minutes. The first million different peers are identified in about 10 seconds, the second million in 50 seconds, thereafter the speed of discovery decreases drastically since most of the encountered peers have already be seen before during the same crawl. A full crawl of KAD produces about 3 GBytes of inbound and outbound traffic each.

A full crawl was done three times a day from 2006/08/18 to 2006/08/26 and from 2006/10/03 to 2006/10/12. Another full crawl has been started 2007/03/20 and is carried out up to now once a day.

A full crawl generates an extremely high amount of trace data and of network traffic (with peak data rates close to 100 Mbit/sec). Carrying out just 3 crawls per day is not really sufficient to capture the dynamics of KAD at short timescales. For this reason, we decided to carry out a **zone crawl** on a 8-bit zone, where we try to find all active peers whose KAD ID have the same 8 high-order bits. Such a zone crawl, that explores one 256-th of the entire KAD ID space, takes less than 2.5 seconds. A zone crawl for the KAD IDs whose 8 high order bits are 0x5b was done once every 5 minutes from 2006/09/23 to 2007/03/21, which is *slightly less than 6 months*. We will see in Section 5 that it is possible to infer the results about the entire KAD ID space from the results obtained with a zone crawl.

4.1 Data Cleaning

Crawling happens in "rounds" with a time difference of five minutes, with the two crawlers being "synchronized". A peer that replied to at least one of the two crawlers during round i is considered to be up at round i.

However, even when crawling from two sites, we realized that it is still possible that the requests sent to a peer or its replies can get lost and a peer that is up may be declared being down. One reason can be that the path between the two crawlers and the peer is disrupted somewhere close to that peer. In this case, the crawlers will not receive a reply from that peer even when it is up and running. While it is not possible to tell exactly why a peer is not answering, we implemented the following data cleaning rule that we consider "reasonable": when a peer P that has been reported up at round $i - 1$ does not reply to either of the two crawlers during the next round i, and then replies again during round $i + 1$, then peer P will be also considered up at round i.

Data cleaning will increase the session and inter-session times. Figure 2(a) plots the cumulative distribution function (CDF) of session and inter-session times. About one third of all inter-session times are just 5 minutes; these values will disappear after data cleaning, when the smallest inter-session time becomes 10 minutes (see 2(a)). Also, after data cleaning some of the adjacent sessions will get merged and the session times increase (see 2(b)).
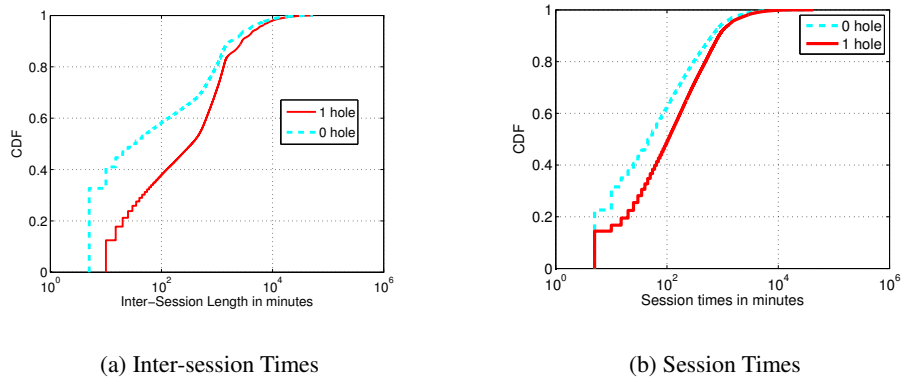
(a) Inter-session Times   (b) Session Times

Figure 2: CDF of the inter-session and session times (1 hole: after data cleaning, 0 hole: before data cleaning).

5 Global View of KAD

In this section, we will present results about the full crawl of the system and the zone crawl, moreover we will characterize the fact of IP address aliasing and KAD ID aliasing.

5.1 Full Crawl

During a full crawl, we found between 3 and 4.3 million different peers. Between 1.5 to 2 million are not located behind NATs or firewalls and can be *directly contacted* by our crawler. Peers behind NATs or firewalls use KAD to publish information about the content they share, but do not *participate* in storing published information, and make therefore no contribution to the operation of KAD.

In the rest of the paper we will only report statistics on the peers that our crawler could contact *directly*. As we can see in Figure 3 the number of peers seen varies according to a diurnal and a weekly patterns and reaches its peak during the weekend, where the population is about 10% higher.

In Figure 4, we plot the distribution of the percentage of peers seen per country. We used the Maxmind database [8] to resolve IP addresses to countries and ISPs. The continent with the highest percentage of peers is Europe (Spain, France, Italy and Germany), while the country with the largest number of peers is China. Less than 15% of all peers are located in America (US, Canada, and South America). We can also see that the geographic distribution of the peers obtained with the two zone crawls of an 8-bit zone each is very close to the result obtained with the full crawl, which is to be expected since the KAD IDs are chosen at random.

We can therefore estimate the total number of peers in KAD by simply counting the number of peers in a zone. We do a zone crawl of one 256-th of the entire KAD ID space and use Chernoff Bounds (see [12] Chapter 4) to estimate the total
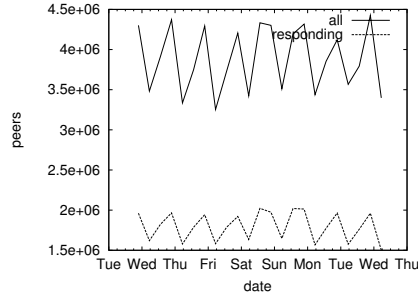
8

Figure 3: The number of KAD peers available in **entire** KAD ID space depending on the time of day.
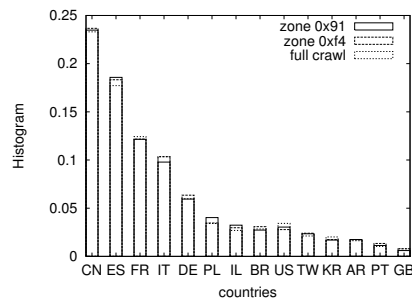

Figure 4: Histogram of geographic distribution of peers seen on 2006/08/30.

population size and to tightly bound the estimation error.

Let $N(t)_{part}$ be the number of peers counted during a zone crawl of an 8–bit zone at time t and $\hat{N}(t) := 256 * N(t)_{part}$ the estimate for the total number of peers in the KAD system. The true value $N(t)$ for total number of peers at time t is very close to the estimate $\hat{N}(t)$, with high probability. More precisely: $Prob[|N(t) - \hat{N}(t)| < 45000] \geq 0.99$, which means that our estimate $\hat{N}(t)$ has most likely an error of less than 3% for a total population of at least 1.5 million peers.

Since the full crawl was quite expensive in terms of resources, we will extensively rely on the zone crawl to obtain much of the relevant information about KAD.

5.2 Zone Crawl

All the results in the following subsection were obtained using the zone-crawl of the 8-bit zone `0x5b` that lasted for 179 days.

In Figure 5, we plot the number of peers seen that originate from China and some European countries. For the peers of each country, we can clearly identify

9

a diurnal pattern, with a peak around 9 PM local time. The eight hour time shift between Europe and China is clearly visible.
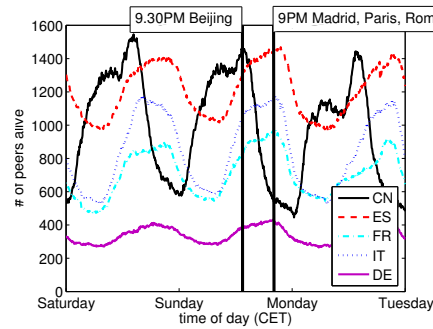

Figure 5: Peers online according to country of origin.

Table 1 summarizes the basic findings on the zone crawl. The peers seen came from 168 different countries and 2384 providers. For the KAD IDs seen the 1st day of our zone crawl, we observe that about $\frac{1}{3}$ of the peers come from Europe and about $\frac{1}{4}$ from China. If we compare the **lifetime** of the peers, which is defined as the difference between the time a given KAD ID was seen the last time and the time this KAD ID was seen the first time, we notice that the lifetime of peers in China is much smaller than the one for peers in the other countries. More than half of the peers in China were seen for the duration of only *one* session. We will come back to this point in subsection 5.3.

|	Total	China	Europe	Rest
Different KAD IDs	400,278	231,924	59,520	108,834
Different IP addresses	3,228,890	875,241	1,060,848	1,292,801
KAD IDs seen for a single session	174,318	131,469	11,644	31,205
KAD IDs with LT $\leq$ 1 day	242,487	183,838	15,514	43,135
KAD IDs seen for the first time on				
- 1st crawl	5,670	455	2,879	2,336
- 1st day	18,549	4,535	6,686	7,328
- 60th day	1,893	1,083	259	551
KAD IDs seen for the first time on 1st day				
- with LT $\leq$ 1 day	2,407	1,568	286	553
- 1 day $<$ LT $\leq$ 1 week	1,368	497	393	478
- 1 week $<$ LT $\leq$ 1 month	2,735	791	944	1,000
- LT $>$ 1 month	12,039	1,679	5,063	5,297
- LT $>$ 3 months	8,423	936	3,679	3,808

Table 1: Key facts about the zone crawl spanning 179 days (LT=Lifetime).

10

Arrivals and Departures

Since we crawl KAD once every 5 minutes, we can determine the number of peers that join and leave between two consecutive crawls. Knowing the arrival rate of peers is useful since it allows to model the load in KAD due to newly joining peers. Each time a peer joins, it first contacts other peers for information to populate its routing table, before it publishes the keywords and source keys for all the files it will share.

In figure 6(a) we depict the CDF (cumulative distribution function) of the number of peers that arrive and that depart between two consecutive crawls. We see that the distributions for arrivals and departures are the same. This is to be expected, since we observe the system in "steady state": in this case, the system should behave like $G/G/\infty$, for which, according to Little's Law, the arrival rate is equal to the departure rate [11].

The arrival process is very well described by a Negative Binomial distribution (see figure 6(b))
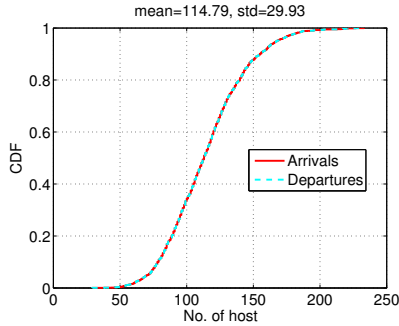
(a) CDF of the number of arrivals and departures.
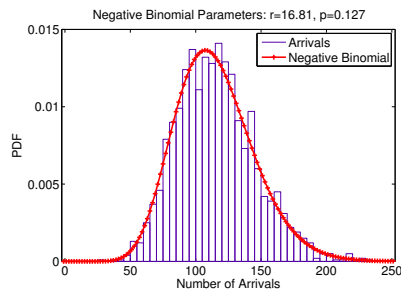
(b) PDF of the number of arrivals

Figure 6: Peer arrivals between two crawls during the first week.

5.3   Aliasing

IP Address Aliasing

It has been known for quite some time [2, 7] that peers may get frequently assigned new IP addresses, which is referred to as **IP address aliasing**. We observed a total of 400,278 distinct KAD IDs and 3,228,890 different IP addresses (see table 1). As we see in figure 7, the number of different IP addresses per peer is strongly correlated with the peer lifetime.
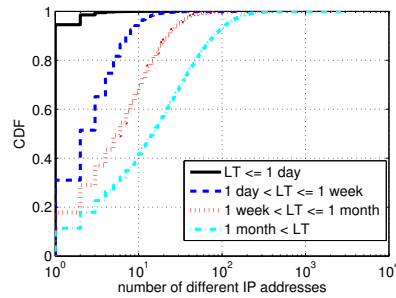
Figure 7: The number of different IP addresses reported per KAD ID during a period of six months.

KAD ID Aliasing

Figure 8 reports the number of *new* KAD IDs per day. i.e. KAD IDs seen for the first time, according to country of origin. More than 50% of the new KAD IDs are from peers in China, which is more than one order of magnitude more than the number of new KAD IDs seen for any other country such as Spain, France, or Germany.

We see in our zone crawl approx. 2,000 new KAD IDs a day, which means that for the entire KAD system the number of new KAD IDs per day is around 500,000. If we extrapolate, this makes about 180 Million KAD IDs a year. It is hard to believe that there exist such a large number of different end-users of KAD.
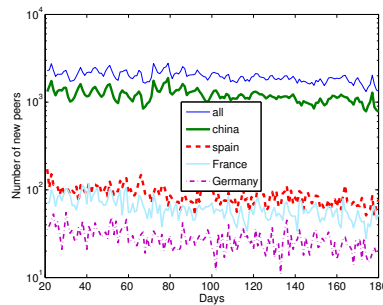
Figure 8: New KAD IDs according to country of origin.

We were curious to find out whether the end-users really stop using KAD after one session, or whether the same users come back with a different KAD ID. We refer to the phenomenon of non-persistent KAD IDs as **KAD ID aliasing**, in contrast to IP address aliasing.

To investigate KAD ID aliasing, we need to look for peers with static IP addresses, wich we can track for non-persistent KAD IDs. We know that, for instance in France, one of the ADSL providers (Proxad) assigns static IP addresses to cus-

12

tomers located in areas where the service offer is completely "un-bundled", while France Telecom-Orange changes the IP addresses of ADSL customers on a daily basis.

To find peers with static IP addresses, we started in March 20th, 2007 to carry out one full crawl a day. We take the logs of the two full crawls of March 20th and March 30th and extract the 160,641 peers that have the same IP address and the same KAD ID in both logs. We call this set of IP addresses **pivot set**. Our hypothesis is that a peer that keeps the same IP address for 10 days is assigned a static IP address. We then take the logs of the full crawls starting April 1st, 2007 to look for peers whose IP addresses are in the pivot set that have *changed their* KAD ID.

In figure 9, we plot the rate of change of KAD IDs for the peers in the pivot set. Up to now, all the publications on KAD assumed persistent KAD IDs. Our observations clearly disagree with this affirmation, since we see that a significant fraction of end-users in different countries change their KAD ID over time. The rate of change among the Chinese peers is highest with about 35% after *one month*, second is Spain with close to 20%, while the average is about 10%.

We have no good explanation for why end-users change their KAD IDs. In the case of China, we suspect that there exists a "Chinese implementation" of the KAD protocol that uses a new KAD ID for every session.
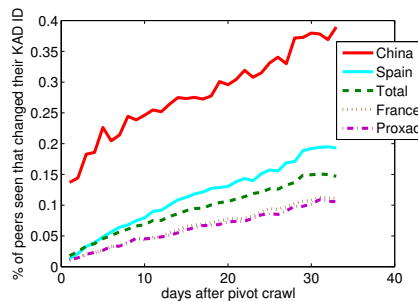

Figure 9: The fraction of peers in pivot set that changed their KAD ID at least once.

Implications of KAD ID Aliasing

The fact that the KAD ID assigned may be non-persistent obliges us to distinguish between a peer and an end-user:

- A **peer** is an instance of KAD identified by a fixed KAD ID.

- An **end-user** is a physical person that launches a peer to participate in KAD. The same end-user can, at different times, participate in KAD via different peers.

When KAD ID aliasing occurs, it is not really possible to characterize the lifetime of *end-users*, as compared to the lifetime of peers. All we can extract from

our crawl data is the lifetime of *peers*, which provides us with a lower bound on the lifetime of end-users.

6 Peer View

In this section, we will present metrics that describe the behavior of individual peers, such as lifetime, session and inter-session time, residual uptime, and daily availability using the observations made with our 179 day zone crawl.

We will mainly focus on the peers that were first seen on the 1st day of our crawl, since we could observe them for the longest period of time. For reference, we will occasionally compare the results with those obtained for the peers seen the first time on day 60. We have chosen day 60, since the largest inter-session times observed very rarely exceed 60 days, which allows us to assume that the peers we see for the first time on day 60 have newly joined the system.

6.1 Lifetime of Peers

We recall that, among the peers seen on the first day, about $2/3$ of the peers have a lifetime larger than one month and close to 45% have a lifetime even larger than three months (Table 1).

For a given KAD ID k, let $t_1^j(k)$ be the time this KAD ID is seen joining KAD for the first time, and let $t_m^l(k)$ be the time this KAD ID seen for the *last* time. The lifetime of KAD ID k is defined as $t_m^l(k) - t_1^j(k)$. Since our crawl is of finite duration, we can never be sure if a peer with KAD ID k will not come back after we stopped crawling. To make such a event very unlikely, we have decided to compute the lifetime only for peers with KAD IDs that have seen that last time 60 days or more before the end of our crawl (remember that the inter-session times seen are very rarely larger than 60 days!). Since at time $t_m^l(k)$ we do not know whether peer k will re-join KAD later, it is clear that our definition of lifetime gives a *lower* bound on the actual lifetime of a peer. Figure 10 depicts the the CCDF of lifetime for KAD IDs seen for the first time during the first crawl, on the first day, and for the first time on the 60th day. It is striking to notice that the KAD IDs first seen on day 60 have a *much lower* lifetime than the KAD IDs seen the first day. In fact, only 40% of the KAD IDs that were first seen on day 60 will be seen for more than one days (Figure 10(b)). As we know from table 1, more than half of the KAD IDs first seen on day 60 are from peers in China; it is for these peers that we could clearly establish that participants change their KAD IDs.

Figure 11 depicts the complementary cumulative distribution (CCDF) of the peers seen the the first day. There is a big difference in the lifetime of peers from China as compared to Europe: more than a third of the Chinese peers disappear after only one day and only 10% have a lifetime of more than 150 days, while close to 40% of the peers in Europe have a lifetime of more than 150 days.

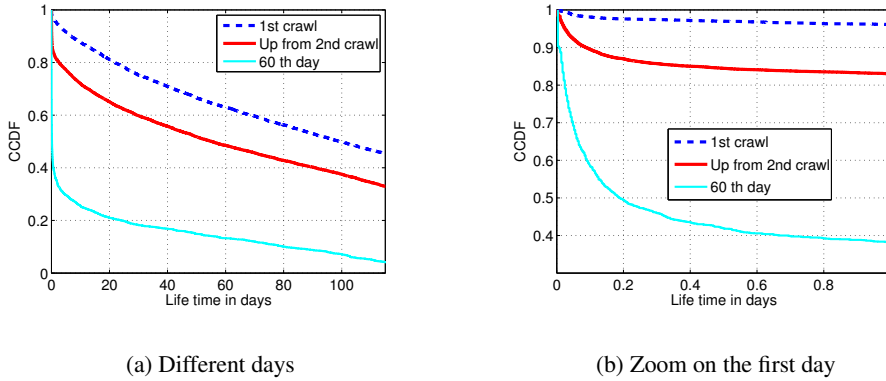(a) Different days   (b) Zoom on the first day

Figure 10: CCDF of the lifetime of peers seen during the 1st crawl, first day up from the second crawl, and on day 60.
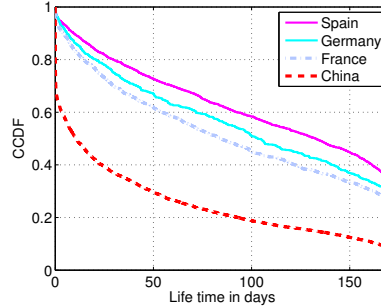

Figure 11: CCDF of the lifetime of the peers seen on the first day according to country of origin.

6.2 Session Statistics

Most of the peers will not be online, i.e. connected to KAD, all the time. By crawling KAD, we can determine for each peer k the instances $t_1^j(k), ..., t_n^j(k)$ when k joined and the instances $t_1^l(k), ..., t_m^l(k)$, with $m = n - 1$ or $m = n$, when k has left KAD. We define the **session length** as the time a peer was present in the system without any interruption, i.e. $t_i^l(k) - t_i^j(k)$ for $i \in \{1, ..., m\}$.

In figure 12 we plot the number of sessions per peer. Among the peers seen on the first day, 20% have more than 100 sessions.

In figure 13, we plot the distribution of the session length. The session length of the peers seen in the *first* crawl is about twice as high as the one of the peers seen for the first time during later crawls of day one. When we crawl KAD for the first time, we have a much higher chance to see peers that are connected "most of the time" than peers that are connected from time to time and only for short periods.
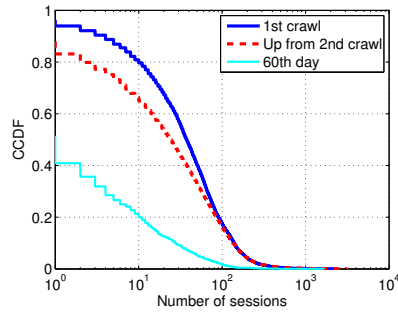
Figure 12: CCDF of the number of sessions per KAD ID.

This means that a single crawl of the system cannot give a representative picture of the characteristics of the peers: Instead, we need to sample the system many times.

For the peers seen in the first crawl, we observe session times (in minutes) with a mean = 670, standard deviation = 1741 and median = 155. For the peers seen during crawls 2-288 of the first day these values are only about half as large with mean = 266, standard deviation = 671 and median = 75. In both cases, the coefficient of variation, which is defined as the ratio between standard deviation and mean, and which is used to characterize the "variability" of a distribution, is between 2 and 3. In fact, the session lengths exhibit a considerable tail of large values (Figure 13(b)) with at least 0.1% of the sessions being longer than 1 week and the longest session observed being 78 days long.
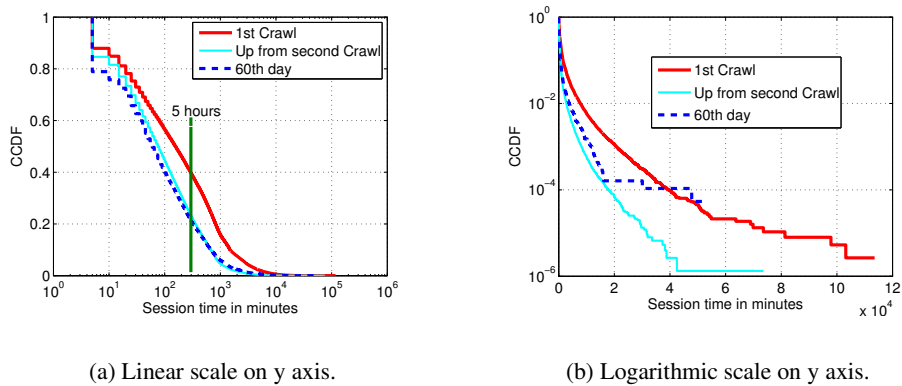

(a) Linear scale on y axis.

(b) Logarithmic scale on y axis.

Figure 13: CCDF of the session lengths per KAD ID for different peer sets.

Weibull fit of the session time distribution

We did a distribution fitting for the session times and found that the Weibull distribution provides a very good fit, as we can see in Figure 14. Visually, the Weibull

16

distribution adequately tracks the dominant shape of the measured distribution: Using only the session length samples larger than 15 minutes, the fit passes the Kolmogorov-Smirnov (goodness of fit) test. However, for the small session lengths of 5, 10, or 15 minutes, the fit is not good due to the too large granularity of time between two crawls (5 minutes).
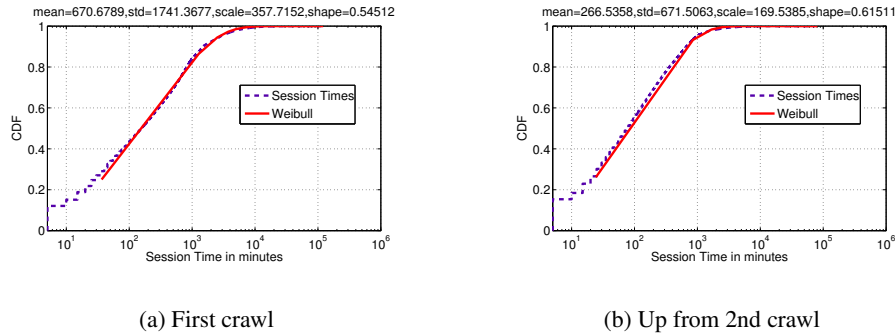

(a) First crawl (b) Up from 2nd crawl

Figure 14: Weibull fit of the session time distribution of the peers seen first day.

The Weibull distribution has two parameters $k > 0$ (*shape*) and $\lambda > 0$ (*scale*). For $k < 1$ the Weibull distribution is part of the class of the so called subexponential distributions, for which the tail decreases more slowly than any exponential tail [6]. This implies that knowing the past (uptime) of a peer allows to predict the future (residual uptime). More formally, if S denotes the session length then the expected residual uptime $\mathbb{E}[S - t | S > t] \sim O(t^{1-k})$, i.e. it grows *sub-linearly* as compared to the Pareto distribution, where the growth is linear, i.e. $O(t)$.

Stutzbach [20] observed that the Weibull distribution provides a good fit for the session lengths of the *BitTorrent* traces. However, due to significant under counting of long sessions in the KAD traces, Stutzbach was not able to determine which distribution best describes the session length of the KAD trace.

Figure 15 shows the expected residual uptime for the scale and shape values that describe the session length of peers seen in the first crawl. There is a nice fit between the empirical values and the interpolation using a function whose growth is $O(t^{1-k})$. We see that for small observed uptime values the remaining expected uptime values are considerable: A peer that has been up for 500 minutes will have a remaining expected uptime of 1,000 minutes.

One occasion where it would be interesting to exploit the knowledge that session lengths are Weibull distributed is in *dynamically adjusting the expiration times of the source keys* published: We saw that in section 3.2 that a source key that points to the peer will expire 5 hours after it has been published. On the other hand, we observed that the median session length of peers is 155 minutes or less. Also, figure 13(a) indicates that less than 40% of the peers have a session length of 5 hours or more. This means that in more than 60% of the cases the peer that publishes a source key will leave KAD before the pointer to the file it owns will expire. As a

result, many of the pointers to sources in KAD will be stale. A more appropriate policy might be to first publish a source key with an expiration time much smaller than 5 hours and progressively increase the expiration time as the uptime of the peer that owns the file increases.
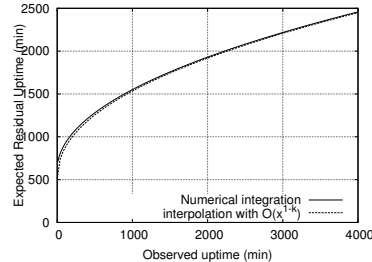

Figure 15: Expected residual uptime for $k = 0.54, \lambda = 357$.

Next Session Time

One may ask the question whether consecutive sessions are correlated in length. If there is a strong positive correlation, one could use information about past session lengths as predictor for the length of future sessions. Such a prediction could, for instance, be used by a publishing peer to choose the optimal value for the time during which information it publishes in KAD should be valid.

If we take all session length samples and compute the coefficient of correlation over consecutive session length we get a value of 0.15, which indicates that there is almost no correlation. For a visual depiction see figure 16(a). However, if one only considers session lengths up to 1 day, there is considerable positive correlation (correlation = 0.85) as can be also seen from figure 16(b). Stutzbach ([20] figure 10(b)), who could only observe session lengths up to one day found a strong correlation. This example nicely illustrates how incomplete data due to too short crawling duration can have a major impact on the conclusion one is able to draw from the observations.

6.3 Inter-Session Time

The **inter-session time** is defined as the time a peer k is continuously absent from the system, i.e. $t_{i+1}^j(k) - t_i^l(k)$ for $i \in \{1, ..., n\}$. Figure 17 depicts the CCDF of the inter-session times. As it was already the case for the session times, the distribution of the inter-session times of the peers seen during the first crawl is smaller than the one of the peers seen during later crawls. The same is true for the mean inter-session time (1110.2 min. vs. 1349.8 min.). The peers first seen on day 60, which mainly come from China, have even larger inter-session times and a mean inter-session time of 1704.3.
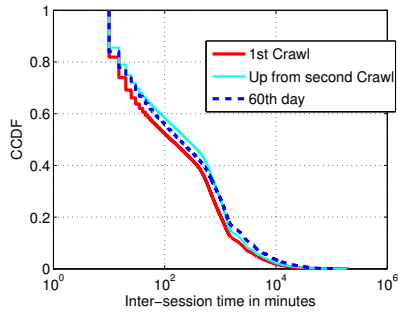
18
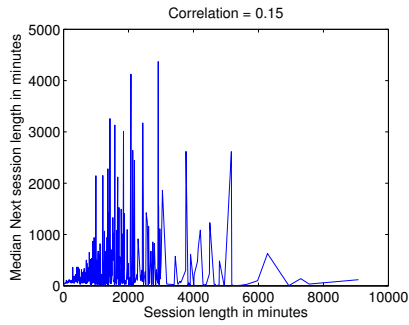
(a) All session lengths.
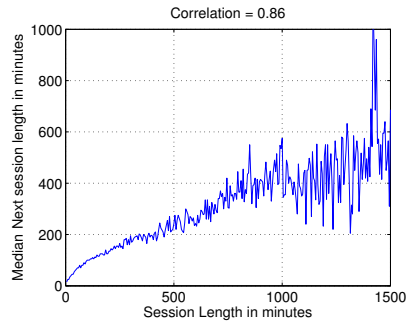
(b) Zoom on small values up to 1 day.

Figure 16: Correlation between consecutive session lengths of the peers seen.

For the inter-session times we could not find a distribution that matches well our observed data. In particular, the Weibull distribution did not fit.

(a) Empirical inter-session times of first crawl, of the first day up from the second crawl and of day 60.
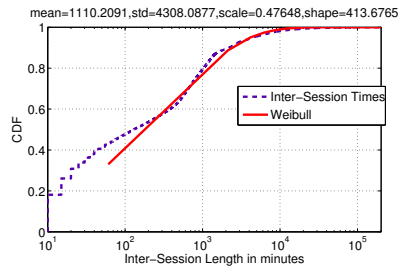
(b) Weibull fit of the Inter-session time for peers of first crawl

Figure 17: CCDF of the Inter-Session times of the peers.

6.4 Remaining Uptime

The eMule and aMule implementation do only publish on peers that have been up for *at least 2 hours*. Source keys will expire after 5 hours and keyword keys after 24 hours. We wonder if selecting a peer that has been up for at least 2 hours will increase the chances that this peer will be up for another 5 or 24 hours.

In figure 18, we plot the remaining uptime of peers given that they have already been up 5 minutes, 1h, 2h, or 8h. We see that on a short time scale of several hours, a higher uptime translates into a higher remaining uptime. However, on a longer

timescale of one day or more, the past uptime will not make much of a difference. This behavior was predicted by the expected residual uptime (Figure 15).

This means that minimum age-based peer selection as implemented in eMule and aMule is quite effective when publishing source keys that will expire after 5 hours, but not for keyword keys that will expire after one day. Also, only about 20% of the peers with an uptime of 2 hours will remain up for at least another 24 hours. Therefore, the only way to ensure that keywords remain available for 24 hours is to publish information about a keyword on *more than one peer*, as it is done by eMule and aMule.
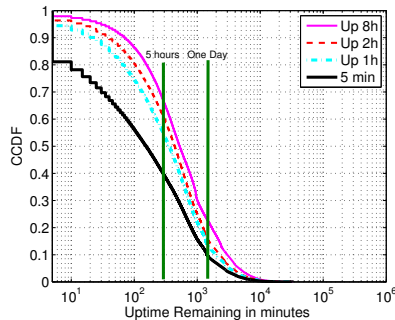

Figure 18: CCDF of the remaining uptime of peers, given the uptime so far, for peers seen in first crawl.

6.5 Availability

Characterizing availability is important for building efficient distributed applications such as overlay multicast or distributed file systems. For instance, availability-guided file placement can help reduce the cost of object maintenance [10], which can be potentially prohibitive, as it was pointed out by Blake [3]. The availability during the interval $[T, T + \Delta]$ of a KAD ID that was first seen at time T is defined as the sum of the times the KAD ID was seen during the interval $[T, T + \Delta]$ divided by the length of the interval Δ. This definition implies that a KAD ID k that has not been seen beyond time $t_m^l(k)$ will see its availability strictly decreasing for increasing values of Δ that satisfy $t_m^l(k) < T + \Delta$. For this reason,it we will later introduce a second notion of availability that considers only the period during which the KAD ID was observed.

Figure 19 depicts the CDF of the mean peer availability computed for different intervals Δ. As it has already been seen by Bhagwan [2], in a peer-to-peer system with churn, the mean availability decreases as the period over which availability is computed increases, which is due to the fact that some peers may have permanently left KAD. For large values of Δ, the availability keeps decreasing. However, for short values for Δ such as 1, 2, or 5 days, there is a significant fraction of peers (30%, 15%, 5%) that have an availability of one.
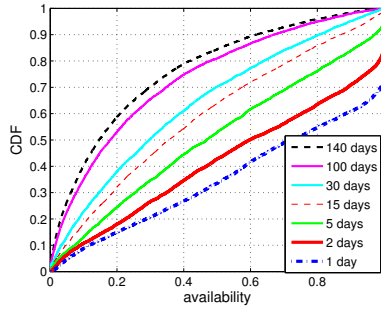
20

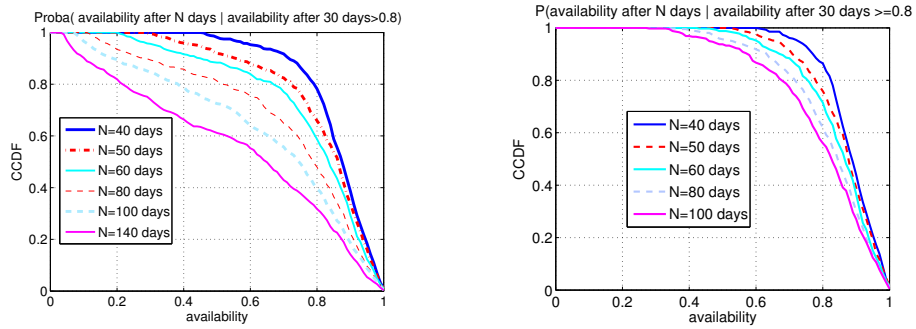Figure 19: CCDF The availability of peers seen on the first day.

Given that the availability of a peer over the last N days is known, we want to study how accurate the future availability of a peer can be predicted knowing its past behavior.

To do so we take the peers with an average availability larger than 0.8 during the first N days and ask how well the availability values from the past predict the future availability. In figure 25(a) we see that a high availability of 0.8 in the past 30 days will result in more the 60% of the cases in an at least as high availability for the next 30 days. As we increase the prediction horizon to $N = 140$ days the distribution of the future availability becomes almost uniform, which means that in a real system (with permanent departure), the long term availability at day $N = 140$ can not be predicted knowing the availability of the past 30 days. The set of peers considered in figure 25(a) contains peers that leave the system definitively before N days, i.e. their lifetime $< N$ days. However, as we have seen before, there is considerable KAD ID aliasing (see section 5.3), which implies that the actual end-user lifetime will in many cases be larger than measured the KAD ID lifetime. To get an "upper bound" on how well we could predict end-user availability in the best case, when there are no permanent departures, we consider in figure 25(b) only the peers whose lifetime is at least 100 days. In this case, a high availability of 0.8 in the past 30 days will result in more than 75% of the cases in an at least as high availability for the next 30 days and in 60% of the case even for the next 50 days ($N = 80$).

6.6 Daily Availability

Daily availability measures the fraction of time a peer is connected per day. Daily availability expresses the "intensity" of participation of users in the exchange of files. For a given peer P, we define **daily availability** of P as the percentage of time P was seen on that day. For a peer that was first seen on day i and last seen on day j, we will get a time series of daily availability values that has $j - i + 1$ elements.

Peers in China spend much less time per day connected than peers in Europe

(a) All peers seen first day.

(b) Peers seen first day with a lifetime $\geq$ 100 days.

Figure 20: Conditional distribution of the availability over N days, given the availability over the first 30 days larger than 0.8.

(Figure 21). The "online times" for peers in Europe are quite impressive, with 40% of the peers being connected more than 5 hours per day and 20% even more than 10 hours per day.
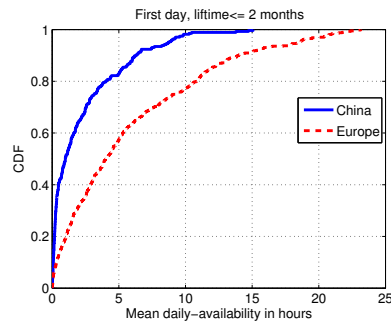

Figure 21: CDF of the mean daily availability of peers seen the first day.

Next day availability

Daily availability measures how many hours a peer is connected per day. Observing this metric over several days or weeks can give an indication about the stability of peer participation in KAD over time. Stutzbach ([20], figure (11b)) compared the daily availability of peers between the 2 days of his crawl. For all peers with a given availability value on day one, he computed the *median* availability of these peers on day 2 of the crawl. A plot of the availability on the first days vs. median availability on the second day indicated that both values as positively correlated. We did

a similar plot (Figure 22(a)) and obtained a coefficient of correlation of 0.81. However, we are not sure if taking the median availability of the peers with the same availability on the first day is the right way to check for correlation. Figure 22(b) shows a scatter plot of the availalibility values of each peer for two consecutive days. We see that for a given availability on the first day the availability of these peers on the second day can *vary widely*, which also results in a lower coefficient of correlation of 0.52.

(a) First day vs. *Median* next day availability

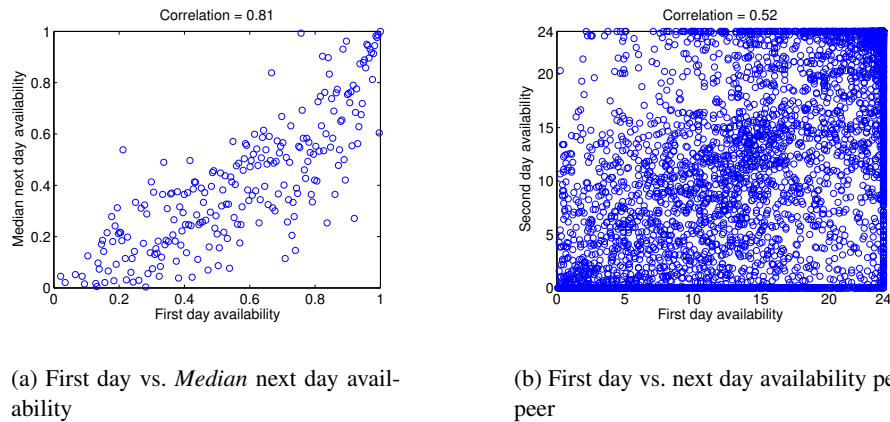(b) First day vs. next day availability per peer

Figure 22: Scatter plot of the availability on the first day vs the availability of the second day for peers seen in the first crawl.

Figure 23 plots the daily availability timeseries for a random set of peers over a duration of 100 days. While there are a few peers for which the daily availability changes little over time, most of the peers exhibit daily availability values that vary a lot.
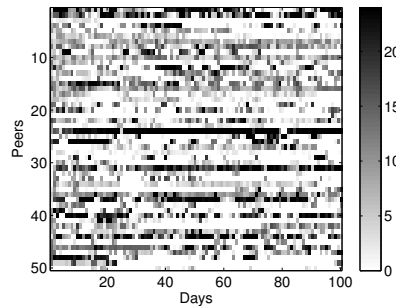

Figure 23: Daily availability in hours (see Section 6.6) of 50 random chosen peers of the first crawl.

To formally quantify the daily availability patterns of peers, we use a metric

called **approximate entropy** ($ApEn$), which is a "regularity statistic" that quantifies the unpredictability of fluctuations in a time series. For details on the approximate entropy see [15]. This metric was recently used by Mickens [10] to analyze several traces of machine availabilities such as the Microsoft trace and the Overnet trace. We calculate $ApEn$ of the daily availability of KAD peers seen on the first day. The smaller the value for $ApEn$, the more regular the daily availability pattern over time. However, if the time series is highly irregular, the occurrence of similar availability patterns will be very unlikely for the following days, and $ApEn$ will be relatively large.

We could confirm the results of Mickens for the Microsoft trace (see figure 24(a)), where 80% of the values of $ApEn(2)$ are close to zero, which indicates that daily availability varies little over time. On the other hand, the $ApEn(2)$ values for the KAD trace a much higher (see figure 24(b)). About 50% of the peers the peers have $ApEn(2)$ values above 0.5, which indicates that the daily availability values are quite irregular. Mickens made a similar observation for the Overnet trace.

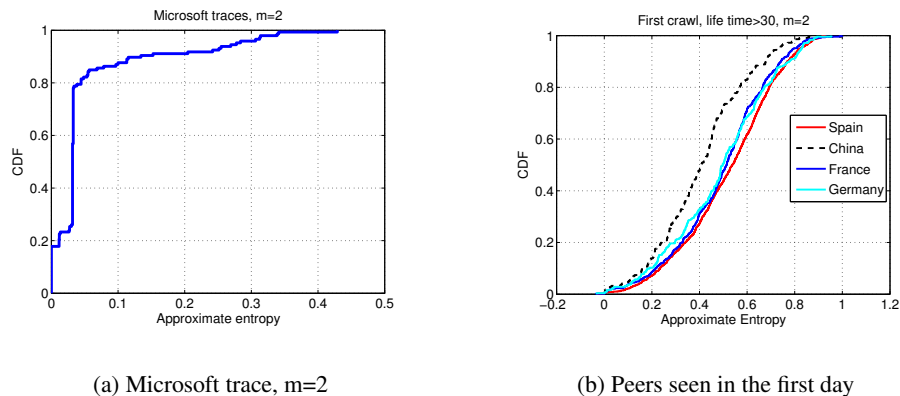(a) Microsoft trace, m=2   (b) Peers seen in the first day

Figure 24: CDF of the approximate entropy.

We also checked if the daily availability behavior of the peers exhibits any diurnal pattern. Douceur[4] analyzed different traces of machine availability[2] from Microsoft, Internet, Gnutella, and Napster. Using Fourier transformation he found cyclic behavior in the daily availability of the Microsoft machines, but did not find any diurnal patterns for the other traces. We did a Fourrier and Wavelet transformation on the daily availability timeseries of our KAD peers and could not find any cyclic behavior or diurnal patterns.

---

[2]The trace is available at
`http://www.cs.berkeley.edu/˜pbg/availability/`

6.7 Availability Prediction

Given that the availably of a peer over the last N days is known, we want to study how accurate the future availability of a peer can be predicted knowing it's past behavior. In all the experiments we consider the peers that were seen the first day of the crawl or a subset of these peers.

We start with a very simple experiment, where we only take the peers with an average availability larger than 0.8 during the first N days and ask how well the availability values from the past predict the future availability.

In figure 25(a) we see that a high availability of 0.8 in the past 30 days will result in more the 60 % of the cases in an at least as high availability for the next 30 days. As we increase the prediction horizon to $N = 140$ days the distribution of the future availability becomes almost uniform, which means that in a real system (with permanent departure), the long term availability at day $N = 140$ can't be predicted knowing the availability of the past 30 days. The set of peers considered in figure 25(a) contains peers that leave the system definitively before N days, i.e. their lifetime $< N$ days. However, as we have seen before, there is considerable KAD ID aliasing, which implies that the actual end-user lifetime will in many cases be larger than measured the KAD ID lifetime. To get an "upper bound" on how well we could predict end-user availability in the best case, when there are no permanent departures, we consider in figure 25(b) only the peers whose lifetime is at least 100 days. In this case, a high availability of 0.8 in the past 30 days will result in more the 75% of the cases in an at least as high availability for the next 30 days and in 60% of the case even for the next 50 days ($N = 80$).
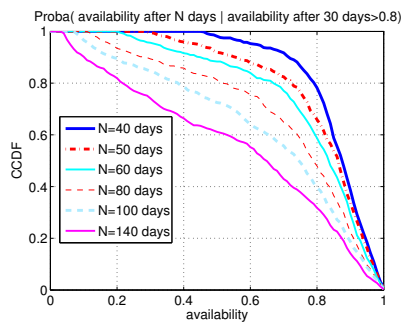
(a) All peers seen first day.
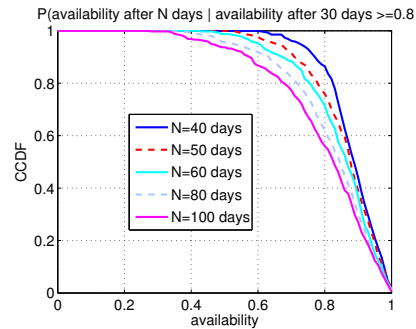
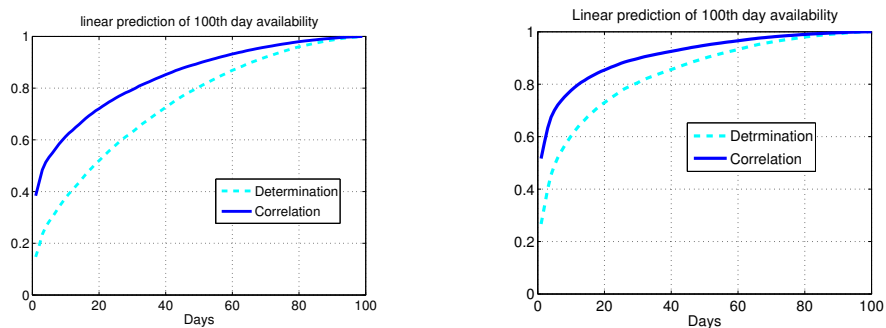(b) Peers seen first day with a lifetime $\geq$ 100 days.

Figure 25: Conditional distribution of the availability over N days, given the availability over the first 30 days larger than 0.8.

We next want to see if it is possible to *linearly predict* the availability of peers. Knowing the past availability of a peer for N days, who well can we predict its availability at day 100. To assess the quality of the linear prediction we have two

measures:

- The **correlation** coefficient indicates the strength and direction of a linear relationship between two random variables. The correlation is 1 in the case of an increasing linear relationship, -1 in the case of a decreasing linear relationship, and some value in between in all other cases, indicating the degree of linear dependence between the variables. The closer the coefficient is to either -1 or 1, the stronger the correlation between the variables. However, correlation alone is be sufficient to evaluate this relationship between two variables.

- The **coefficient of determination**, R^2, measures the goodness of the global fit of the model. Specifically, R^2 takes a value in [0,1] that represents the proportion of variability in Y_i that may be attributed to some linear combination of the regressors (explanatory variables) in X. To consider the prediction to be good if $R^2 \geq 0.9$ Thus, $R^2 = 1$ indicates that the fitted model explains all variability in y, while $R^2 = 0$ indicates no *linear* relationship between the response variable and regressors.

As before we use all the peers seen the first day to get a "lower bound" on how well linear prediction works (see figure 26(a)) and all the peers seen the first day whose lifetime is greater than 100 days to get an "upper bound" (see figure 26(b)). To have a good prediction of the availability, we need to know at least the availability over the first 70 days, if we take all peers, and over the first 50 days, if we take only the peers that stay for at least 100 days.

(a) All peers seen in the first Crawl (5669 peers)

(b) Peers seen in the first crawl with lifetime$\geq$ 100 days (2823 peers)

Figure 26: Linear predictability of the availability over 100 days given the availability over the first N days.

7 Conclusion and Future Work

We have presented results on the peer behavior in KAD, the largest currently deployed DHT. The duration of our crawl was 179 days, which makes it to our knowledge, the longest crawl of a peer-to-peer system ever carried out.

The speed of our crawler allowed us to crawl the entire KAD systems and the results we obtained could be used to validate our approach to crawl in the following mostly a single zone whose results will be representative for the entire system.

The most important findings are that session lengths are Weibull distributed, and that session length and daily availability varies a lot for a given peer. Also, KAD IDs are not necessarily persistent as was assumed so far. Nevertheless, the most important metrics such as session times, inter-session times or daily availability are not affected by the non-persistent KAD IDs.

It remains an open problem to explain why KAD IDs are non-persistent and under what circumstances peers change their KAD ID. To be able to model the lifetime of peers we need to find a method that allows to characterize the process of permanent departure for *end-users* as opposed to peers.

This paper contributes to a better quantitative understanding of the peer dynamics in KAD. However, the current implementation of KAD in eMule and eMule does not yet exploit this knowledge. Instead, some important parameters such as the expiration time of keys or the number of copies are static. While we have already commented on some of the parameter choices in the paper. We feel that our work opens numereous interesting perspectives for improving the design and implementation of KAD.

References

[1] A-Mule. `http://www.amule.org/`.

[2] R. Bhagwan, S. Savage, and G. Voelker. Understanding availability. In *Proceedings of the 2nd International Workshop on Peer-to-Peer Systems (IPTPS'03)*, pages 256–267, 2003.

[3] C. Blake and R. Rodrigues. High availability, scalable storage, dynamic peer networks: Pick two. In *HotOs 03*, 03.

[4] J. Douceur. Is remote host availabilty governed by a universal law. In *Sigmetrics Performance Evaluation Review*, 2003.

[5] E-Mule. `http://www.emule-project.net/`.

[6] C. Goldie and C. Klueppelberg. Subexponential distributions. In R. Adler, R. Feldman, and M. Taqqu, editors, *A Practical Guide to Heavy Tails: Statistical Techniques for Analysing Heavy Tails*. Birkhauser, Basel, 1997.

[7] K. Kutzner and T. Fuhrmann. Measuring large overlay networks - the overnet example. In *Proceedings of the 14th KiVS*, Feb. 2005.

[8] Maxmind. `http://www.maxmind.com/`.

[9] P. Maymounkov and D. Mazieres. Kademlia: A Peer-to-peer informatiion system based on the XOR metric. In *Proceedings of the 1st International Workshop on Peer-to-Peer Systems (IPTPS)*, pages 53–65, Mar. 2002.

[10] J. Mickens and B. Noble. Exploiting availability prediction in distributed systems. In *Proc. 2006 NSDI*, 2006.

[11] I. Mitrani. *Probabilistic Modelling*. Cambridge University Press, 1998.

[12] M. Mitzenmacher and E. Upfal. *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge Press, 2005.

[13] N. Naoumov and K. Ross. Exploiting p2p systems for ddos attacks. In *International Workshop on Peer-to-Peer Information Management*, May 2006.

[14] Overnet. `http://www.overnet.org/`.

[15] S. M. Pincus. Approximate entropy as a mesure of system complexity. *In Proceedings of the National Academy of Science*, 88:2297–2301, December 1990.

[16] S. Ratnasamy, M. Handley, R. Karp, and S. Shenker. A scalable content-addressable network. In *Proc. ACM SIGCOMM*, 2001.

[17] A. Rowstron and P. Druschel. Pastry: Scalable, distributed object location and routing for large-scale Peer-to-peer systems. In *Proceedings of Middleware*, Heidelberg, Germany, November 2001.

[18] I. Stoica, R. Morris, D. Karger, M. Kaashoek, and H. Balakrishnan. Chord: A scalable Peer-to-peer lookup service for Internet applications. In *Proceedings of SIGCOMM*, pages 149–160, San Diego, CA, USA, 2001. ACM Press.

[19] D. Stutzbach and R. Rejaie. Improving lookup performance over a widely-deployed DHT. In *Proc. Infocom 06*, Apr. 2006.

[20] D. Stutzbach and R. Rejaie. Understanding churn in peer-to-peer networks. In *Proc. Internet Measurement Conference (IMC)*, Oct. 2006.