[HLDBMSt05] *Oliver Heckmann, Nicolas Liebau, Vasilios Darlagiannis, Axel Bock, Andreas Mauthe, Ralf Steinmetz;* **From Integrated Publication and Information Systems to Information and Knowledge Environments: Essays Dedicated to Erich J. Neuhold on the Occasion of His 65th Birthday;** Lecture Notes in Computer Science, chapter A Peer-to-Peer Content Distribution Network, Springer-Verlag GmbH, January 2005, S. 69-78.

A Peer-to-Peer Content Distribution Network

Oliver Heckmann, Nicolas Liebau, Vasilios Darlagiannis,
Axel Bock, Andreas Mauthe, and Ralf Steinmetz

Multimedia Kommunikation (KOM)
Technische Universitt Darmstadt
Merckstr. 25, 64293 Darmstadt
[heckmann, liebau, bdarla,
bock, mauthe, steinmetz]@kom.tu-darmstadt.de

Abstract. The distribution of large content files, like videos, over a large number of users is a demanding and costly operation if done using a traditional client/server architecture. Peer-to-peer based file-sharing systems can be used as an alternative for content distribution.
The eDonkey file-sharing network is one of the most successful peer-to-peer file-sharing networks, especially in Germany. eDonkey forms a hybrid network that capitalizes both on the client/server and peer-to-peer paradigms in the design of its architecture.
In this paper, we describe the eDonkey protocol, the constructed overlay network, the critical operations and their characteristics, as well as the results of measurements of the network and transport layer and of the user behavior. The measurements were made with the client software and with an open-source eDonkey server we extended explicitly for these measurements. Our study shows that eDonkey is particularly well suited for content distribution and not surprisingly also intensively used for the distribution of large files, mainly videos.

1 Introduction

Content distribution realized either as inelastic media streams or as elastic downloadable files over a large number of users is a challenging and demanding service. In an idealistic solution, both the transport network and the deployed applications should collaborate to achieve an optimally operating service. However, with the currently best-effort deployed service of the Internet, the support of the network is limited to very basic delivery operations. In order to realize content distribution over a large number of users, application developers typically design virtual networks at the Application Layer, the so-called *overlay networks*.

A significant number of applications that employ overlay networks to interconnect a large number of users have been deployed and heavily operating lately over the Internet. Such applications harvest the resources of the end-systems and are called peer-to-peer (P2P) applications. To understand the influence of these applications and the characteristics of the traffic they produce as well as their impact on network design, capacity expansion, traffic engineering and shaping, it is important to empirically analyze the dominant file-sharing applications.

The eDonkey file-sharing protocol is one of these file-sharing protocols. It is used by the original eDonkey2000 client [eDonkey] and additionally by some advanced open-source clients like mldonkey [mlDonkey] and eMule [eMule]. According to [San03], the eDonkey protocol is the most successful P2P file-sharing protocol in Germany (52% of the generated file-sharing traffic). It is more successful than the FastTrack [FTrack] protocol used by KaZaa [KaZaa] (that contributes to 44% of the traffic). Contrary to other P2P file-sharing applications, such as Gnutella [Gnut], that are widely discussed in literature, the eDonkey protocol is not well analyzed yet. In this work, we shed light on this protocol and its suitability for content distribution with a measurement study of the observed traffic, the user behavior, and the topological characteristics of the constructed overlay network.

In the next section, we give an overview of related work. Then, in Section 3 we describe the eDonkey protocol. In Section 4, our measurements are presented. They were gathered with the eMule client software and with an open-source eDonkey server we extended for the purpose of this paper. We also present the results of the measurements before we give a summary and draw the conclusions in Section 5.

2 Related Work

P2P file sharing applications received a considerable attention after the wide deployment of Napster [Napster]. Though Napster did not follow a pure P2P architecture since it required centralized infrastructure for indexing of published documents, file exchange itself was taking place in a P2P manner. However, Napster was mostly appropriate for exchanging relatively small files such as compressed music files (mp3) since each file could be requested and downloaded from a single user only. Legal issues caused by the exchange of copyright protected content forced Napster to cease its operation, mainly due to Napster's centralized infrastructure. However, the Napster protocol is still active and utilized by OpenNap, an open-source implementation of the Napster network. Here, instead of one single server location, multiple servers exist to help coordinate the search queries of connected users. While the network is still structured around central servers, the abundance of them would make it harder to target than the original Napster. OpenNap is still targeting on music files.

Gnutella [Gnut] became the most popular P2P file sharing application after Napster's operation discontinued. Gnutella was the first file sharing application with pure P2P architecture deployed in such a large scale. However, the flooding mechanism, which was used for searching, made its operation very costly in terms of network overhead. Again, Gnutella users were focusing on small music files, however, video exchange started to appear. Gnutella's exchange mechanisms did not allow for downloading from multiple peers in parallel, so large content files exchange could not be efficiently supported. In addition, the aforementioned high network overhead motivated the design of a new architecture that used

UltraPeers [Gnut2] to perform common overlay operations such as indexing and searching.

The KaZaa [KaZaa] client followed Gnutella in popularity almost synchronously with eDonkey2000. KaZaa uses the FastTrack [FTrack] protocol for its overlay operations. Its architecture introduces the term of "super-peers" to handle the common overlay operations. Peers are dynamically assigned the role of a super-peer. KaZaA enables parallel downloads of the same file from multiple users, so increasing its suitability to exchange large content files. However, KaZaa does not make optimal use of the available content since users can download only from remote peers that have complete copies of the files. Incomplete ones are simply ignored.

BitTorrent [BitTor] is a unique file distribution system different then traditional P2P systems. BitTorrent is used to distribute large files and its operation is as follows. Initially, peers must share and host the published source files. Other peers may start downloading from the original source, each a different part of the file. Shortly afterwards, they start downloading the parts they do not have from each other. The file transfer is therefore spread and distributed over the total number of peers downloading the file, who are forced to upload as well[1]. BitTorrent, however, is not a general purpose P2P network and its searching capabilities are not sufficient.

There is a large number of additional P2P file sharing applications with different characteristics. For example, MojoNation [Wil02] was an attempt to enforce micro-payment mechanisms to provide incentives for users to share documents and avoid free-riders. MNET [Mnet] is MojoNation's successor with reduced complexity. FreeNet [CSWH00] is another file sharing network that provides a significant degree of anonymity for anti-censorship reasons. DirectConnect [DirCon] follows a Hub-based architecture. A Hub is a server with indexing and searching capabilities which brings users with similar interests together.

In addition to the media exchange as a file downloading approach, there is an attempt lately to address streaming of media in Internet-deployed P2P applications. A significant effort is the Mercora [Mercora] network that supports real time audio streaming. Also, Skype [Skype] is a free IP-Telephony application that became very popular because of the good quality of speech encoders and the relatively small end-to-end delay. Skype uses FastTrack (such as KaZaa do) to find remote users and media transfer takes place directly between the endpoints (multiparty teleconferences are supported as well).

3 The eDonkey Network

The eDonkey network is a decentralized hybrid peer-to-peer file-sharing network with client applications running on the end-systems that are connected to a distributed network of dedicated servers.

[1] This same mechanism is implemented in the eDonkey2000 P2P network (and one of the reasons why eDonkey is so efficient at distributing large files).

Contrary to the original Gnutella protocol, it is not completely decentralized as it uses servers; contrary to the original Napster protocol it does not use a single server (or a farm of them), which is a single point of failure. Instead it uses servers that are run by power users and offers mechanisms for inter-server communication. Unlike super-peer protocols like KaZaa, or the modern UltraPeer-based Gnutella protocol, the eDonkey network has a dedicated client/server based structure. The servers are slightly similar to the KaZaa super-nodes, but they are a separate application and do not share any files. Their role is to manage the information distribution and to operate as several central dictionaries, which hold the information about the shared files and their respective client locations.

In the eDonkey network the clients are the only nodes sharing data. Their files are indexed by the servers. If a client wants to download a file or a part of a file, it first has to connect via TCP to a server or send a short search request via UDP to one or more servers to get the necessary information about other clients offering that file. Figure 1 illustrates the eDonkey network structure.

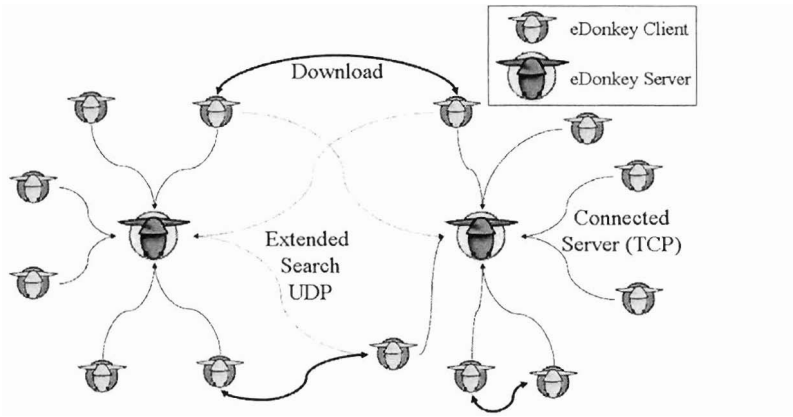


Fig. 1. eDonkey Network Structure

The eDonkey network uses 16 byte MD4 hashes to (with very high probability) uniquely identify a file independent of its filename. The implication for searching is that two steps are necessary before a file can be downloaded. First, a full text search is made at a server for the filename. The corresponding reply includes those file hashes that have an associated filename which matches the full text search. In a second step, the client requests the sources from the server for a certain file-hash. Finally, the client connects to some of these sources to download the file. File transfer takes place directly among the participating clients without involving the server entities to minimize the cost of the operation.

The eDonkey protocol supports the download of a single file from multiple sources. This can improve the download speed dramatically and reduces the risk

of unfinished downloads. Files are downloaded in chunks of 9MB. Files that are downloaded even to a part are already shared by a client and can be downloaded by other clients. The MD4 hash allows to identify whether a certain shared file is the requested file or not. As individual chunks are also hashed, corrupted chunks can be identified. All these properties make the eDonkey protocol well suited for sharing large files like video files. Our measurement study below verifies that it is also being used for that.

4 Measurements

4.1 Experiments Setup

We ran two types of experiments to analyze the eDonkey protocol and to collect traffic samples:

In the first experiment, we measured the traffic at two clients connected to 768/128 kbps ADSL connection and a 10 MBit/sec at the TU Darmstadt, respectively. The most popular client of the network, eMule [eMule] under Windows XP, was used for this experiment. From the measurements, we derive traffic characteristics such as the protocol overhead, downloading behavior and message size distributions.

For the second experiment, we modified an open-source eDonkey server [Lus03] that was connected to the university network collecting application level information about the usage of the network, requested files, offered files and the overall size of the network[2].

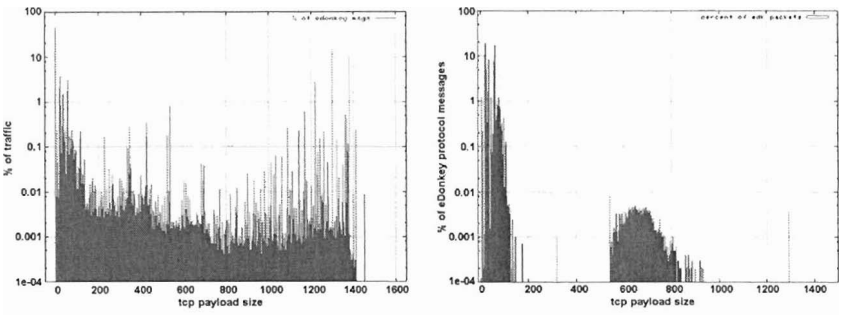
A more detailed protocol analysis based on measurements with TCPDump can be found in our technical report [HB02].

4.2 Results

TCP/UDP summary. For the clients-focused experiments, the share of the TCP traffic of the overall network traffic was 94% (number of packets) or 99.8% (of the payload size), respectively. Those values do not differ significantly between ADSL and broadband connections. It should be noted that all TCP ACK packets with zero payload were counted, too. On the ADSL line the packet loss was approximately 5.5%, about 2.25 times higher than on the broadband line (2%).

The server-side measurements differ: TCP traffic only forms about 2,4% of the packets and 6% of the payload. The server is mainly busy with handling UDP requests from clients *not* directly connected: If a search of a client does not deliver enough results from the server it is connected to, further known servers can be searched. This behavior has implications on dial-up connections. If the dial-up IP address was assigned to an operating eDonkey server before, many clients all over the world still reference to that IP address as a server. Server

[2] The server proved to be quite popular and due to the sheer amount of flows was ranked on the second place of the traffic statistic of the whole TU Darmstadt network for the duration of the experiment.



(a) Total TCP packet size distribution (b) Packet size distribution of protocol messages

Fig. 2. eDonkey traffic statistics

entries are typically kept 1 day or longer in most client implementations. These clients will still send UDP based search queries to that IP address consuming bandwidth[3] until the expiration of the corresponding entry.

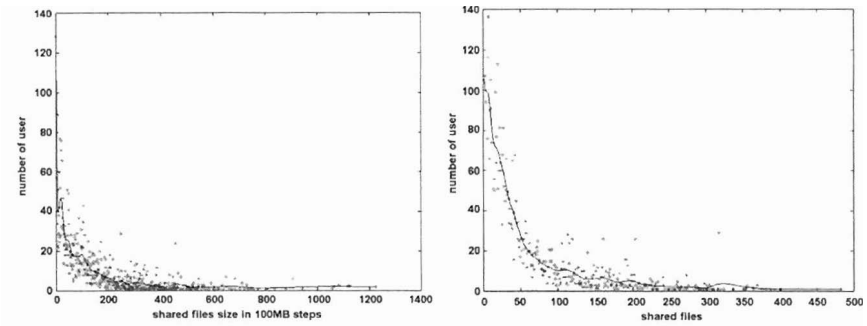
The TCP packet sizes range greatly from header only to MTU size (see Figure 2 a). Some characteristic peaks can be identified, e.g., for TCP messages of payload size 24, which are typically used for the frequent *QUERY SOURCES* messages. In most cases a TCP packet carries only a single protocol message. Figure 2 shows the TCP packet size distribution for eDonkey protocol messages. Protocol messages can be identified quite reliably as the payload starts with "E3" (for details see [HB02]). The UDP packets are much less variable in size, the size clearly indicating which kind of message is transported.

Protocol Messages. Looking at client's UDP traffic, the most observed protocol message type is by far *QUERY SOURCES* (approximately 65%). On the other hand, TCP messages are distributed far more evenly where all common message types have a percentaged share between one and ten percent. The only surprising exception is TCP *QUERY SOURCES* message type, which appeared approximately 0.008% only.

On the server side the most frequently seen messages types is the *QUERY SOURCES* (approximately 36% for (TCP) and 95% for (UDP)).

Throughput. On the eMule/ADSL and eMule/broadband clients we observed an average overall throughput of 30 kB/s respectively 45 kB/s while downloading, which decreased to the preset maximum upload bandwidth (10 kB/s in our case) after downloads were finished. UDP traffic is not very important in the client scenario.

[3] This even led to a permanent entry in the FAQ section of the computer magazine c't [ct04].



(a) Size of shared files by number of users (b) Number of shared files by number of users

Fig. 3. User Statistics

On the server side, the UDP in/out ratio was 16/2 kB/s after roughly a day (so we have 18 kB/s overall throughput), while TCP was about 0.75/0.25 kB/s after the same time. In the starting phase though, TCP throughput was approximately 1.5/1 kB/s. As the server popularity was increasing, the UDP throughput was increasing massively and the TCP throughput was decreasing significantly thus, suppressing it to the aforementioned rates.

Connection Statistics. We observed about 100 (eMule broadband), 85 (eMule ADSL) and 150 (server) connection requests per minute. The share of connections actually used for data exchange was 77%, 74% and 72%. The number of simultaneous connections was 30 to 50 for eMule/broadband, 30 to 45 for eMule/ADSL and approximately 700 for the server (gathered by TCP trace file analysis) (see Figure 5 (a)).

For getting the average bandwidth use per connection, we looked at all connections carrying more than 0.5 MB incoming payload. Of those the great majority (several hundred) utilized roughly two to four kB/s and some of them a higher bandwidth (about 40) ranging from 5 to 10 kB/s. About ten connections utilized bandwidth rates up to 55 KB/sec. Those values are connection independent.

Our measurements show that an average of roughly four MBytes of data was transferred per (TCP) connection. The maximum size transferred we encountered was 150 MBytes. The average TCP connection time was 30 minutes while the average idle time 875 seconds (for ADSL) and 2500 seconds (for broadband).

User Statistics. We concentrated on getting information about the amount of files the users share in terms of size (see Figure 3 a) and in terms of file numbers (see Figure 3 b), as well as what they search most (indicated by search terms and identified files, see Figure 4). Our research showed that the average eDonkey user shares 57.8 (max. 470) files with an average size of 217 MB (max 1.2 TByte).

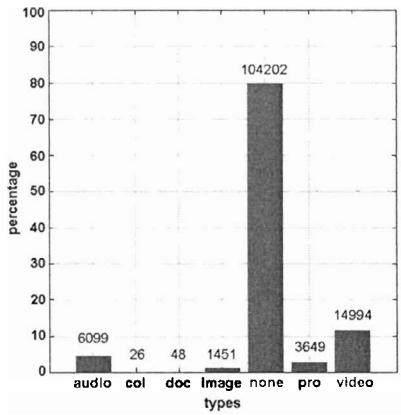


Fig. 4. Searched file types

Most searched-for keywords were media-related words like "MP3", "AVI" and "DVD" and a very high share of clearly recognizable words from current blockbuster movie titles.

Because all files are identified by their unique hash values we were able to identify the most wanted files by analyzing *QUERY SOURCES* messages. The corresponding filenames were extracted from the *PUBLISH FILES* messages. This showed that the vast majority of the identified files were movies just played in the cinemas at the time of the experiments.

Looking at these results we can state that eDonkey is mainly a movie distribution network.

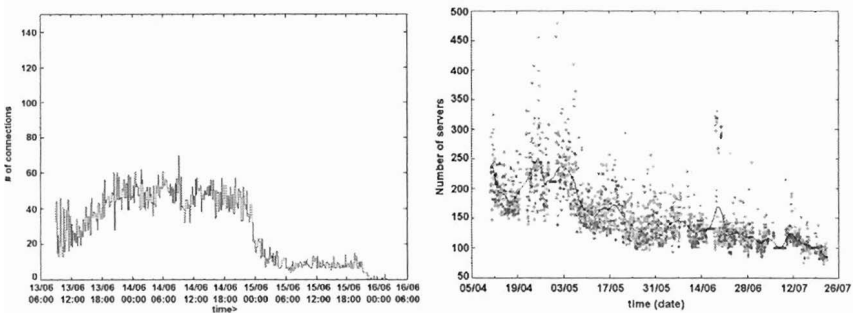
Geographical Analysis and Network Size. The IP Addresses of all the clients connected to the broadband test client were reverse resolved to lookup their top-level domains for a rough geographical overview. Table 1 shows the results.

Region	Number of addresses	Percentage(%)
.de	25,559	66.21%
.net	3,939	10.20
.com	2,038	5.28
.fr	2,491	6.45
.at	367	0.95
.br	365	0.95
.ch	562	1.46
.es	972	2.52
.il	470	1.22
.it	455	1.18

Table 1. Clients by region (Top 10)

The .de domain dominates with 66.21%, followed by the "dot-net"-Domains (10%), then .fr (6%) and .at (1%). There is a tend towards higher interconnectivity among clients from a certain region, because they exchange movies in their spoken language. This is also an additional indication why the eDonkey network is more popular in Germany than in other countries. The latter is also supported by the study in [San03].

To estimate the network size we monitored the number of servers in a serverlist provided by "ocbmaurice" [Ocb04]. This showed that the average size shrunk down from roughly 220 servers in beginning of 2002, over 100 servers in 2003 down to 47 servers in 2004 (see Figure 5 (b)).



(a) Simultaneous connections of clients (b) Development of network size

Fig. 5. Network statistics

This indicates that the eDonkey network as a whole is shrinking. One reason could be the release of the serverless eDonkey network called Overnet. Moreover, since the end of 2002 some people operating an eDonkey server faced accusations by authorities, since the exchanged material was copyrighted[4].

5 Conclusions

In this chapter, we presented results from two measurement experiments in the eDonkey network, which is the most successful peer-to-peer network in Germany. A protocol analysis shows that eDonkey supports parallel downloads by identifying files with hashes instead of file names and separates the full text search for files from the search for sources for actual downloads. These features are very useful for using eDonkey for the distribution of large content files like video files. In fact, our measurements show that the eDonkey network is primarily used for distributing movies. The ten most frequent downloads in our experiments were all current movie blockbusters. With respect to the generated network traffic,

[4] For example the operator of the Danish site http://www.siffan.dk

most traffic is caused by long-lived TCP connections and therefore very different from the web traffic, which is mostly short-lived TCP traffic and was dominating the Internet traffic only a few years ago.

References

[BitTor] BitTorrent Homepage. http://bittorrent.com/

[CSWH00] Ian Clarke, Oskar Sandberg, Brandon Wiley and Theodore W. Hong. *Freenet: A Distributed Anonymous Information Storage and Retrieval System.* ICSI Workshop on Design Issues in Anonymity and Unobservability, 2000, Berkeley, CA, USA

[DirCon] DirectConnect Website. http://www.neo-modus.com/

[DirCon++] DirectConnect++ Website. http://dcplusplus.sourceforge.net/

[FTrack] FastTrack Website. http://www.fasttrack.nu

[Gnut] Gnutella Website. http://www.gnutella.com

[Gnut2] Gnutella 2 Website. http://www.gnutella2.com

[Wil02] Bryce Wilcox-O'Hearn. *Experiences Deploying a Large-Scale Emergent Network.* Proceedings of the 1st International Workshop on Peer-to-Peer Systems (IPTPS02), March 2002

[Mnet] MNET Website. http://mnet.sourceforge.net/

[Skype] Skype Website. http://www.skype.com/

[Mercora] Mercora Website http://www.mercora.com/

[Napster] Napster Website http://www.napster.com/

[San03] **Sandvine Incorporated.** ***Regional Characteristics*** *of P2P - File Sharing as a Multi-Application, Multinational Phenomenon.* White Paper, 2003.

[eDonkey] eDonkey2000 Original Client Homepage. http://www.edonkey2000.com.

[eMule] eMule Project Homepage. http://www.emule-project.net.

[mlDonkey] mlDonkey Homepage. http://savannah.nongnu.org/projects/mldonkey/.

[KaZaa] Sharman Networks. *The KaZaa application for the FastTrack network.* Official Homepage. http://www.kazaa.com.

[Kli02] Alexey Klimkin. *pDonkey, an eDonkey Protocol Library.* http://pdonkey.sourceforge.net.

[Lug04] "Lugdunum". *Building an Efficient eDonkey Server on Linux/FreeBSD/Win32.* http://lugdunum2k.free.fr/kiten.html 2004.

[Lus03] Thomas Lussnig. *cDonkey, an Open-Source eDonkey Server with Overnet and eMule Extensions.* http://cdonkey.suche.org. 2003.

[**Mul02**] Tim-Philip Muller. *eDonkey2000 tools.* http://ed2k-tools.sourceforge.net.

[Ocb04] "Ocbmaurice". *eDonkey2000 Server-lists and Statistics.* http://ocbmaurice.dyndns.org 2004.

[HB02] Oliver Heckmann, Axel Bock. *The eDonkey2000 protocol.* KOM technical report. http://www.kom.e-technik.tu-darmstadt.de/publications/abstracts/HB02-1.html

[ct04] c't - Magazin für computer technik. *FAQ Section Entry - "Nervige Port Scans" ("Annoying Port Scans").* Heise Verlag. http://www.heise.de/ct/faq/qna/nervige-port-scans.shtml