Monitoring Challenges and Approaches for P2P File-Sharing Systems

Danny Hughes
Computing, InfoLab21,
Lancaster University,
Lancaster, UK.
+44 (0)1524 510351
danny@comp.lancs.ac.uk

James Walkerdine
Computing, InfoLab21,
Lancaster University,
Lancaster, UK.
+44 (0)1524 510352
walkerdi@comp.lancs.ac.uk

Kevin Lee,
School of Computer Science,
University of Manchester,
Manchester, UK.
+44 (0) 161 2756132
klee@cs.man.ac.uk

Abstract

Since the release of Napster in 1999, P2P file-sharing has enjoyed a dramatic rise in popularity. A 2000 study by Plonka on the University of Wisconsin campus network found that file-sharing accounted for a comparable volume of traffic to HTTP, while a 2002 study by Saroiu et al. on the University of Washington campus network found that file-sharing accounted for more than treble the volume of web traffic observed, thus affirming the significance of P2P in the context of Internet traffic. Empirical studies of P2P traffic are essential for supporting the design of next-generation P2P systems, informing the provisioning of network infrastructure and underpinning the policing of P2P systems. The latter is of particular significance as P2P file-sharing systems have been implicated in supporting criminal behaviour including copyright infringement and the distribution of illegal pornography.

1. Introduction

Peer-to-Peer (P2P) applications take advantage of resources such as storage, CPU cycles, content or human presence available at the edge of the Internet [1] in order to provide a service. P2P file-sharing, wherein users donate content, storage space and network resources to provide a distributed library of files, was the 'killer' application that drove the emergence of P2P with Napster [2] in 1999 and file-sharing has since remained the dominant P2P application [3].

The first generation of P2P file sharing systems largely followed the client-server paradigm. However, legality and scalability issues have driven the development of more decentralized and anonymous file-sharing protocols. Today these systems, which include FastTrack [4], Gnutella [5], eDonkey [6], Direct Connect [7] and Bittorrent [8] have millions of users worldwide.

Empirical studies have shown that P2P file-sharing systems account for a significant proportion of Internet traffic. A 2000 study by Plonka [9] found that P2P file-sharing accounted for 23% of Internet traffic on the University of Wisconsin-Madison campus network. A 2002 study by Saroiu et al [10] on the University of Washington campus network suggested that P2P traffic consumed 43% of total network bandwidth. These studies confirm the growing significance of P2P in the greater context of Internet communications.

Today's P2P file-sharing systems difficult to monitor due to the use of non-standard ports, a proliferation of different P2P protocols and attempts by developers to maintain user anonymity. Moreover, recent work has suggested that users are migrating to systems which are more difficult to monitor [11]. Despite these issues, high quality empirical studies of P2P systems are critical for a number of reasons:

- P2P traffic accounts for a large and growing proportion of Internet traffic [11]. Understanding this traffic is therefore important both for traffic engineering and for provisioning network services.
- Meaningful evaluation of P2P systems can only be achieved when informed by a realistic P2P workload.
- Emergent and often unpredictable issues such as file-availability [12] and undesirable user behaviour [13] [14] have been shown to have significant implications for P2P systems.

Monitoring P2P systems is therefore a difficult, yet *critical* problem. Several studies have attempted to address this issue and each has illuminated a subset of P2P traffic characteristics, though none provides a complete picture. This paper seeks to clarify the situation by introducing a classification scheme for monitoring approaches and performing an investigation into the current state of research in the field of P2P traffic monitoring.

The remainder of this document is structured as follows: Section 2 presents a classification scheme for P2P monitoring approaches. Section 3 describes significant studies of P2P systems. Section 4 discusses the shortcomings of existing P2P studies and finally, section 5 concludes.

2. Classifying P2P Monitoring Approaches

Several significant empirical studies of P2P systems have been performed. These studies vary in terms of their duration, the systems they analyze and the characteristics that they seek to monitor, however, they all follow one of three discrete tracing methodologies: ***network-level*** tracing, ***passive application-level*** tracing, or ***active application-level*** tracing. Each class of tracing methodology is described in detail below.

Network-level traces are performed by deploying code at suitable points in the network infrastructure to perform IP-level packet monitoring. P2P traffic is identified from the greater body of general Internet traffic by matching the observed traffic with the known behaviour of P2P systems. Network-level tracing is relatively transparent and traffic from different P2P systems can be compared side-by-side and with traffic from other domains, such as the web. However, in order to gather a sufficient sample of traffic, monitoring code must be deployed at a key point in the underlying network infrastructure such as at the gateway to a large private network (e.g. an academic network).

Passive application-level traces are performed by monitoring the messages that a P2P system routes at the application level. Modern P2P file-sharing systems are highly decentralized and each peer is expected to participate in the system by routing resource discovery and network maintenance messages. As each peer participates in message passing, passive monitoring can be achieved simply by deploying a modified peer on the P2P network to log the messages that it is required to route. As with network-level tracing, passive application-level tracing is transparent to the underlying P2P network. Unlike network-level P2P tracing, a passive trace can be performed without the necessity for access to the underlying network infrastructure.

Active application-level traces address the shortcomings of passive application level tracing. While passive traces have advantages over network-level traces in terms of their ease of deployment, passive monitoring will typically gather a smaller body of data than a network-level trace due to the 'search horizon' effect that arises from the small-world properties of modern P2P networks [13]. Because of this issue, it would be infeasible to use passive monitoring to trace a significant subset of a P2P network on the scale of today's popular systems [4] [5] [6] [7] [8]. One way of addressing this shortcoming is to employ an aggressive querying and connection policy where the monitoring peer attempts to connect to and interrogate as much of the application-level network as possible; '*crawling*' the P2P network in order to maximize the size and typicality of trace data. While this has the advantage of increasing the size and typicality of data, it does so at the expense of transparency due to the disruptive effect of repeated reconnections and the generation of a large number of resource-discovery messages.

3. Empirical Studies of P2P Systems

This section presents significant P2P traffic monitoring studies belonging to each of the classes introduced in section 2, spanning a period from 2000 to 2005. The specific methodology of each study is described alongside its significant findings. Based upon this survey, the benefits and limitations of each class of monitoring approach are discussed along with the general limitations of current P2P studies. While this survey is not exhaustive, it covers the most significant and oft cited studies of P2P networks.

3.1 Network-Level Monitoring

The first network-level study of P2P traffic was performed by Plonka et al. [9]. This study analyzed the bandwidth consumed by Napster [2] on the University of Wisconsin-Madison network during March 2000. A seven hour trace was gathered using a specially developed tool called FlowScan to monitor Napster traffic. FlowScan first identified nodes communicating with the napster.com servers as potential P2P participants and then applied simple heuristics to the node's incoming and outgoing traffic in order to identify Napster-related traffic. The Plonka study found that as early as 2000, P2P applications generated a comparable volume of traffic to the web at 23.1% of total bandwidth, compared to 20.9% for web traffic. Unfortunately, it is difficult to assess the accuracy of this study due to the lack of published details regarding FlowScan's traffic-catagorisation system. However, the short duration of the trace is likely to have resulted in inaccuracy, particularly as other studies have found significant time-of-day variations [12]. Nevertheless, the Plonka study was useful in highlighting the increasing bandwidth consumption of P2P applications.

The growing volume of traffic being generated by P2P applications was corroborated by in June 2002 by a University of Washington study conducted by Saroiu et al [11] Their nine day trace found that P2P traffic consumed 43% of campus bandwidth, compared to just 14% for web traffic - a significant increase since the Plonka study. The Saroiu study identified traffic generated by the two dominant P2P systems of the day; Gnutella 0.4 [15] and Kazaa [4] based upon common port usage. In addition to raw traffic data, the Saroiu study reported more fine-grained information about the P2P work-load. This included the finding that, on average, objects retrieved from P2P networks were three orders of magnitude larger than objects retrieved from the web and the finding that a small subset of peers are responsible for the majority of P2P traffic - a finding that corroborates the results obtained by Adar et al [13] in their passive application-level study.

Gummadi et al. continued P2P monitoring work at the University of Washington with a 200-day trace of Kazaa traffic in 2003 [16]. This was recorded using a similar methodology to the 2002 trace, except that traffic was

identified based upon Kazaa-specific HTTP headers rather than by port use. Uniquely Gummadi's 2003 trace was long enough to observe seasonal variations in P2P traffic and the effect of changing network policies on P2P workloads. Using this trace, Gummadi developed a detailed parameterized model of P2P workloads, which can be used by developers to generate realistic evaluation data.

Accurate identification of P2P traffic is a vital component of network-level P2P monitoring. In the case of the Plonka trace [9], identification was simplified by Napster's semi-centralized architecture [2], while the Saroiu [11] and Gummadi [16] trace identified traffic by port number and header data respectively. However, recent research [10] has demonstrated that users are increasingly moving to P2P systems that are more difficult to monitor as they use non-standard ports and encrypted header data. To address this issue, Subhabrata et al. [17] have developed a system for real-time network-level identification of P2P traffic. This system was implemented as an extension to the AT&T's Gigascope [18] high speed traffic monitor. Subhabrata et al. evaluated their traffic identification approach using a 24 hour week-day trace and an 18 hour weekend trace gathered in November 2003 on a major internet backbone. This was augmented with a 6 day trace of traffic on a VPN where administrators attempt to block P2P traffic, also conducted in November 2003. Subhabrata's approach proved capable of identifying traffic from today's popular P2P systems in real-time for traffic flows of up to 1gbps while maintaining misidentification rates of less than 5%. While the trace data gathered for this study was used to evaluate their traffic monitoring approach, the authors did not attempt to further characterize the P2P traffic that they observed. The extended version of Gigascope used in this study is capable of identifying traffic from Gnutella [5], Fasttrack [4], eDonkey [6], Direct Connect [7] and Bittorrent [8]. It does not depend upon identification by port and instead uses a more versatile approach based upon *application signatures*, which can be used to categorize traffic where identification data may be at variable offsets in the header. Application signatures can also be used to categorize traffic based upon functional header data. For example in the case of eDonkey [6], identification is based upon the presence of packet-size data located at a known offset.

In each of the studies discussed above, network-level tracing was used to record the low-level characteristics of P2P traffic flows on private networks. Network-level tracing is potentially transparent, scalable and allows comparison of traffic from multiple domains side-by-side. However, this approach is dependent upon access to core network infrastructure, which is not always feasible. While researchers may have access to gateway infrastructure on large private networks, such as academic networks, data obtained from such sources should be viewed as potentially biased due to differences between the characteristics of the private network's users and general Internet users.

3.2 Passive Application-Level Monitoring

The first passive application-level trace of a P2P system was performed by Adar and Huberman in 2000 on the Gnutella 0.4 network [15]. This 24-hour trace logged resource-discovery traffic which was then used to assess the prevalence and characteristics of a problem known as 'free riding', wherein users download resources from, but do not upload resources to a P2P file-sharing system. The Adar trace was performed by modifying the open-source 'Furi' Gnutella client (no longer available) to monitor search, response and peer discovery messages. Adar and Huberman discovered that participation in Gnutella was highly asymmetrical with 66% of peers sharing no files at all and almost 50% of all files being served by the top 1% of hosts. This finding was significant as it contradicted the (then) conventional wisdom that user participation in P2P file sharing systems is symmetrical. Adar's result was later corroborated by Saroiu's 2002 network-level study [10].

Hughes et al [19] revisited the results of the Adar trace in 2004 on the Gnutella 0.6 network [4] based upon a one week trace. The trace was performed using a specially developed monitoring tool based on the Jtella base classes [19]. The monitoring peer connected to the Gnutella network as an Ultrapeer [4] and periodically reconnected in order to maximize the size and typicality of its sample-base. Hughes discovered that in the four years since the Adar study, the proportion of free-riders had increased from 66% to 85%, while corroborating Adar's finding that the top 1% of hosts serve almost 50% of all files. Hughes speculated that the increase in free riding may be the result of an increase in prosecution of copyright infringement.

Hughes et al performed an additional month-long trace in 2005 using a similar methodology in order to assess the level of illegal pornographic material being distributed on the Gnutella network [14]. The study found that an average of 1.6% of searches and 2.4% of responses contained references to illegal pornography and that this material is distributed by a tiny subset of peers that typically share nothing else.

In each of the cases discussed above, passive application-level monitoring is used to study application level properties in an Internet-wide context. Like network-level monitoring, passive application-level monitoring is transparent, however, it does not require access to low-level network infrastructure. Unfortunately, in cases where a very large sample of network traffic is required, passive monitoring would be unsuitable due to the small-world properties of modern P2P networks [13].

3.3 Active Application-Level Monitoring

Ripneau and Foster [21] performed the first active application-level trace of the Gnutella network from November 2000 to May 2001. This study attempted to map the Gnutella network in terms of the average number of links between hosts and the number of hops that these links represent on the underlying IP network. To achieve this, a specialized Gnutella peer known as a '*crawler*' was developed. The crawler connects via the normal Gnutella boot-strapping system and uses Gnutella's peer-discovery mechanism [15] to find new peers. The IP address of these peers is added to the list of those observed and the crawler attempts reconnection in a new location, repeating the process and gradually building a 'map' of the network. The resulting map includes the total number of nodes, the total number of links and average traffic data. Based upon the findings of this study, Ripneau concluded that the emergent structure of the Gnutella network was such that the network's bandwidth consumption would limit its scalability, as predicted by Ritter [22]. Unfortunately, Ripneau's crawling approach is invasive, as repeated reconnection affects the P2P network. It is also un-scalable due to the computational and network expense incurred when crawling the application-level network.

Saroui et al. [23] extended Washington University's work on monitoring P2P systems to the application level with a one month crawl of the Gnutella network in May 2001. The crawler used a similar methodology to Ripneau and observed between 8,000 and 10,000 unique peers, which at that time would have accounted for between 25% and 50% of the Gnutella network. The 2001 Saroiu trace recorded low-level data, including each peer's IP address, latency and bottleneck bandwidth between peers; along with higher level data including each peers advertised bandwidth and the number and size of files being shared. These high-level properties were measured by logging Gnutella's resource discovery and network maintenance messages, while bottleneck bandwidth was measured using SProbe [24], a network tool that uses a TCP exploit to accurately measure bottleneck bandwidth without the need for remote cooperation.

Chu et al [12] performed the first study that attempted to quantify the availability of peers and files on the Gnutella network using a forty day trace performed in early 2002. This trace was gathered by a tool based upon the Jtella API [20] that followed a similar methodology to the Ripneau crawler [21]. Search-response messages were intercepted by the crawler and unique peers were identified based upon their advertised IP and port pairs. The crawler was used to gather a list of 20,000 unique peers using the BearShare [25] and SwapNut (no longer available) clients, at which point a second program, known as the 'tracking manager' attempted to download each peer's file-list using proprietary BearShare and SwapNut extensions. Using this methodology, the availability of peers and files was monitored for a period of 40 days beginning on March 28th. Chu reported a strong correlation between time-of-day and node availability and proposed a model to describe peer availability. Additionally, Chu provided a breakdown of relative file-type popularity and corroborated the finding of Saroiu [15], that file popularity is highly skewed with the top 10% of files accounting for more than 50% of shared data. A clear limitation of Chu's study lies in the use of proprietary extensions to obtain file lists, which limits the size of the trace and introduces possible bias due to the limited user-group studied.

In each of the cases discussed above, active application-level monitoring has been used to study P2P traffic properties in an Internet-wide context, where a very large and typical body of trace data was required (e.g. mapping the Gnutella network). Active application-level monitoring is easy to deploy and should not contain local bias; however, the aggressive reconnection and interrogation approach employed makes this approach invasive and limits its scalability.

3.4 Summary of Monitoring Approaches

This paper introduced a classification scheme for empirical studies of P2P file sharing systems based upon the tracing methodology that they employ: network-level monitoring, passive application-level monitoring or active application-level monitoring. In the context of this classification, significant empirical studies were reviewed along with the benefits and drawbacks of each approach. These are summarized below:

Network-level monitoring is transparent to the network and highly scalable. It is capable of comparing traffic flows from multiple P2P systems side-by-side and is well suited to characterizing P2P traffic on large private networks; however, it is poorly suited for performing global monitoring of P2P systems due to the possibility of local bias. Moreover, network-level monitoring requires low-level access to core network infrastructure, which is often unfeasible. Examples of network-level monitoring studies include [9], [10] and [16].

Passive application-level monitoring is also scalable and transparent to the network. It can be performed without access to core network infrastructure, though it does not provide as large a volume of trace data as network-level monitoring or crawler-based application-level monitoring. Furthermore, it is inherently protocol specific. Passive application-level monitoring is thus best suited to instances where network-level monitoring is impossible or where a non-invasive approach is desirable. Examples of passive application-level monitoring studies include [13], [14], [17] and [19].

Active application-level monitoring is less transparent and scalable than either network-level or passive

application-level monitoring; however, it allows large volumes of trace data to be gathered without low-level access to the network infrastructure. It is thus the best approach where global network information is required and access to the underlying network infrastructure is not possible. Examples of passive application-level monitoring studies include [21], [23] and [12].

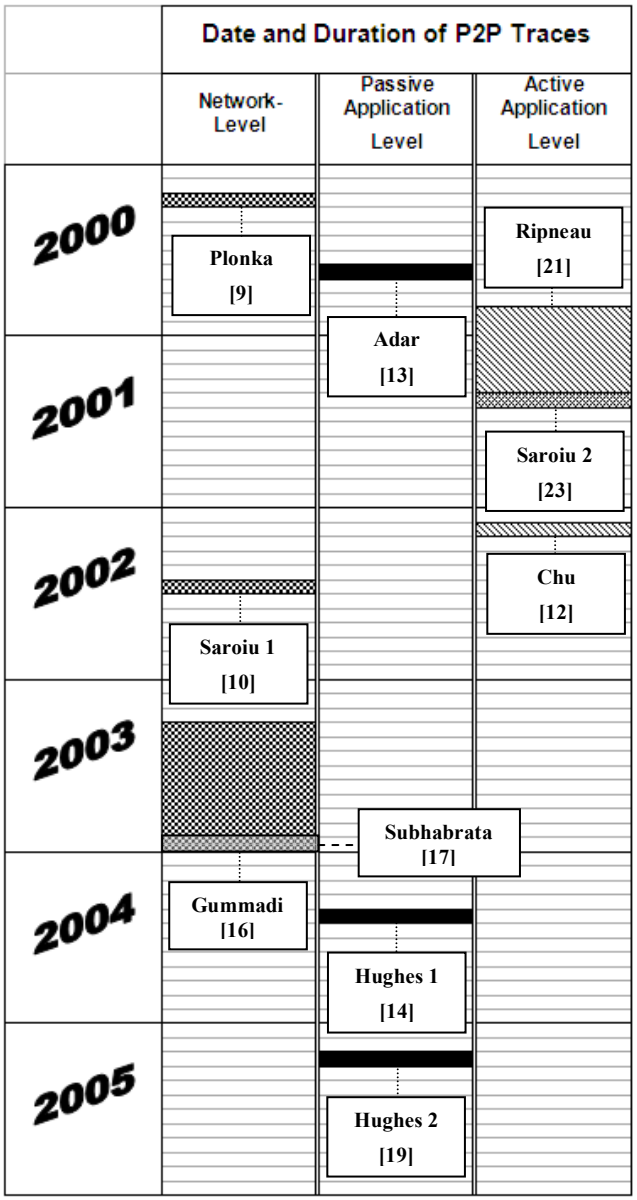

Figure 1 – Time Distribution of P2P Traces

P2P traces such as those presented in this paper have proved invaluable in informing research in the field of P2P systems, however, each of these studies provides only a piece of the puzzle; describing a subset of P2P traffic characteristics for a subset of protocols over the duration of the trace. Often, papers which cite these studies fail to adequately consider such limitations. For example, the data-point provided by Adar's 2000 study of free riding [13] has been used in a significant body of research until the present day, however, when this study was revisited by Hughes et al. [19] in 2005, it was discovered that free riding had increased, revealing a significant, and (until that point), unidentified trend.

Figure 1 illustrates the date and duration of each of the P2P traces discussed in this paper. As figure 1 illustrates, few of the P2P studies presented in this paper are of sufficient duration to identify trends in P2P traffic, rather they simply provide a data-point for the monitored characteristics. The notable exceptions to this are Gummadi's 2003 Kazaa trace [16] which was long enough to observe seasonal variations and Hughes' 2005 study of free-riding [19] which, by revisiting Adar's 2000 experiment [13] was able to show an intervening trend in user behavior.

4. Limitations of Existing Work

There are a number of significant shortcomings in the current body of research on P2P traffic monitoring. The first and perhaps most significant of which is the wide-spread use of closed data sets. As can be seen from Figure 1, P2P studies may require weeks or even months of P2P traffic data. While it is understandable that after investing significant time and effort in gathering a data set, researchers may be reluctant to make this data public, this prevents the findings of studies being verified using different methodologies and prevents trace data being revisited in new contexts.

Another significant gap exists in the body of work on P2P monitoring regarding the identification of underlying trends. For example, the data-point provided by Adar's 2000 study of free riding [13] was revisited by Hughes in 2004 [19] and a significant intervening trend was discovered. It may be is possible that other equally significant trends might be discovered by revisiting past studies. For example, would the growing popularity of digital video be reflected by an increase in the availability of such files since Chu's [12] 2002 study of file availability? Despite the possibility of exposing significant trends in user behaviour, few studies choose to revisit earlier data-points.

Most empirical studies of P2P file sharing systems are concerned only with the technical characteristics of P2P traffic (files shared, bandwidth usage etc.). While this information is critical for simulation of P2P traffic and for the development of approaches to encouraging positive user behaviour, the next step, reasoning about the social and psychological factors which produce this behaviour, is rarely taken. Furthermore, most studies do not take into account the real-world factors which may affect P2P traffic. Notable exceptions to this are the studies by Adar [13] and

Hughes [14] [19], which explicitly consider the social factors which are responsible for observed behaviour.

5. Conclusions

This paper presents a classification scheme for empirical studies of P2P traffic. Past studies using each class of methodology were presented and the strengths and weaknesses of each approach were discussed. Finally, we discussed key shortcomings of existing work in this field and made a number of recommendations designed to address these shortcomings. These include: The use of open data sets, revisiting data-points provided by past studies and increased focus on the socio-technical factors that drive user behaviour on P2P file-sharing systems.

6. References

[1] "What Is P2P", Shirky C., published in Open P2P, O'Reilly, 2000, available online: http://www.openp2p.com/pub/a/p2p/2000/11/24/shirky1-whatisp2p.html

[2] Napster: www.napster.com

[3] "Looking Beyond the legacy of Napster and Gnutella", Nagaraja K., Rollins S., Khambatti M., IEEE Distributed Systems Online, vol. 7, no. 3, 2006, art. no. 0306-o3005

[4] Kazaa: www.kazaa.com

[5] "The Gnutella Protocol Specification v0.6": http://rfc-gnutella.sourceforge.net/src/rfc-0_6-draft.html

[6] eDonkey: http://www.edonkey2000.com

[7] Direct Connect: http://dcplusplus.sourceforge.net

[8] Bittorrent: http://www.bittorrent.com

[9] "Napster Traffic Measurement", Plonka D., Univeristy of Wisconcin-Madison, available online at: http://net.doit.wisc.edu/data/Napster, March 2000

[10] "An Analysis of Internet Content Delivery Systems" Saroiu S., Gummadi K., Dunn R. J., Gribble S. D., Levy H. M., published in the proceedings of the 5th International Symposium on Operating Systems Design and Implementation (OSDI) Boston, USA, 2002

[11] "Is P2P Dying or Just Hiding?", Karagiannis, T., Broido, A., Brownlee, N., Faloutsos, M., In the Proceedings of Globecom 2004, Dallas, U.S., December 2004.

[12] "Availability and locality measurements of peer-to-peer file systems" Chu J., Labonte K., Levine N., published in ITCom: Scalability and Traffic Control in IP Networks. July 2002, vol. 4868 of Proceedings of SPIE.

[13] "Free Riding on Gnutella". Adar, E., Huberman, B., First Monday, October 2000, available online at: http://www.firstmonday.dk/issues/issue5 10

[14] "Is Deviant Behaviour the Norm on P2P File Sharing Networks?" Hughes D., Gibson S., Walkerdine J., Coulson G., in press in IEEE Distributed Systems Online, vol. 7, no. 2, February 2006. http://csdl.computer.org/comp/mags/ds/2006/02/o2001.pdf

[15] "The Gnutella Protocol Specification v0.4": http://www9.limewire.com/developer/gnutella_protocol 0.4.pdf, 2000

[16] "Measurement, Modeling and Analysis of a P2P File-Sharing Workload", Gummai K., Dunn R. J., Saroiu S., Gribble S. D., Levy H. M., Zahorjan J., published in the proceedings of the 19th symposium on Operating Systems Principles (SOSP'03), Bolton Landing, New York, October 2003.

[17] "Accurate, Scalable Network-level Identification of P2P Traffic Using Application Signatures" Subhabrata S., Spatscheck O., Wang D., published in the proceedings of the thirteenth international world wide web conference (WWW2004), New York, USA, 2004.

[18] Gigascope: http://public.research.att.com/viewProject.cfm?prjID=129

[19] "Free Riding on Gnutella Revisited: the Bell Tolls?" Hughes D., Coulson G., Walkerdine J., published in IEEE Distributed Systems Online, vol. 6, no. 6, June 2005. http://csdl2.computer.org/comp/mags/ds/2005/06/o6001.pdf

[20] "JTella", http://jtella.sourceforge.net

[21] "Mapping the gnutella network," Ripeani M., Iamnitchi A., Foster I., IEEE Internet Computing., vol. 6, no. 1, pp. 50-57, Jan./Feb. 2002.

[22] "Why Gnutella Can't Scale, No Really" Ritter, J, http://www.darkridge.com/~jpr5/doc/gnutella.html.

[23] "Measuring and Analyzing the Characteristics of Napster and Gnutella Hosts" Saroiu S., Gummadi K., Gribble S. D., published in Multimedia Systems 9, pp 170-184, 2003.

[24] SProbe: http://sprobe.cs.washington.edu

[25] Bear Share: http://www.bearshare.com