UNITED STATES DISTRICT COURT
SOUTHERN DISTRICT OF NEW YORK

ARISTA RECORDS LLC; ATLANTIC RECORDING CORPORATION; BMG MUSIC; CAPITOL RECORDS, INC.; ELEKTRA ENTERTAINMENT GROUP INC.; INTERSCOPE RECORDS; LAFACE RECORDS LLC; MOTOWN RECORD COMPANY, L.P.; PRIORITY RECORDS LLC; SONY BMG MUSIC ENTERTAINMENT; UMG RECORDINGS, INC.; VIRGIN RECORDS AMERICA, INC.; and WARNER BROS. RECORDS INC., Plaintiffs, v. LIME GROUP LLC; LIME WIRE LLC; MARK GORTON; and GREG BILDSON, and M.J.G. LIME WIRE FAMILY LIMITED PARTNERSHIP Defendants.	CIVIL ACTION NO. 06 CV. 5936 (GEL)

DECLARATION OF DR. STEVEN GRIBBLE IN SUPPORT OF DEFENDANTS' RESPONSE IN OPPOSITION TO PLAINTIFFS' MOTION FOR PARTIAL SUMMARY JUDGMENT

I, Steven D. Gribble, the undersigned, hereby declare as follows:

1. My name is Steven D. Gribble. I am over eighteen years of age, of sound mind, and in all ways qualified and competent to make this declaration. I have personal knowledge of the facts contained in this declaration and they are true and correct.

2. I am an Associate Professor in the Department of Computer Science and Engineering at the University of Washington. I received my masters and Ph.D. degrees in Computer Science from the University of California at Berkeley, and my B.Sc. degree in Computer Science and Physics from the University of British Columbia. My teaching and

research in part focuses on peer-to-peer systems, content delivery systems, and the Web. More broadly, my research specializes in computer operating systems, distributed systems, and computer security, and I have taught both undergraduate and graduate courses on these topics. My curriculum vitae is included as Exhibit 1; my CV also includes a list of my publications within the last ten years.

(a) I make this declaration in support of Defendants' response to Plaintiffs' motion for partial summary judgment and in particular, I am responding to some of the Plaintiffs' claims about the design and operation of the LimeWire client.

3. Except where otherwise indicated, I make this declaration based on my own personal knowledge and experience, including my review of certain documents, deposition testimony and other artifacts referred to in this declaration, including the LimeWire software and source code. If I am called as a witness before the Court, I could and would testify to the matters set forth herein.

Gnutella

4. The file searching and sharing capabilities of LimeWire are based on Gnutella. Gnutella is an *open, decentralized* file-sharing network. Users that participate in the Gnutella network can make files available to others, search for files, and download files from other users on the network. Moreover, users of the Gnutella network (including LimeWire clients) can search for and share any kind of digital file, including text, images, audio, video, and software files. Peers participating within the Gnutella network consist of privately owned and operated computers.

5. By open, I mean that the communications protocol used within the Gnutella network is a public specification, allowing anybody who so chooses to implement Gnutella-

compliant software that can participate in the network. By decentralized, I mean that there are no centralized components that the Gnutella network relies on to operate. Instead, the network is based on a hybrid peer-to-peer architecture (see Exhibit 4), and it is comprised of a vast number of voluntarily participating peers, ultrapeers, and other components. Because of Gnutella's decentralized nature, it would be virtually impossible for any one entity to monitor or control the Gnutella network or its users.

6. When a user wants to find a file on Gnutella, the user types a search query into their Gnutella client software. On behalf of its user, the client software will send a message to its ultrapeer, asking the ultrapeer to help it find matching files. An ultrapeer uses a number of techniques to locate files within the Gnutella network that match the query. Using the "Query Routing Protocol" ("QRP"), an ultrapeer will forward the query to those of its leaf nodes that are likely to have matches. Using "Dynamic Querying" ("DQ"), an ultrapeer will slowly broaden the search by relaying the query to additional ultrapeers, and transitively, to those ultrapeers' leaf nodes.

7. If a Gnutella peer receives a query that matches a file that it is currently sharing, the peer will generate response "query hit" messages that includes "metadata"[1] about the matching files. This metadata typically includes information such as the IP address of the host that has the file, the size of each matching file, the name of each matching file, a hash fingerprint of each matching file, and other type-specific information about matching files. A query hit message contains information that is intended to be displayed to the user that initiated the query, as well as information that is needed by the user's client software to find and download the file if

[1] Metadata is simply data describing a file and its contents. For example, the metadata associated with a music file might include the file name, artist, track title, musical genre, album name, and track length.

the user so chooses. Query hit messages flow back to the peer that initiated the query, either directly over the Internet or indirectly through the peers' ultrapeers.

8. The Gnutella protocol is extensible, in that client software can add "Gnutella Generic Extension Protocol" ("GGEP") blocks to existing Gnutella messages. GGEP allows client software vendors to add features to Gnutella, such as additional metadata to include in query hit messages, without breaking compatibility with existing Gnutella clients.

9. Gnutella clients usually have a notion of an "upload folder." Only files that are placed within the upload folder are candidates for matching queries or being transferred to other users. Some Gnutella clients place files that they have downloaded inside this upload folder. By doing so, a file that has been downloaded becomes available to the Gnutella network from an additional peer. This tends to reinforce the availability of frequently requested files.

10. As the Gnutella network processes a query and returns a set of query hit messages, the user's client software will display a list of files that match the queries the user has issued. If a user chooses to download one of those files, the Gnutella client software will initiate a transfer of the file from one or more of the peers that possess it.

11. In the simplest case, a peer that wants to download a file will initiate a network connection to one peer that has the file, causing the file to be transferred directly between these two peers. Since frequently requested files can usually be found on many peers, Gnutella clients often include the ability to transfer content via a "swarming protocol," in which pieces of the file are downloaded concurrently from multiple remote peers. Swarming increases the performance of transfer, since the downloader can potentially enjoy the aggregate upload rate of all of the peers participating in the swarm.

12. Some Gnutella clients incorporate non-Gnutella file transfer protocols. For example, recent versions of the LimeWire client include the ability to participate in "BitTorrent" swarms. Thus, if a LimeWire user encounters a "torrent file" within the Gnutella network, LimeWire is able to process the file and join the associated BitTorrent swarm directly, without requiring additional BitTorrent software to be installed on the user's computer.

The LimeWire Client

13. The LimeWire client software is a Gnutella- and BitTorrent- compliant client implemented in the Java programming language. There are two versions of the LimeWire software. LimeWire "basic" is free software, while the slightly enhanced LimeWire "Pro" software must be purchased. Lime Wire LLC maintains both versions of the software. In this declaration, I focus only on LimeWire basic; it is my understanding that the enhancements within LimeWire Pro tune some of the parameters settings within the LimeWire software to provide its customers with better performance, but that they do not affect my conclusions.

14. The LimeWire client contains a number of additions to the Gnutella protocol. Some of these additions improve the client's performance, some add new filtering mechanisms to help combat spam, pornography, or hostile peers, and some provide operational support by allowing Lime Wire LLC to propagate updates to LimeWire client users. I will now discuss some of the more notable software features of the LimeWire client, and their impact on Lime Wire LLC's ability to monitor and control users.

15. The LimeWire client software has been released as open source. As a consequence, anyone can download its source code, create derivative works based on it, and either use the derivative code themselves or redistribute it to others. Indeed, as shown in Exhibit 5, several Gnutella clients are available that are derivatives of LimeWire's code.

16. Because the source code is open, Lime Wire LLC does not have full control over how its software gets used. For example, there is nothing preventing a technologically savvy user from downloading the source code, eliminating some features from the code base, optimizing some aspects of how the software behaves, and then compiling and running this modified software.

17. Perhaps more importantly, the open source nature of the LimeWire client permits the Gnutella community to participate in its development. For example, Lime Wire LLC encourages its community of users to submit patches to the LimeWire code base to fix bugs or add features. In contrast, many other peer-to-peer file-sharing clients are closed-source, as was the case with the original Napster client. As a result, their users had no ability to inspect the source code of the software they were using, or to contribute to the software development. Instead, they had to result to complex reverse engineering techniques if they wanted to try to implement compatible software.

18. To begin using LimeWire, a user must first download a LimeWire installer program to her computer, and then execute the installer, which can be downloaded by visiting the "Download.com" web site (at http://download.com) that is owned and operated by CNET Networks, Inc. In addition to being available through Download.com, the LimeWire installer can also be accessed from LimeWire LLC's website. Once the user has downloaded the LimeWire installer, the user must launch the installer. The installer places various files in different folders on the user's computer; at this point, the user can go ahead and launch the LimeWire client software itself.

19. When the user launches the software for the first time, the LimeWire client presents the user with a series of configuration screens, culminating in a dialog box asking the

user to agree not to commit copyright infringment. The user is unable to begin using the LimeWire client software until the user has agreed not to commit copyright infringement. A copy of a screen shot showing this prompt is attached as Exhibit 6.

20. In order to share files, a user of LimeWire must designate on her computer the files or folders that she wishes to make available for sharing over the Gnutella network. When LimeWire is first installed, a new folder (called "Shared") is created on behalf of the user and is automatically designated for sharing. By default, all files that are downloaded by LimeWire are automatically placed in the Shared folder. The user may also add or remove individual files and folders from the list of items that are shared, through a series of configuration screen available within LimeWire. Exhibit 7 shows screen shots of these LimeWire configuration screens. A user is not required to share any files in order to use LimeWire, and LimeWire will not share any files or folders that have not been designated for sharing in the manner described here. Based on my analysis of the LimeWire source code and the operation of LimeWire, neither LimeWire LLC nor any LimeWire LLC server is involved in any way in the user's voluntary and individual act of designating certain files for sharing over Gnutella network, and at no time does the act of designating such files require or involve the exchange of any message or other information with a LimeWire LLC server.

21. File sharing activity begins when a user initiates a search for a file.[2] To initiate a search, the user types a search string into the "search" box on the LimeWire user interface. The search string costs of a string of words to describe the file that the user wants to find on the Gnutella network. For example, a user might type in the word "Shakespeare" to try to find documents that contain the works of William Shakespeare. Exhibit 8 is a screen shot of the basic

[2] Unlike Napster, which could only search and share MP3 audio files, LimeWire – like all Gnutella clients – can be used to search and share all types of digital files, including text, images, audio, video, and software.

search screen for LimeWire. Using more advanced search screens, in addition to specifying a list of general words to match, a user of LimeWire may also generate searches for files that have specific attributes (or "meta-data"), such as a search for a text document with a particular title, or an audio file from a particular artist. The LimeWire user interface allows a user to indicate what type of file to search for, and for each file type (such as text, audio, and video), the user interface contains a list of attributes that can be searched against. See Exhibit 9.

22. The candidate files listed in the LimeWire user interface are anonymous and are not distinguished by platform. LimeWire only displays certain basic information about the candidate file (e.g., file type, filename, and file size). No information identifying the user of a hosting peer, the hosting peer itself, or whether the candidate file is from a LimeWire peer or a non-LimeWire peer is displayed. (See Exhibit 10.)

23. The sending and receiving of query and query hit messages in connection with searching the Gnutella network is a fundamental element of the Gnutella network protocol. No interaction with LimeWire LLC servers is required for these messages to be exchanged between Gnutella peers. Based on my review of the LimeWire source code, and on my observations of LimeWire while executing, none of the query or query hit messages generated as a result of searches conducted by voluntarily-participating Gnutella peers (including LimeWire peers) are reported or relayed to LimeWire LLC servers. Therefore, it is my opinion that LimeWire has no ability to monitor or control the content of these messages.

24. Each query hit message that flows back to a LimeWire peer contains the file name of a candidate matching file, the IP address of the hosting peer, and other metadata. If the LimeWire user initiates a download of a candidate matching file, LimeWire uses this IP address to create a temporary Internet connection directly between the LimeWire peer and the hosting

peer (i.e., the peer hosting the file to be downloaded). LimeWire uses this temporary Internet connection to transfer the file from the hosting peer to the LimeWire peer. (See Exhibit 11.) This transfer is accomplished using "HTTP", the standard World-Wide Web file transfer protocols.[3]

25. One theme in Plaintiffs' motion is that the LimeWire client software is optimized for popular audio files that are not authorized. The LimeWire client, like all Gnutella clients, is capable of finding and transferring any kind of file. The Gnutella network itself is largely agnostic towards file type and content, and as a result it equally supports the publication, search, and transfer of non-infringing and infringing files. Based on my examination of the client, and on my discussions with Sam Berlin and my review of deposition transcripts, I conclude that the primary design goals of the LimeWire client engineers were (1) to help users find content they are looking for, regardless of type, and (2) to maximize the performance of the file transfers that users attempt to initiate.

26. In my opinion, the LimeWire client has not been optimized for infringing use. The LimeWire client equally supports non-infringing and infringing uses.

27. Plaintiffs take issue with the fact that the content authority filtering mechanism in LimeWire is disabled by default, and concludes there is no technical reason why Lime Wire LLC could not have set the default to ON. While this is technically correct, it is worth noting that turning content authority filtering off by default is consistent with the LimeWire client design principles communicated to me by Sam Berlin, namely that users are given enough flexibility to enable, disable, or configure filtering, and that filtering is enabled by default when security issues are concerned.

[3] These diagrams depict the simple case in which the LimeWire client software downloads a file from one peer. As previoiusly mentioned, the LimeWire client is also able to download pieces of a file from many peers in parallel through a process called "swarming."

28. Plaintiffs also comment on how the "bitrate" column is always present in search results, regardless of whether the user has specified that he is searching for audio content. This is true, but it is not compelling evidence that LimeWire is optimized for the transfer of audio content. By way of example, many email clients display an "attachments" column when listing messages within a folder to indicate how many attachments a message has. This column is typically empty for email messages that contain no attachments, but this is not evidence that the email client is optimized for messages with attachments above messages with no attachments.

29. Plaintiffs correctly observe that the LimeWire client contains a built-in media player that is capable of playing MP3 audio files, but is incapable of playing other media types (such as video files or images). From my discussion with Sam Berlin, I am informed that until recently, Java did not support the simple and high performance rendering of video files, leading the LimeWire engineers to choose not to include video support in the LimeWire client. As well, I am informed that the engineers chose not to support inline image rendering as third party rendering tools are plentiful and using them is adequately non-disruptive for LimeWire users. Also, note that the built-in media player was not always present in the LimeWire client application; I am informed that it was first added once Lime Wire LLC determined that Java's audio support was adequate.

30. Plaintiffs' motion also discusses how the LimeWire client contains support for the ID3 metadata specification. ID3 was designed as a way to tag an audio file (usually an MP3-formatted file) with extra "metadata" describing its content. The ID3 specification contains many standard kinds of tags, including those that describe the genre of music. Not all files contain rich metadata, but as it happens, many audio files do, in part due to the existence of the ID3 specification.

31. In my opinion, it is perfectly reasonable for a law-abiding user to want to search for audio content using a file-sharing client, even if some audio content found is non-infringing while other content is infringing. Given that a user might want to search for audio content, a natural engineering choice is to allow that user to specify ID3 tag metadata as search criteria; doing so helps the user find the content he is looking for, which is consistent with the LimeWire design goals described to me by Sam Berlin. Thus, in my opinion, the fact that LimeWire includes ID3 metadata mechanisms is not an indication that it is optimized to support infringing use.

32. Plaintiffs also refer to genres such as Chamber Music, Classic Rock, Disco, and others, as "commercial categories." These genres are not necessarily commercial; music in these genres could be commercial or free, just as music in these genres could be non-infringing or infringing.

33. Moreover, Plaintiffs use the example of the Classic Rock or Top 40 genre tag, and comments that a search within this genre is "clearly designed to generate search results containing unauthorized works." In my opinion, this is a false conclusion. Not all Classic Rock or Top 40 audio files are infringing or unauthorized. For example, my understanding is that the Classic Rock band "the Grateful Dead" promoted the free circulation of bootleg tapes made by fans during its concerts. Assuming this is true, it is example of an authorized audio file in the Classic Rock genre.

34. Plaintiffs also conclude that LimeWire is optimized for downloading popular audio files. However, they fail to mention mechanisms within the LimeWire client designed to help users find unpopular files. In particular, the "Mojito DHT" functions within LimeWire were

designed to help users locate rare files that only exist in a few locations within the Gnutella network.

35. It is worth noting that the Gnutella protocol and the LimeWire client have both evolved over time to improve the overall performance, scalability, and responsiveness of the Gnutella network. For example, dynamic querying and the query routing protocol (described briefly earlier in this report) were both developed to make searching on Gnutella more efficient, scalable, and effective, and both. These two mechanisms optimize searches for any file type, not just audio files.

Filtering

36. Filtering is a technique for controlling access to content. When employing filtering, automated processes attempt to determine what content should be accessible and what content should be suppressed. There are many different kinds of filters that are possible; for example, a filter can be extremely precise, identifying a very specific individual file or object, or it can be broad, identifying a large collection of files or objects, each of which matches the filter's criteria.

37. Designing a filtering system is complex, and there are several aspects that one must consider. A filtering system could *filter out*, only allowing access to an object if no filter matches it, or it could *filter in*, only allowing access if at least one filter matches.[4] The selection criteria specified by a filter could be based on matching textual words or phrases in file metadata, on the precise hash value of file data, or on a "semantic" description of the content of a file, such as an acoustic fingerprint. A filtering system could exist in an adversarial environment, in which publishers and consumers of files attempt to evade filters, or the operator of the filtering system attempts to suppress authorized content.

[4] Filtering out is also referred to as *blacklisting*, while filtering in is referred to as *whitelisting*.

38. Filtering solutions suffer from two classes of errors. A false positive occurs when a filter erroneously or unintentionally matches an object. A false negative occurs when a filter fails to match an object that, according to some external goal or purpose, it should have matched. In general, a tradeoff exists between the two classes of errors. By making filters very narrow, a filtering system will tend to reduce the number of false positives generated, at the expense of increasing the number of false negatives. Conversely, by making filters very broad, a filtering system will tend to reduce the number of false negatives at the cost of increasing the number of false positives.

39. Filtering is a complex topic; in my opinion, it is not yet known how to build an effective, affordable, and scalable filtering technology for deployment in a peer-to-peer network. This is particularly true in the face of adversaries, with which any peer-to-peer network would have to contend.

40. Plaintiffs also attack the IP address filtering mechanisms within the LimeWire client, and claim that these mechanisms circumvent or reduce the effectiveness of certain anti-piracy techniques. These conclusions are misleading. IP filtering is a very broadly applicable and commonly found tool that can be used to protect a client from exposure to various kinds of danger. From my conversations with Sam Berlin, I am informed that Lime Wire LLC uses IP filtering to shield users from Gnutella peers that propagate spam and Junk content, and that IP addresses appearing within the blacklists that are maintained and distributed by Lime Wire LLC meet the criteria of being spam and junk content distributors. More specifically, Sam Berlin informs me that Lime Wire LLC has not attempted to replicate PeerGuardian functionality; if PeerGuardian IP addresses happen to exist within any of these blacklists, it is because those IP addresses were found to meet the criteria indicating that they are spam or Junk distributors.

41. Plaintiffs also attack the use of a "magic string" by the LimeWire store. This magic string is attached to files purchased from the LimeWire store. If a magic string is detected by a LimeWire client, the client moves that file to a separate folder for LimeWire store material. As a consequence, the file is prevented from being shared by the LimeWire client. Plaintiffs argue in essence that Lime Wire LLC itself appears unwilling to rely on its hash-based filtering system to protect its content.

42. This conclusion is not supported by fact. The magic string approach and the hash-based filtering used by content authority filtering were designed for very different scenarios, and have different constraints. With the magic string approach, Lime Wire LLC had the luxury of adding the magic string tagging mechanism to the centralized LimeWire store. In contrast, content authority filtering must content with files hosted on the millions of peers participating in the Gnutella network that Lime Wire LLC does not control: Lime Wire LLC cannot impose the same kind of tagging on this content. Given the substantial differences between these two scenarios, it is not surprising that different designs were used. Plaintiffs' conclusion that Lime Wire LLC appears unwilling to rely on hash-based filtering is unfounded.

43. Plaintiffs promote the use of Audible Magic's audio fingerprinting technology, and its deployment within the iMesh P2P network. Though Audible Magic reports accuracy rates of 99%, I am unaware of any objective, quantitative third-party evaluation of the false positive and false negative rates of Audible Magic's technology over a corpus of the size of the Gnutella network. As well, Plaintiffs cite a declaration from Benjamin Sorensen, who claims that Audible Magic can scale to tens of millions of checks per day; however, Plaintiffs do not attempt to quantify how many checks per day would need to be performed in a LimeWire-like setting, nor do Plaintiffs discuss the cost of adopting this proprietary technology or attempt to describe its

impact on user privacy. Finally, Plaintiffs does not attempt to determine whether Audible Magic's technology would be effective in the face of adversarial users that attempt to bypass or evade it.

44. There are many challenges and shortcomings in Plaintiffs' filtering proposal that may not be immediately apparent, but are related to the complexities faced by any filtering system. To name a few examples: (1) The proposed solution requires incorporating proprietary Audible Magic technology within the open source LimeWire client codebase, which could be problematic. (2) The proposed solution would cause LimeWire clients to interact with Audible Magic's servers for many downloads; if Audible Magic's servers are slow or inaccessible, for example during a denial of service attack or the failure of the company's network connection, this would negatively impact the LimeWire user experience. Similarly, if Audible Magic becomes adversarial towards users of LimeWire or Gnutella, that company would effectively gain the ability to disrupt the activities of the users. (3) Nothing would prevent users from modifying the LimeWire source code to excise the filtering technology; this is an example of an adversarial strategy that a filtering system must contend with. (4) There does not appear to be any mechanism by which a user or content provider could object to and override a filtering decision in the case of a false positive.

45. It should be noted that LimeWire already provides a mechanism by which content providers can supply hashes to Lime Wire LLC. Plaintiffs do not discuss the design of filtering systems that take advantage of this existing functionality. For example, content providers could search for files on Gnutella that are potentially infringing, contract with Audible Magic to verify the content of those files, generate hash values from files that are confirmed as infringing, and supply those to Lime Wire LLC. All of this could be done in an automated fashion. Note,

however, that this alternate solution would still suffer from the same kinds of shortcomings as any filtering system. The accuracy of searching for infringing files on Gnutella would be roughly equivalent to the accuracy of keyword filtering, and any false positives or false negatives generated by Audible Magic would impact the effectiveness of this hypothetical system.

SIMPP

46. The SIMPP mechanism allows Lime Wire LLC to transmit "viral messages" to LimeWire clients, as described earlier in my report. Viral messages primarily exist to let Lime Wire LLC to tune parameters that affect how the LimeWire client interacts with the Gnutella network. The SIMPP mechanism does not permit Lime Wire LLC to control the activities of LimeWire users.

iTunes Integration

47. Under the section entitled "LimeWire is optimized for downloading popular audio files," Professor Horowitz comments on the iTunes integration of the LimeWire client using the "DAAP protocol." The DAAP protocol allows a program that manages collections of audio to make those audio files available to DAAP-compliant client applications on the same local-area network. Because LimeWire contains some support for DAAP, if one user is running a DAAP capable client, such as iTunes, she will be able to browse and play music from the LimeWire shared folder of another user, assuming those two users are on the same local network (e.g., within the same house or office).

48. LimeWire acts as a DAAP source, but not as a DAAP client. That is, the LimeWire client does not contain the necessary mechanisms to allow its users browse audio collections shared by other DAAP sources. Accordingly, the integration of DAAP into LimeWire essentially allows programs like iTunes to act as media players for LimeWire-downloaded

media. However, DAAP integration does not augment LimeWire's ability to search for or download audio files, popular or otherwise.

Anti-Freeloading

49. Plaintiffs also argue that several anti-freeloading mechanisms within the LimeWire client were developed to locate popular music more easily. Peer-to-peer networks rely on the contribution of content, network bandwidth, and storage by their constituent peers in order to provide a useful and scalable service. Peers that choose to consume resources without contributing any can degrade the performance and scalability of a peer-to-peer network. Accordingly, many P2P protocols and clients are engineered with anti-freeloading mechanisms to combat this problem.

50. Interestingly, the Gnutella protocol itself does not contain anti-freeloading mechanisms. In contrast, protocols like BitTorrent attempt to combat freeloading through the use of "incentive compatible" design. For example, a peer in BitTorrent will measure the rate at which other peers within the swarm are transmitting data, and will preferentially provide content to peers with a track record of uploading data. In BitTorrent, peers that want to achieve good download performance have an incentive to upload, since peers that do not upload are implicitly punished.

51. The LimeWire client features to combat freeloading generally serve to improve the overall Gnutella network performance, rather than to affect the performance of any individual peer. By defaulting to make content available for upload after it has been downloaded, more replicas of content will be accessible on the network, and therefore peers will tend to be able to find the content more easily and transfer the content with higher performance.

52. These anti-freeloading mechanisms only benefit the LimeWire client in an indirect, non-immediate way. It is my opinion that these anti-freeloading mechanisms do not serve to optimize the download of popular audio files specifically. Instead, they serve to optimize the overall performance of the Gnutella network.

Use of "Cookies"

53. Plaintiffs' motion discusses how the LimeWire.com web site would ask a user to confirm that they will not use the LimeWire client for infringing uses, and would decline to provide the user with the software if the user indicates they might commit infringing acts, and then comment on the ability of a user to change their answer and subsequently download the software, and speculates on using "Web cookies" to make it more difficult for a user to do so.

54. However, even though this suggestion may on the surface seem like a reasonable design choice, it has some potentially negative consequences. Consider a user that accidentally clicks on the wrong radio button choice, or one that uses a friend's Web browser and clicks on the choice that indicates an intention to infringe. In this case, the use of a Web cookie would make it more difficult for the user to undo this action and proceed as actually intended. Moreover, even though using a Web cookie would increase the difficulty of changing their answer, a user could simply clear cookies from their browser in order to try again.

55. More broadly, because the LimeWire client is open source, redistributable, and accessible from many sources, it would be very difficult for Lime Wire LLC to prevent a determined user that intends to infringe from downloading and using their software.

56. Plaintiffs also note that a user can download the LimeWire from Web sites other than limewire.com, including download.com and gnutelliums.com, but that these other sites do not ask the user to state their intentions with respect to copyright infringement. The LimeWire

basic client is free software released under the Gnu Public License (GPL),[5] and anyone can choose to redistribute it from any Web site. Accordingly, Lime Wire LLC cannot control the manner in which these Web sites make the software available, and in particular, they cannot require that the Web sites include mechanisms asking the user to state their intentions.

57. It is also worth noting that more recent versions of the LimeWire client ask the user to state their intentions the first time that the user launches the client, and that the limewire.com Web site no longer requires the user to confirm their intention before downloading the client. Because this functionality now exists within the client itself, users will be presented with it regardless of the Web site from which they download the client software.

Shortcoming in Waterman's Report

58. Professor Waterman's report describes a series of experiments performed to quantify the nature of content shared on and downloaded from the Gnutella network. Measurement studies of this nature are quite tricky to conduct in a way that produces unbiased, accurate, and representative results. I will now make several observations on the methodology used in Professor Waterman's study and the some potential the methodology has for introducing bias into the results.

59. Professor Waterman's report describes the three phases of his study; I will structure my comments to parallel those phases.

60. In this phase of the study, Professor Waterman compiled a large list of files that are available through the Gnutella network and observable by the LimeWire client. An implicit goal of this phase was to make the compiled list a representative sample: if the list is representative, than conclusions drawn from it (and random subsets of it) would likely be valid

[5] The text of the GPL is available at: http://wiki.limewire.org/index.php?title=Gnu.

for the Gnutella network at large. There are a few aspects to Professor Waterman's study that call into question the representativeness of his results.

i. Professor Waterman seeds his list by using the "What's New" feature of the LimeWire client. However, only those Gnutella clients that have implemented support for What's New queries will respond to such queries. Thus, a biased subset of Gnutella peers – only those that support What's New – were sampled by Professor Waterman's study.

ii. Professor Waterman disconnected his client from its set of Ultrapeers after each "What's New" query, and reconnected to a different set of Ultrapeers for successive queries. Professor Waterman presumably did this to attempt to probe the Gnutella network from a diversity of vantage points. However, Professor Waterman's report does not describe which Ultrapeers he connected to, nor does it analyze how diverse that set of Ultrapeers is or whether the "correct" kind of diversity was captured to ensure that the vantage points are representative of the Gnutella network at large.

iii. Professor Waterman used the "browse host" feature of the LimeWire client to obtain additional files for inclusion in the master hash library from hosts that respond to the What's New request. However, additional files would only be obtained from those clients that responded both to What's New requests and to "browse host" requests. Some Gnutella software will not respond to one, the other, or both of these kinds of requests, and as a result, any files made available through such software would be systematically excluded from the master library.

iv. Professor Waterman's study only considered files shared on the Gnutella network. The LimeWire client, however, is able to download files using BitTorrent as well. Accordingly, Professor Waterman's results do not consider the significant amounts of non-infringing content available for transfer via BitTorrent. Specifically, Professor Waterman draws conclusions such as "I estimate that 92.7% of the files available for download through the LimeWire client are not authorized for free distribution on peer-to-peer networks." Given that BitTorrent files are available for download through the LimeWire client, it is likely that this conclusion is incorrect.

61. Given the master hash library, Professor Waterman randomly selected a list of 10,000 files for additional study. Professor Waterman attempted to download these files, but excluded files that (a) could not be successfully downloaded, or (b) appeared to contain viruses or child pornography. The first 1,800 files that survived this pruning were analyzed manually.

62. For better or for worse, an appreciable number of files made available on the Gnutella network contain spam, viruses, or other malware. By excluding files that contain viruses from his study, Professor Waterman likely overestimates the fraction of files available on Gnutella that are infringing or unauthorized.

63. The final stage of Professor Waterman's experiment involved the offering of these 1,800 files on the Gnutella network, and measuring the fraction of requests that correspond to infringing or unauthorized files. From this, Professor Waterman projects his measurements to draw conclusions about the fraction of requests to the global population of LimeWire clients that are for infringing or unauthorized files. There are several reasons why this projection is problematic, however.

i. Given that there were many potential sources of bias leading to the selection of these 1,800 files, there is concern that the requests observed are biased as a result. To reiterate, the fact that 92.7% of the files made available by Professor Waterman were potentially infringing has a strong effect on the kinds of queries and transfer requests that Professor Waterman's peer observed. The concerns over the representativeness of this set of offered files flow to the set of requests and transfer requests observed.

ii. It appears as though Professor Waterman offered the files from a single instance of the LimeWire client running in a single location on the Internet, though not enough details about the study were provided in his report for me to be certain of this. If true, the global Gnutella network might not have had sufficient opportunity to observe or request those files. The set of peers whose searches and transfer attempts would arrive at Professor Waterman's client might be biased relative to the global Gnutella network, or to the global population of LimeWire clients, as a result.

64. To draw conclusions about the download behavior of the population of LimeWire clients, one would need to conduct a study that attempts to sample the behavior of this global population. A study that monitors requests appearing at a single location on the Gnutella network might not be representative.

65. The issue of measurement bias and representativeness has been studied in the context of peer-to-peer systems. I have seen two alternative methodologies used to conduct studies that might be less likely to suffer from bias in the data. The first method involves gathering measurements from one or more network links of large-scale ISPs. (See, for example,

"Analyzing peer-to-peer traffic across large networks" by Subhabrata Sen and Jia Wang, published in the 2002 Internet Measurement Workshop.) This technique tends to be useful for measuring aggregate traffic characteristics. Admittedly, though, it is difficult to arrange for access to such links and to adequately preserve user privacy.

66. A second method involves "crawling" the Gnutella network in a way that leads to an unbiased sampling of peers. This technique is described in detail in "On Unbiased Sampling for Unstructured Peer-to-Peer Networks" by Daniel Stutzbach, Reza Rejaie, Nick Duffield, Subhabrata Sen, and Walter Willinger, published in the 2006 Internet Measurement Conference. By using this technique, it might be possible to gather a more representative sample of peers within the network, and from there, it might be possible to characterize files that are made available with less bias.

Ultrapeers

67. The LimeWire file-sharing software, which incorporates more recent versions of the Gnutella protocol and specification, takes advantage of an architectural component called an "ultrapeer." An individual Gnutella peer can choose to promote itself to become an ultrapeer. Ultrapeers provide service to a relatively small number of Gnutella peers; a peer that uses the services of an ultrapeer is called a "leaf" of that ultrapeer.

68. An ultrapeer acts in a manner somewhat reminiscent of Napster's centralized servers, in that it manages searches on behalf its leaf peers. However, instead of an ultrapeer indexing all files from all peers participating in the Gnutella network, an ultrapeer learns a small amount of information about the files made available by its leaves. Using this information, the ultrapeer routes queries between its leaves, and also potentially forwards queries on to other ultrapeers within the Gnutella network.

69. Ultimately, ultrapeers reduce the number of hosts with which a peer must communicate in order to find a file, and as a result, ultrapeers increase the efficiency, performance, and scalability of the Gnutella network. An ultrapeer does not, however, have the ability to force a peer to become its leaf, and no ultrapeer has a complete view of the entire Gnutella network. Accordingly, ultrapeers have only a very limited ability to monitor or control the behavior of peers in Gnutella.

Copyright Notices

70. Users of the LimeWire client are presented with several notices about copyright infringement and the transfer of potentially unauthorized files. I show some screenshots of these notices in Exhibit 12.

71. The first screenshot in the attachment shows a Web page that the limewire.com Web site would present to a user when they attempt to download the LimeWire client software. If a user indicates that they "might use LimeWire for copyright infringement," the Web site presents a user with a page that indicates that "Lime Wire LLC does not distribute LimeWire Basic to people who intend to use it for purposes of copyright infringement. [...] Thank you for your interest; however, we cannot complete this download." Also note that the Web page shown in this screenshot includes a link called "Find out more...". This link, if selected, takes the user to page discussing safety issues with peer-to-peer software, including some material on copyright infringement. This page is shown in the second screenshot in Exhibit 12.

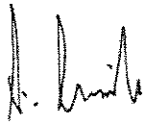
72. The third screenshot shows another Web page hosted on the limewire.com Web site that discusses copyright issues. The page contains information for LimeWire client users about the illegality of using the software for copyright infringement, some suggestions on determining the legality of sharing files, and information about filtering and privacy. As well, the

page provides a link to information about a filtering system that Lime Wire LLC has implemented and made available to copyright owners.

73. The fourth screenshot shows a dialog box that is presented to a user by the LimeWire client software if the user attempts to download a file for which the LimeWire client is unable to find a license. Note that the user can choose to complete the download or terminate it. As well, the user can ask the LimeWire client to cease presenting this dialog box on subsequent file downloads.

74. In my opinion, Lime Wire LLC has taken several reasonable steps to provide informational notices to the user to educate them about uses of the software, the illegality of using the software for infringing use, and some of the risks of using the software.

I declare under penalty of perjury under the laws of the United States of America that the foregoing is true and correct and that this declaration is executed in Seattle, Washington on September 10th, 2008.



Steven D. Gribble, Ph.D.