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' MOTIONS FOR 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.

3. I make this declaration in support of Defendants' motion for partial summary judgment. In particular, I have focused on the following:

(a) the structure and operation of versions 4.12.6 4.16.6, and 4.18.3 of the LimeWire software ("LimeWire"), including the implementation of the Gnutella file-sharing functions within the software;

(b) the interactions between LimeWire and Lime Wire LLC's servers; and

(c) differences between the Gnutella Network (as accessed via LimeWire) and other content delivery systems, including other peer-to-peer systems such as Napster.

4. 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.

Summary of Opinions

5. **The Gnutella network is "open" and the LimeWire client software is just one of many different software packages that are available that enable users to share files via Gnutella.** In many ways, this is similar to how the Internet Explorer web browser is just one of many Web browsers that users can install and use in order to view Web pages.

6. **Based on my analysis of the LimeWire client software, as it ships and in its default configuration, I conclude that the defendant, Lime Wire LLC, does not have the ability to monitor or control the file-sharing activities of its users.**

7. **Given the open source nature of the LimeWire client, Lime Wire LLC does not have the ability to fully control how its code and software is ultimately used.** In particular, third parties can modify the source code to disable, add, or modify existing functions, and third parties can (and do) create and distribute derivate versions of the client software.

8. **Even if Lime Wire LLC ceased all operations, existing and new users of the LimeWire client software (and its derivatives) would be able to join and use the Gnutella network.** The Gnutella network is a "decentralized" system, and as such, no specific hosts, servers, or peers are needed for the network to operate or its participants to use it.

9. **The LimeWire client is able to transfer material available via the BitTorrent protocol as well as over the Gnutella network.** As such, users of LimeWire can use the software to access large amounts of non-infringing content made available via BitTorrent, including open source software.

Technical Background on Peer-to-Peer File Sharing

10. The LimeWire client software provides its users with the ability to share computer files with each other, and with users of other Gnutella-compliant client software, over the Internet. For example, using LimeWire software, a user can make photos, movie files, audio files, computer software, word processing documents, and any other kind of computer file available for other users to find and download. As well, LimeWire allows users to download files using the "BitTorrent" protocol, which is second style of peer-to-peer network. Before diving into the technical details of how LimeWire's software functions, it may be helpful to provide a technical overview of the Internet and of peer-to-peer systems.

The Internet

11. The Internet is a worldwide computer network that interconnects hundreds of millions of computers and allows these computers to communicate with each other electronically. The Internet consists of both physical components, such as the wires and routers that together make up the Internet network topology, and communications standards and protocols, such as TCP/IP, that define how messages are formatted and how these messages are routed over the network. Each computer connected to the Internet is usually referred to as a "host," and the messages sent between hosts are usually referred to as "packets."

12. Each host within the Internet usually has two different kinds of names associated with it. An "IP address" is the machine-readable name associated with a host; an example of an IP address is "64.156.82.107." For one host to send a message to another host, it constructs a packet that contains the IP address of the destination host, and it sends that packet into the Internet, which then routes and delivers the packet to the destination. Most hosts also have human-readable names associated with them, called "DNS" names. For example, "www.limewire.com" is the DNS name associated with the host that has IP address "64.156.82.107." People often use DNS names when referring to a specific Internet host, such as when constructing the name of a Web page they want to visit. Computer software automatically translates DNS names into IP addresses.

13. The basic function of the Internet is to allow hosts to exchange packets with each other, using the TCP/IP protocol. A vast number of applications have been developed for the Internet, including email, the World-Wide Web, Internet telephony, and peer-to-peer ("P2P") file-sharing. Each of these applications consists of a set of computers running software that causes the computers to interact with each other according to a higher-level protocol. For example, "SMTP" is the protocol that defines how Internet hosts interact with each other to

allow one computer user to send an email message to another computer user. Similarly, "HTTP" is the protocol that defines how Web client software retrieves Web pages from remote Web servers. There are several different peer-to-peer file-sharing application protocols, including the "Gnutella" protocol. These P2P protocols define how P2P clients interact with each other to provide their users with the ability to search for files shared by other users and to exchange files with each other.

14. Protocols define the sequence and format of packets that hosts use to communicate with each other. In addition to having protocols, Internet applications such as the Web, email, and P2P networks also have "architectures," which are blueprints that shape the global structure and roles of the hosts participating in the application. Over time, several different canonical architectures have been identified and applied to Internet applications, including "client-server", "peer-to-peer", and "hybrid" architectures.

Client-Server Systems

15. In an application that uses a client-server architecture, the functionality of the application is partitioned between two distinct classes of hosts: servers and clients. (See Exhibit 2.). A server is typically a powerful computer, or even a large collection of computers, that provides some useful service to its clients. A client, on the other hand, is typically a simple computer that sends requests to the server, and receives responses back. A server usually provides its service to many clients, whereas clients usually provide no service to each other or to servers. Without servers to provide them service, clients are essentially helpless. Because a server is typically implemented as a dedicated host (or collection of hosts) that is located in a single place within the Internet, a server is often referred to as providing a "centralized" service.

16. The World-Wide Web is based primarily upon a client-server architecture. Web clients, which are hosts that run Web browser software such as Microsoft's Internet

Explorer, send requests for Web pages to Web servers, Web servers receive these requests, retrieve or generate the appropriate page data, and transmit the data back to the client within a response. In the Web, users describe the pages they want to fetch using a "URL"[1] such as "http://www.cs.washington.edu/homes/gribble/index.html". A URL contains the DNS name of the Web server that is responsible for serving that page. The most popular Web sites receive thousands of Web requests from around the globe every second, and they serve tens of millions of different Web clients each day.

17. The client-server architecture is simple to understand, and many interesting and powerful Internet applications use it. However, it has several fundamental challenges. If a server grows in popularity, it is faced with escalating demand from its clients. To satisfy this demand, the operator of a server typically must spend considerable money to provision it with an adequate amount of computing and networking resources, so that it can "scale up" to handle its large and growing client base. As well, because many clients depend on its services, a server must not fail very often; if a server crashes, all of its clients are potentially denied service. A server can in this way be a "single point of failure" for a given application, and operators often go to great lengths to engineer enough redundancy into their servers to prevent a single component failure from causing the server to become unavailable.

18. Because clients rely on servers to provide services, a server is in a strong position to monitor and control the activities of its clients. For example, a Web server that provides search engine functionality might attempt to log all requests of its users, deny service to certain clients, or filter the search results that it transmits to prevent clients from discovering specific content.

[1] URL is an acronym that stands for "uniform resource locator." A URL is simply a name for a Web page that contains enough information to tell a Web browser which server to contact, and to tell the server which page is being requested.

Peer-to-Peer Networks

19. In contrast to the client-server architecture, a peer-to-peer ("P2P") architecture consists of a single class of participant: the peer. Each peer within a P2P network acts both like a client and like a server, in that it requests service from some peers and provides service to others. (See Exhibit 3). Unlike a client-server system, a P2P network does not require any dedicated or centrally managed components for it to operate. In this way, a P2P network is often referred to as being decentralized. To provide functionality to users, a P2P network relies on the cooperation of its voluntarily participating peers.

20. The original Gnutella file-sharing network, circa 2000, is an example of an Internet application with a "pure" peer-to-peer architecture. In the original Gnutella network, each peer was functionally identical. A peer could make files available to the network, initiate search requests to find files of interest, return responses to requests in order to indicate possession of a matching file, and route requests and responses to neighboring peers in order to propagate messages across the network. Once a peer found a file of interest on a remote peer, the requesting peer would transfer the file directly from the remote peer.

21. Unlike a client-server system, no one participant is responsible for providing service in a peer-to-peer network. If any given peer crashes or leaves the network, the P2P network should continue to operate, since the remaining peers still provide service. In this sense, a well-designed P2P network has the potential of being more robust than a client-server system, since there is no single point of failure in the network.

22. As more peers join the network, the total amount of computational, storage, and network bandwidth resources in the peer-to-peer network grows. Accordingly, if it is well designed, a P2P network becomes more powerful as it grows more popular. The capacity,

availability, and diversity of content on the network can scale with the population of peers participating in it.

23. In a P2P network, no one participant has the ability to monitor and control the activities of the users on the network. A given peer usually only interacts with a small subset of the other peers on the network. As a result, it is difficult for one peer to monitor the activities of a specific other peer, since the majority of transactions on the network will not be observable to it. Similarly, while a peer can deny its services to other peers, it is difficult for a peer to cause other peers to deny their services as well.

Hybrid Architectures

24. Centralized client-server systems and decentralized peer-to-peer networks represent two endpoints on a spectrum of Internet application architectures. In practice, many applications can be thought of as hybrids of these two architectures. A notable example of a hybrid architecture was the original Napster file-sharing system. Like most P2P file-sharing services, Napster provided two main functions to its users: the ability to search for files made available by other users, and the ability to transfer files from other users.

25. Napster was commonly described as a peer-to-peer file-sharing application, and indeed a substantial part of how it worked took advantage of a decentralized, peer-to-peer architecture. However, Napster in fact had a hybrid architecture, as its search functionality was centralized. When connecting to the Napster service, a Napster client uploaded to Napster's centralized servers a list of files the client possessed and was willing to share. When a user searched for a file, the user's client software sent a query to the Napster server, which then looked through the lists of files that clients had previously reported to it in order to find a match. (This aggregate list of files available through Napster is commonly called an "index.") The

server then returned a list of matching files, and the IP addresses of the Napster clients that possessed them.

26. To transfer files, Napster clients communicated directly with each other. The client that wanted to download the file initiated a network connection to one or more clients that possessed the file, and the file was then transferred over this direct connection. Thus, while searching was centralized in Napster, file transfers were implemented in a decentralized, peer-to-peer fashion.

27. Because Napster clients had to rely on the centralized Napster servers when searching for files, and because clients reported the list of files they were sharing to those servers, it was possible for the operators of Napster to monitor and control the activities of its users.

28. Not all hybrid peer-to-peer architectures enable monitoring and control of user activity. For example, the LimeWire file-sharing software, which incorporates more recent versions of the Gnutella protocol and specification, takes advantage of a new 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. (See Exhibit 4.)

29. 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.

30. 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.

The Gnutella File-Sharing Network

31. 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.

32. 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.

33. Gnutella is an open protocol, which means that its implementation details are publicly available for everyone to use, and its development and evolution are the result of collaboration in an open forum in which anybody can participate. Anyone can implement the protocol by creating Gnutella-compatible software (a "Gnutella client") that will enable users to access and use the Gnutella network.[2] Because all Gnutella clients are based on implementations of the same, open protocol, a Gnutella-compliant client will interoperate with other Gnutella-compliant clients developed by other entities, thereby forming a single, worldwide network that is described as the Gnutella network. The Gnutella protocol and network was first created in early 2000 by developers working at a company called Nullsoft, which was a subsidiary of AOL-Time Warner. Since then, the Gnutella protocol has been refined substantially to improve the performance, reliability, features, and scalability of the network. Today, there are many Gnutella-compliant software clients. Users of any one of these products can search for and share files with users of the others..

34. The decentralized, peer-to-peer nature of Gnutella has various implications for its operation:

(a) First, its decentralized structure makes Gnutella robust. Because there is no central, single point of failure, the failure or termination of a single peer has virtually no impact on the continued operation of the network as a whole. Even if a majority of peers shut down, the network will most likely continue to function. Indeed, research into peer-to-peer networks has demonstrated that most of the peers that participate in the Gnutella network only remain connected for short periods of time, implying that over the period of several hours, the structure and membership in the

[2] Indeed, many people have developed Gnutella-compatible client software. Wikipedia currently lists 20 different client software packages; see http://en.wikipedia.org/wiki/Gnutella.

Gnutella network undergoes significant change, all the while continuing to operate correctly.

(b) Second, no one entity, including Lime Wire LLC, has the power to control the Gnutella network or to shut it down. In a client-server based system, the central server is typically owned and operated by, or on behalf of, a specific entity (e.g., Yahoo). This entity has the ability to control the operation of the central server, and as a result, this entity can affect the operation of the service. For example, if a central server is accessed or used in an unauthorized manner, the server operator can deny access to unauthorized users, or completely shut down the server. In contrast, because the Gnutella network has a decentralized structure where all peers play similar roles, no one peer has the ability to control the network or to shut it down without the willing acquiescence of the great majority of other peers in the network. In other words, because the Gnutella network is a decentralized collection of individually owned and operated peers, it is self-organizing and self-sustaining, and operates solely based on the collective decisions of its peer members. No one entity or peer has the ability to dictate the operation of the entire network, or for that matter, to completely dictate the operation of any other peer owner/operator in the network.

How Gnutella Works

35. Before a user can participate in the Gnutella network, they must download and install Gnutella client software on their computer. Once installed, the user runs the software and is presented with a user interface that allows them to make files available to others, search for files, and download files that were discovered during a search.

36. To operate, the Gnutella client software must first "bootstrap" itself onto the Gnutella network. Once it has been bootstrapped, client software can send queries into and receive queries from the Gnutella network, and participate in the transfer of files between peers in the network. I will discuss each of these aspects of Gnutella in turn.

Bootstrapping

37. Bootstrapping is the process through which a Gnutella peer finds other peers in the Gnutella network with whom it can communicate. Once a peer has been bootstrapped, it knows of a collection of other peers and ultrapeers that are willing to exchange Gnutella messages with it. A bootstrapped peer is a fully functioning member of the Gnutella network; it has grafted itself into the Gnutella network topology, and it is able to search for, download, and make available files.

38. To help peers bootstrap, the Gnutella network includes some nodes called "UDP host caches", or UHCs, that serve to introduce peers and ultrapeers to each other. By participating in the Gnutella network, over time a UHC learns about peers, ultrapeers, and other UHCs. A peer can ask a UHC for a list of other peers and ultrapeers that the UHC knows of; using this list, the peer can then attempt to initiate a connection to one of these other peers in order to join the network.

39. Gnutella client software will typically ship with a list of UHC nodes. The first time that a user launches the software, the client will contact this initial list of UHC nodes in order to bootstrap itself. Alternatively, most Gnutella clients allow the user to manually bootstrap by typing in the DNS name or IP address of other peers and ultrapeers in the Gnutella network. Once bootstrapped, a peer uses the Gnutella protocol itself to maintain a fresh list of UHCs and other peers. As long as the user continues to use the software periodically, a Gnutella

client will replenish itself with a fresh set of peers to communicate with, as well as UHCs to fall back upon.

Servicing Queries

40. 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.

41. 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"[3] 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 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.

42. The Gnutella protocol is extensible, in that client software can add "Gnutella Generic Extension Protocol" ("GGEP") blocks to existing Gnutella messages. GGEP

[3] 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.

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.

43. 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.

File Transfer

44. 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.

45. 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.

46. 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.

Other Gnutella Mechanisms

47. Over time Gnutella has accumulated additional features, mechanisms, and protocols. Some of these provide alternative bootstrap mechanisms. Others enhance or optimize the way searches and file transfers are performed. In all cases, to the best of my knowledge, these mechanisms all preserve the decentralized nature of Gnutella.

Gnutella Software Distributors and their Lack of Ability to Control User Activities

48. The Gnutella protocol itself does not provide a software distributor with any mechanism to control or supervise acts of direct infringement. Acts of direct infringement involve the peers that transfer those files across the network. The Gnutella protocol itself does not give a software distributor the ability to observe these acts of direct infringement, let alone to control them.

Gnutella Software Distributors and their Lack of an Operational Role on the Gnutella Network

49. A Gnutella software distributor provides client software to users. Once the user has downloaded the software, the distributor need not interact with the user or the user's client software again. Thus, a Gnutella software distributor does not need to play an operational role on the Gnutella network.

50. Instead, the client software can interact with peers, ultrapeers, and UHCs run by other parties participating in the Gnutella network. If a user chooses to perform an infringing act, the user's computer and network, and the computers and networks of the peers from which the infringing file is downloaded, provide the computing resources that power the infringing transfer.

Content Available on the Gnutella Network

51. There is a diversity of content that is made available via Gnutella. Gnutella itself supports the publication and transfer of any kind of content; the nature of the files

that are published, search for, or transferred is determined by the users of the network and what they choose to do. In many regards, this is similar to the Web: users can choose to make any kind of file available by placing it on their Web site, and other users can elect to search for and download these files.

52. As a consequence, Gnutella supports both infringing and non-infringing uses. It is possible to search for, find, and download infringing files. As well, it is possible to search for, find, and download non-infringing files. For example, I was able to use the LimeWire client to interact with the Gnutella network to find, download, install, and run the free, open source "Firefox" Web browser.

BitTorrent

53. BitTorrent is a peer-to-peer technology that facilitates the cost-efficient, high performance, secure, large-scale distribution of content across the Internet. BitTorrent has been optimized to make it particularly suitable for distributing large-sized files, such as Linux distributions, large data sets, and media files. The set of peers that are actively participating in the distribution of a particular file is often referred to as that file's "`swarm."

54. To make a file available via BitTorrent, a publisher must perform three steps. First, the publisher must create a special metadata file called a *dot-torrent* file, and arrange to make it available for users to download. A dot-torrent file contains a "hash" of the file that will be distributed; this allows software to verify a downloaded file has arrived intact. As well, a dot-torrent file contains the Internet name or address of one or more trackers. Second, there must be a *tracker* running at a name/address specified within the dot-torrent file. A tracker introduces peers to each other within the swarm, so that they can upload or download pieces of the distributed file to each other. Third, at least one peer that has a full copy of the file must join the swarm; such a peer is known as a *seed.*

55. To exchange content with the swarm, a peer contacts the other peers it has learned about from the tracker. The BitTorrent protocol specifies how peers inform each other about the pieces of the file that they possess and which they are missing, and it also specifies the manner and circumstances under which peers should decide to give each other pieces. Note that a tracker never participates in the distribution of the file content itself; instead, the primary function of a tracker is to coordinate the introduction of peers to each other within a swarm, while the primary function of peers is the exchange of data.

Comparison with Gnutella

56. BitTorrent is not a full content distribution system; for example, it contains no provisions for helping users find content. Instead, users rely on other mechanisms (such as "torrent search engine" Web sites) to find content of interest, ultimately leading to a dot-torrent file. BitTorrent clients simply transfer files by participating in swarms.

57. As was the case with Gnutella, the distributor of BitTorrent client software has no intrinsic ability to monitor or control the activities of its users. Users are free to download whatever dot-torrent files they wish, and once a user has given a dot-torrent file to her BitTorrent client software, the client interacts only with the tracker specified within the dot-torrent file and the other peers participating in the swarm. The entity that distributes the BitTorrent client software has no participation in this process, and therefore has no ability to observe or control it.

Content Available via BitTorrent

58. Similar to Gnutella, BitTorrent itself supports the publication and transfer of any kind of content, though the protocol is optimized for the efficient and scalable transfer of large files. Users can find a myriad of content offered for transfer through BitTorrent, including both non-infringing and infringing files. Because BitTorrent transfers are powered by peers in a

swarm, rather than by a centralized server, many legitimate organizations take advantage of BitTorrent to distribute large files to many people while avoiding large bandwidth bills.

59. For example, a company called VMware has pioneered the use of "virtual appliances" to mitigate the complexity and security challenges of software distribution. A virtual appliance contains a pre-configured, ready-to-run software application packaged with an operating system. Virtual appliances are typically large -- often hundreds of megabytes in size - and therefore would be expensive to store and transfer from a centralized server. Many of the virtual appliances offered through VMware's "virtual appliance marketplace"[4] are therefore made available via BitTorrent. There are hundreds of different virtual appliances available through the marketplace, including security tools, Web content authoring frameworks, databases, scientific software tools, and educational course management systems.

The LimeWire Client Software

60. 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.

61. 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

[4] The Website for VMware's virtual appliance marketplace is http://www.vmware.com/appliances/.

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. As well, I will discuss the degree to which the LimeWire client software depends on services operated by Lime Wire LLC.

The LimeWire Client is Open Source Software

62. 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.

63. 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.

64. 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.

Downloading and Installing the LimeWire client

65. 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.

66. 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 infringement.[5] (See Exhibit 6.) 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.

67. 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 also not required to share any files in order to use

[5] The dialog box shown in Exhibit 6 is from version 4.18.3 of the LimeWire client software. Earlier versions of the software did not have this dialog box; however, it also used to be the case that the LimeWire LLC web site would prompt the user with a similar dialog box before allowing the user to download the LimeWire client software.

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 sending of any message or other information to/from a LimeWire LLC server.

Searching for Files on Gnutella

68. File sharing activity begins when a user initiates a search for a file.[6] 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 search screen for LimeWire. In addition to containing 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.

69. 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

[6] 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.

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.)

70. 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 is part of the inherent functioning of the Gnutella network. 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.

71. 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.[7]

72. Based on my review of the LimeWire source code, and my observations of LimeWire while executing, it is my opinion that no interaction with the LimeWire LLC servers is required in order for a user of LimeWire to carry out the file sharing operations described

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

above. It is also my opinion that, in its default configuration, no information relating to file searches or transfers between Gnutella peers will flow to any servers operated by LimeWire LLC. As a consequence, LimeWire LLC has no ability to monitor or control file downloads performed by LimeWire peers in their default configuration.[8]

LimeWire Client Enhancements

73. The LimeWire client contains several features and enhancements that augment the capabilities of the basic Gnutella protocol. I will now discuss these.

74. *Viral Messages.* The LimeWire client software, like all Gnutella clients, must make a number of engineering decisions about how it interacts with the Gnutella network. For example, the LimeWire client must decide how many ultrapeers to connect to, and it must tune a parameter that affects how accurately an ultrapeer can determine whether a particular leaf is likely to have files that match a query. The LimeWire source code contains many parameter settings for these and other engineering issues.

75. Consider the case in which LimeWire LLC decides that a new parameter setting might result in better network performance for its clients. One way to change this parameter is to distribute new client software, and hope that existing clients accept and install the updated software. Obviously, this is a costly and time-consuming step to take. Given that parameter settings might need to be updated often, the LimeWire client includes a different, lightweight mechanism for updating these software parameters.

76. LimeWire clients have the ability to receive "viral messages" through the Gnutella network itself. A viral message contains new parameter settings that the LimeWire clients should adopt. To protect against malicious parties from forging harmful viral messages,

[8] I will discuss a featured called "content authority filtering" later in this declaration. This feature is disabled by default. If enabled, this feature does transmit some information about file transfers to a server operated by LimeWire LLC.

each viral message is cryptographically signed, ensuring that only LimeWire LLC (or its authorized parties) are able to create viral messages that LimeWire clients will accept. Thus, to change parameters on its clients, Lime Wire LLC can inject a signed viral message into the network, and eventually all active LimeWire clients will learn of and adopt the parameter changes specified within the message.

77. There are many kinds of parameter changes that can be set in a viral message. Broadly speaking, viral messages can affect: (i) Gnutella network parameters that affect performance and reliability; (ii) filter settings, including lists of "blacklisted" peers that are known to behave in a hostile manner; (iii) some operational aspects of the client software, such as a list of IPs that are permitted to crawl the network and gather statistics about the number of connections each peer maintains; and, (iv) user interface elements such as pictures or URLs that are displayed to users at various times.

78. Viral messages give Lime Wire LLC a limited ability to affect how its deployed clients interact with the Gnutella network. Having examined the various parameters that viral messages can affect, my opinion is that this mechanism does not give Lime Wire LLC the ability to monitor or control the activities of its users. Instead, it gives Lime Wire LLC the ability to tune various performance and operational aspects of client interactions with the network.

79. It is conceivable that Lime Wire LLC could devise a viral message that sets parameters in such a way that clients become dysfunctional. I have not attempted to construct any experiments to verify or refute this possibility, however, were Lime Wire LLC to do so, it would be an abuse of the mechanism as it is intended. It is worth noting that this is

typical of most software systems: unanticipated or maliciously crafted parameter settings and input can cause a system to crash or behave in unexpected ways.

80. *Software Updates.* From time to time, a new version of the LimeWire client software becomes available. The LimeWire client contains an "in-network update" mechanism; a viral message can notify a client that a software update is available, and the client can find and download a copy of the software update using the Gnutella network itself. Using these two mechanisms (a viral message with an update notification and the "in-network" transfer of the update using the Gnutella network itself), Lime Wire LLC can cause a new version of the client software to be pushed out to user's hosts quickly and efficiently.

81. Note that the LimeWire client is capable of discovering and downloading a software update automatically. However, once the update has been downloaded, the user is notified of the update and given the opportunity to approve or reject the installation of the update. Thus, Lime Wire LLC does not have the ability to force an update installation on its users; the user has complete control over whether updates are installed. The fact that the LimeWire client downloads the update automatically serves as a performance optimization, but the downloaded update is only installed with the consent of the user.

82. *Filtering.* The LimeWire client includes several different filtering mechanisms. These mechanisms serve two broad purposes: (1) to defend the client against "spam" files on the Gnutella network, potentially objectionable content, or malicious hosts, and (2) to filter out content that a copyright owner has requested to not be shared. I will discuss each of these filtering mechanisms in turn below.

83. *Junk File Filtering:* In peer-to-peer file-sharing networks, some participants act maliciously by making available to others "junk" content. Junk content purports

to be one file, while in reality is another. Malicious parties might want to distribute spyware, infect peers with viruses, or simply cause peers to waste resources downloading bogus data. To do this, they advertise content using misleading filenames or metadata; a victim that is searching for a piece of content might encounter these junk files that appear to match their search, but in fact contain junk.

84. To help combat junk files, the LimeWire client includes a simple trainable junk file filtering mechanism, similar in mechanism to the spam filter present in many email clients. A user is able to explicitly mark a search result as being junk, or as not being junk. Over time, these user markings serve to train LimeWire's junk filter, so that LimeWire can attempt to predict whether a newly encountered search result corresponds to junk content. Through LimeWire's user interface, the user is able to control whether content marked as junk is hidden, prioritized low on search results, or treated identically to non-junk content.

85. *Keyword Filtering:* The LimeWire client also contains a fairly simple textual keyword filtering mechanism. The client contains a set of predefined adult keywords such as "porn" or "sex." The client also contains keywords associated with file types that users are might choose to categorically exclude from search results; for example, the filename extension ".vbs" is associated with Visual Basic scripts, which are likely to contain malicious code. Finally, through LimeWire's user interface, users can create keyword filters of their choosing. If the user has enabled one of these filters, LimeWire will look for the filtered keywords within the filenames of search results. If any keyword is found, that result is dropped so that the user will not see it.

86. *Host Filtering:* Some peers on the Gnutella network are malicious; for example, some peers systematically forge search results to propagate junk content. To help

combat these malicious peers, the LimeWire client includes a mechanism to filter out Gnutella traffic associated with "blacklisted" hosts' IP addresses. Lime Wire LLC maintains a list of blacklisted IP addresses that is propagated to clients using viral messages. Users have the ability to enable or disable Lime Wire LLC's IP blacklist. As well, users are able to add blacklisted IP addresses of their own choosing through the LimeWire user interface.

87. Conversely, users can choose to "whitelist" IP addresses. If a user has decided to leave enabled Lime Wire LLC's automatically updated blacklist, but wants to make sure that a particular IP address is never blacklisted, they can add that IP address to the whitelist as a way of overriding potential blacklisting.

88. *Content Authority Filtering:* The LimeWire client contains a feature, disabled by default, called "content authority filtering." If a client enables this feature, that client will be prevented from downloading certain content that copyright owners have indicated should not be transferred. When the user installs LimeWire, they are given the option of enabling content authority filtering. As well, the user can choose to enable or disable content authority filtering at later times through the LimeWire client's user interface.

89. Content authority filtering, if enabled, controls whether a LimeWire client will initiate the download of a particular file. When a user selects a file to download, if content authority filtering has been enabled, the LimeWire client will send a content hash (essentially a fingerprint) of the requested file to a "content authority host." If the content authority host responds to the client and instructs it to not allow the transfer, the LimeWire client will refuse to download the file. If the content authority fails to respond, or if it responds affirmatively, the LimeWire client will proceed with the transfer. As mentioned above, by default content authority filtering is disabled in the LimeWire client.

90. The LimeWire client that I examined was configured to use the host "fserv1.limewire.com" as its content authority. (However, the identity of content authority could be updated by a viral message.) I have not examined the logic or source code of the fserv1.limewire.com content authority. However, I can make a few observations given how content authority filtering is incorporated into the LimeWire client itself.

91. First, if a user enables it, content authority filtering causes a message to be sent to the content authority for every transfer attempted by the user. Given this, if enabled, content authority filtering would give the content authority the ability to monitor some aspects of the file transfers initiated by that client. Second, if content authority filtering is enabled in the client, the authority could decide to prevent specific transfers from taking place, granting that authority some control over client activity. Third, there is no mechanism in place that forces a user to enable content authority filtering: the LimeWire client is built in such a way that it ensures the user can leave filtering disabled. Moreover, since the LimeWire client is open source, there is nothing preventing users from excising the content authority filtering mechanism from the source code itself.

Services Operated by LimeWire LLC

92. Certain aspects of the LimeWire client refer to hosts operated by LimeWire LLC. For example, as previously described, the LimeWire client contains the names of various UDP host cache (UHC) servers to provide "bootstrapping" services for newly downloaded clients.

93. To determine the degree to which the LimeWire client depends on services operated by Lime Wire LLC, I ran an experiment in which I configured my computer to block its programs from communicating with IP addresses associated with Lime Wire LLC's computers. More specifically, I took advantage of the configurable firewalling capability of my

computer to add rules that prevent IP packets from being sent to or receive from a list of IP addresses associated with those computers.

94. In my experiment, I initially allowed my computer to communicate with all hosts on the Internet, including Lime Wire LLC's computers. Then, I installed and ran the LimeWire client (version 4.16.6 for the Macintosh), giving it the opportunity to bootstrap itself to the Gnutella network. After this, I enabled my firewalling rules, thereby cutting off the ability of my computer (and with it, the LimeWire client) to communicate with any of Lime Wire LLC's computers. Even after doing this, my LimeWire client was able to successfully interact with the Gnutella network, including performing searches, observing results, and downloading files. In addition, I verified that I was able to quit the client and restart it at a later time (all the while preventing my computer from communicating with Lime Wire LLC computers).

95. From this experiment, I conclude that Lime Wire LLC does not operate any services that are necessary for a LimeWire client to interact with the Gnutella network, assuming that LimeWire client has made it past its initial post-installation "bootstrap."

LimeWire and BitTorrent

96. The LimeWire client version 4.13.0 introduced support for the BitTorrent protocol. More specifically, the LimeWire client has the ability to participate in BitTorrent swarms and transfer files using BitTorrent. BitTorrent support is integrated into the LimeWire client user interface. Users can initiate the transfer of file via BitTorrent through one of three means: (a) typing in the URL of a dot-torrent file into a dialog box; (b) selecting a dot-torrent file found via Gnutella search; or, (c) opening a dot-torrent file downloaded via some external application such as a Web browser.

97. Exhibit 12 shows a series of screenshots in which I used Google to search for a dot-torrent file for the Linux operating system, visited one of the resulting sites and clicked

on a link to a dot-torrent file within it, and as a result, had LimeWire automatically open up that dot-torrent file and begin transferring the associated Linux operating system file using the BitTorrent protocol.

98. Attached hereto as Exhibit 1 is a true and correct copy of my curriculum vitae.

99. Attached hereto as Exhibit 2 is a true and correct pictorial representation of a centralized network structure.

100. Attached hereto as Exhibit 3 is a true and correct pictorial representation of a decentralized, peer-to-peer network structure.

101. Attached hereto as Exhibit 4 is a true and correct pictorial representation of a hybrid peer-to-peer network structure.

102. Attached hereto as Exhibit 5 is a true and correct copy of a screen shot showing a list of Gnutella-compliant software applications depicted on the Wikipedia entry for Gnutella (http://en.wikipedia.org/wiki/Gnutella#Software).

103. Attached hereto as Exhibit 6 is a true and correct copy of a screen shot depicting the dialog box presented to a user when running the LimeWire client (version 4.18.3) for the first time.

104. Attached hereto as Exhibit 7 is a true and correct copy of screen shots of the configuration screens available to users of LimeWire to designate certain user files for sharing over the Gnutella network.

105. Attached hereto as Exhibit 8 is a true and correct copy of a screen shot of the search page used in LimeWire to enter a search string.

106. Attached hereto as Exhibit 9 is a true and correct copy of a screen shot of the search attributes page used in LimeWire to enter search attributes.

107. Attached hereto as Exhibit 10 is a true and correct copy of a screen shot showing search results displayed to a user when searching for "firefox" using the LimeWire client software.

108. Attached hereto as Exhibit 11 is a true and correct pictorial representation of the process by which a search "Query" message generated by LimeWire is distributed across the Gnutella network, "Query Hit" response messages are returned to the LimeWire client, and the process by which LimeWire establishes a direct and temporary connection with another Gnutella peer in order to transfer a file.

109. Attached hereto as Exhibit 12 is a true and correct set of screen shots depicting the act of searching for a dot-torrent file using Google and using the LimeWire client software to open that dot-torrent file and transfer the associated file over BitTorrent.

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 July 17, 2008.

[illegible]

Steven D. Gribble, Ph.D.