Connecting and Extending Peer-to-Peer Networks

LionShare White Paper

October 2004

Table of Contents

Appendices

Table of Figures

LionShare Executive Summary

LionShare is an academic-oriented, peer-to-peer (P2P) networking technology that merges secure and expanded electronic file-exchange capabilities with information gathering tools into a single, open source application. In an era of personal and community knowledge management, the benefits of the LionShare approach are four-fold:

1. LionShare offers a secure, authenticated environment in which users are known to their institution and to each other. Personal collections and community collections can be shared with the efficiency of a P2P network without the threat of unauthorized access or undesired content.
2. LionShare peer servers provide a persistent mirror for content to ensure that designated files can be available for sharing when a personal peer, such as an instructor's laptop, is disconnected.
3. LionShare incorporates international interoperability protocols that provide access to a growing mass of content stored in networks of learning object repositories around the world.
4. LionShare allows publishers to describe their resources using a relevant metadata schema (eg IEEE LOM, etc), and allows searchers to query against this metadata.

In short, Lionshare provides a secure way for institutions to enable and encourage staff and learners to participate in the world-wide exchange of knowledge.

LionShare developed out of Penn State's Visual Image User Study (VIUS) - a 26-month project funded by the Andrew W.Mellon Foundation - that was tasked with assessing how academic communities use digital images for teaching research and service. The study also identified the types of tools and services VIUS participants (faculty, students and collection managers) most desired in a digital image system.

The information collected during the study indicated that current academic collaboration and sharing tools fell short. Specifically, a new application was needed that provided more flexible user-controlled tools and expanded capabilities for the discovery, management, and sharing of multimedia files. The assessment also suggested that peer-to-peer (P2P) technology would offer unique opportunities for expanding and building on the types of tools and services that VIUS participants identified as desirable.

Most P2P technology enables individuals to store and contribute their personal collections for the benefit of a larger community. The VIUS team, motivated by the assessment findings, began considering ways that P2P technology could be adapted for use in the academic environment. The notion of using authenticated P2P technology to securely exchange academic data was compelling enough to launch a project to build a prototype of LionShare. The success of this prototype was significant enough for the Andrew W. Mellon Foundation to award Penn State University a $1.1 million grant to develop LionShare.

At the start of the project, LionShare developers decided to base their code on the Limewire 4.0 Open Source Project's implementation of the widely utilized Gnutella P2P protocol (an agreed-upon format for transmitting data between two peers). While Gnutella technology provides the foundation for LionShare, the Gnutella protocol is only capable of search and retrieval functions. Because the LionShare application needs to perform many tasks beyond basic file search and retrieval, LionShare partners are developing additional capabilities on top of the Gnutella protocol that will support the overall goals of the project.

LionShare, which began at Penn State, now has a number of formal partner organizations including: Internet2, Simon Fraser University of Canada, and the Massachusetts Institute of Technology's Open Knowledge Initiative (OKI). Internet2 is contributing a variant of their Shibboleth technology, which will facilitate the secure sharing of resources between institutions using the LionShare network. The OKI is designing coding standards that will allow easy integration of shared information between participating

institutions and easy access to infrastructure service. Developers from Simon Fraser University are working on technology that will make it possible for LionShare users to access national digital repositories using a single search inquiry.

Below is a brief overview of the extended features currently under development. The remainder of this paper is dedicated to more detailed explanations of these expanded features including: 1) the private LionShare Network Architecture; 2) the LionShare P2P Security Model; and 3) the LionShare Federated Repositories technology.

Network Architecture

The network topology of LionShare is a hybrid of traditional client/server and a decentralized P2P network that will allow the LionShare Peers to search for and retrieve files based on keyword searches. LionShare Peers are truly decentralized in the sense that they will directly communicate and share files with each other rather than through a central server. However, LionShare Peers will also connect to centralized services such as PeerServers, authentication services, and digital repositories. These provide expanded features beyond those of traditional P2P networks. For instance, LionShare will support federated authentication by using ideas from a technology called Shibboleth, developed by the Internet2 Middleware Group. Federated authentication allows users from multiple institutions to connect to one federated and secure network and share resources with specific users, groups, and institutions.

Additionally, the LionShare network gives users the ability to share files persistently by using a technology called PeerServers. A LionShare PeerServer is a place for users to continually share files on the network, even when their Peer is disconnected from the network. On the network, PeerServers appear as just another peer. However, the implementation is designed to provide a robust service, run on servers by IT organizations or academic departments. The LionShare PeerServer is also a centralized location for administrators to make configuration changes, update Peer software, and administer the network.

LionShare also provides organizational tools that can be used to manage personal media collections regardless of a user's willingness to share.

Security Features

Unlike controversial file-swapping tools that can be anonymously used to exchange copyrighted files, LionShare offers an authenticated system designed for secure academic collaboration and legitimate file-sharing. Typically most P2P networks allow users to remain anonymous, LionShare, however, will be require users to create a verifiable identity before sharing. This identity will be attached to the metadata associated with the shared resource and is meant to encourage responsible file-sharing. Through the use of access control lists (ACLs), users will be able to specify the access policies associated with the resources being shared.

Accessing Centralized Digital Repositories Outside of the LionShare Network

Most P2P technologies are closed systems and limit searches within their respective networks. LionShare users, however, will be able to search multiple national and international repositories outside of the LionShare network through the use of the eduSource Communications Layer (ECL) Connector. This feature will provide easy access to an expanded number of specialized learning object repositories not currently available to most faculty and students.

LionShare's 1.0 Open-Source Release

The LionShare 1.0 application is slated for a release in Fall 2005. This will be an open source application which means that any programmer can read, redistribute, and modify the source code. This will allow Lionshare to evolve after its release as programmers will be able to freely improve and customize it.

Introduction

LionShare's Core Vision

The central vision of LionShare emanates from a belief that knowledge exists at a personal level and requires human contextualization for its proper interpretation and application. Thus, the most appropriate place for knowledge to be stored is close to its originator and close to its user. Decentralized P2P is highly flexible and gives individuals the ability to locally hold, organize, control and contribute their personal collections for the benefit of a larger community. This does not rule out the long standing archival and distribution roles of centralized knowledge repositories, such as libraries or portal-style repositories. However, extending a knowledge framework to every member of a community means going beyond simply giving everyone a library card; it means enabling everyone to be a collector and a contributor to their personal and community knowledge pool.

Birth of LionShare

The LionShare concept originated from a number of different ideas and sources. However, a major impetus for its development was the Visual Image User Study (VIUS), a 26-month Mellon-funded project that ended in September of 2003. VIUS was housed in the Penn State libraries and was geared toward exploring how digital images are used for teaching, research, and outreach in academe. The final report of the VIUS project is available online at http://www.libraries.psu.edu/vius/. It is appropriate to give a brief description how the VIUS project helped stimulate interest in the LionShare project.

The most important connection between VIUS and LionShare is that LionShare evolved from a proof-of-concept prototype developed by the VIUS team. VIUS was primarily an assessment project, but several proof-of-concept development efforts were initiated. The VIUS study included extensive interviews surveys, protocol analysis, and focus groups with stakeholders (students, faculty, and collections managers) along with analyses of transaction logs. That assessment was geared towards exploring how individuals and groups were using images for scholarly purposes, where they were finding images, and what would serve them well in terms of digital image systems.

Information collected and analyzed for VIUS identified the need for more flexible, user-controlled tools and services for the discovery, management, and sharing of multimedia files. It also suggested that P2P technologies might offer unique opportunities for expanded yet controlled queries that users desired. Those findings of the VIUS project – regarding what users want and about what technology can deliver – helped lead discussions about P2P and how it seemed to perfectly fit the needs of users. Thus, LionShare was created. In this sense, the VIUS experience provided the inspiration for LionShare by providing substantive knowledge and documentation about the kinds of features that stakeholders want and the concerns that they have, as well as the development of the first LionShare prototype.

The LionShare Application in Brief

Designed to meet the goals elaborated above, the LionShare Peers operate within a P2P network using code originating from the Limewire 4.0 Open Source project (limewire.com). Limewire uses the Gnutella protocol for the basic search and retrieval functions (see Appendix B for a Gnutella details). LionShare will go far beyond this basic functionality by providing extensive additions to the Gnutella base. These additions form the heart of the LionShare project and will extend its functionality and utility in an academic environment far beyond what is now possible with standard P2P applications. The advanced features will include 1) a private and secure P2P network; 2) communication with PeerServers for persistent file sharing and some centralized management; 3) advanced security and networking features such as obtaining authentication related credentials, requesting certified attributes about users, and access control to files; 4) various built-in collaboration tools; and 5) querying independent digital repositories outside the LionShare network. Each of these actions uses its own protocol and these are being added on top of the Gnutella based P2P protocol.

Layout of this Whitepaper

The following sections of this report detail the basic concepts and approaches used to support the LionShare architecture. The report begins with the core, networking functionality of the LionShare Peer application and how it interconnects with the PeerServer. After these basic features are discussed, the additional security/authentication and digital repository features will be explored in more detail.

Section 1: LionShare Peer Application and PeerServer

1.1 Introduction

The LionShare project is dedicated to harnessing the promise of P2P file-sharing and integrating P2P with organizational services to create a collaborative environment for use in academic communities. The LionShare Peer is built around the themes of collaboration, security, personal responsibility and access control of shared resources, and access to large digital repositories. Additionally, this will be an open source application which means that any programmer will be able to read, redistribute, and modify the source code. This will allow Lionshare to evolve after its release since programmers around the world will be able to freely improve and customize it.

1.2 Network Architecture

The network topology of LionShare is a hybrid of a traditional centralized (client/server) network and a decentralized P2P network. This topology will allow the LionShare Peers to search for and retrieve files based on keyword searches and advanced structures based on recognized standards. LionShare Peers are truly decentralized in the sense that they can directly communicate and share files with one another, rather than through a central server. However, LionShare Peers can also connect to centralized resources, such as PeerServers, authentication services, and digital repositories, in order to offer an expanded set of services as described in the following sections of this summary.

Figure 1, *The Basic LionShare Network*, reveals the conceptual design for the Lionshare environment. Local Lionshare *Peers* must authenticate to an institutional *Authentication Service* before sharing files on the local LionShare network on *Institution A*. This process enables the creation of the credentials for sharing on the network. Authentication is not necessary to search or use LionShare tools to organize personal resources on an individual's personal computer. It is only when a resource is requested from another member of the network that authentication is required.

Once authenticated, LionShare members can publish resources to an institutional or departmental *PeerServer* or share files from their Peer application. *PeerServers* provide the ability to keep files on the network continuously, even when a member of the network is not physically connected to the network with their Peer application. Additionally, when a user wants to bridge between two institutions, *Institution A* to *Institution B*, a *trust fabric* needs to be enabled. This *trust fabric* is transparent to the member, but is based on the local authentication process and the ability for institutions to form federations of trusted partners. Access to a resource is enabled by the ability to embed access control language (ACL) within a shared file. With ACL, only individuals with permission to access files are able to retrieve shared files.

Additionally, when individuals wish to find resources in large institutional repositories, like SMETE, RDN or CAREO, LionShare supports federated search of other institutional P2P resources and institutional repositories through the Secure eduSource Communication Layer (ECL). Thus, one search can retrieve local resources (on your own computer), P2P resources (on the LionShare network) and objects from large centralized repositories (through ECL).

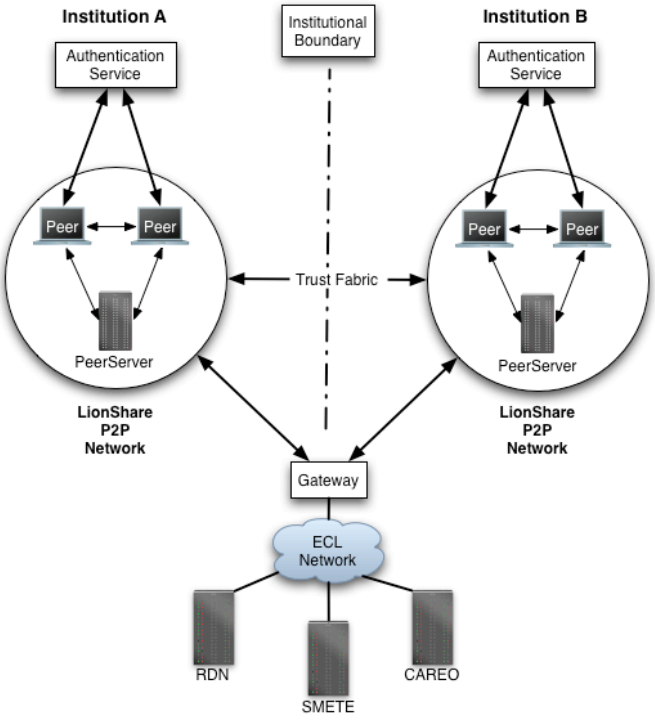

Figure 1: The Basic LionShare Network

1.3 LionShare Peer Application

1.3.1 Introduction

The LionShare Peer is the portion of the LionShare network installed by each individual user. The following illustration provides a detailed description of features of the LionShare Peer. The four main categories of functionality for the LionShare Peer are: Search, Share, Collaborate, and Organize as seen in Figure 2.

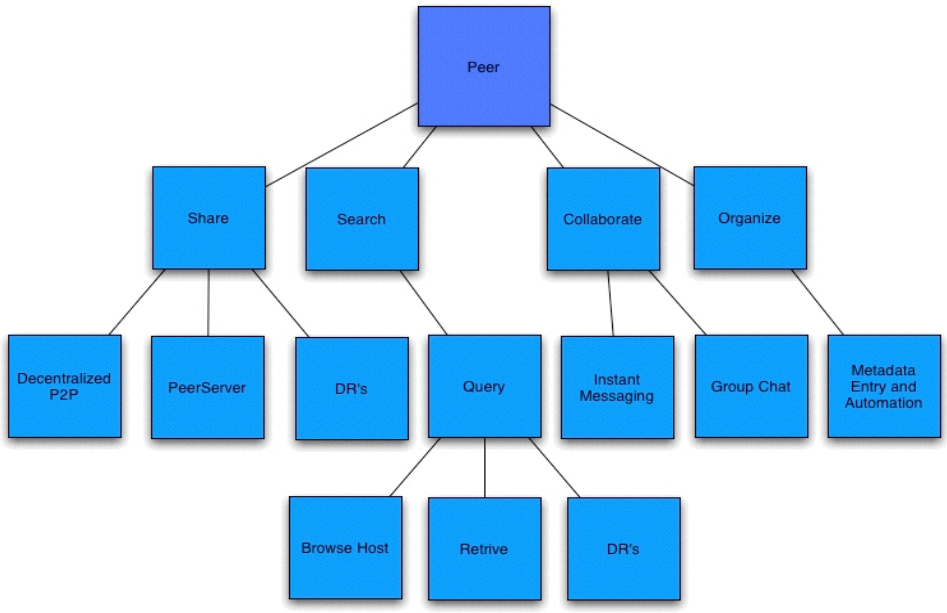

Figure 2: Peer functionality

1.3.2 Search

Central to traditional file-sharing applications are the basic components for search and retrieval. These capabilities are implemented within the LionShare Search function. A LionShare Search is not a typical P2P query. When a user sends a query from the LionShare Peer, the query extends to other LionShare Peers but also to accessible PeerServers and traditional repositories. These are made available to a Peer using the Secure ECL technology.

The LionShare Search interface gives users flexibility when executing a query on the network. For example, a LionShare search can be directed towards a specific media type, such as video, images, documents, etc. Additionally, after selecting the media type, a user can perform a more focused search by selecting a specific field in the metadata schema or searching all the fields of the selected media type by default. A user also has the option of specifying which repositories or networks to search. One user may want to only query specific learning object repositories connected to the LionShare network. Another user may want to search other LionShare Peers exclusively, or just search their own personal local repository. By default, the LionShare Peer searches all the networks and repositories available.

The final search and retrieval feature in the LionShare Search Tab is the Browse Host feature. Browse Host allows a user to view another user's entire shared library. This capability is particularly useful in finding other resources that may be of interest. If another user has one interesting file, it is likely there are other shared files of interest. Browse Host allows a user to see the complete shared collection without having to match specific queries.

1.3.3 Share

The sharing capabilities of the LionShare Peer extend beyond traditional P2P file sharing technologies. LionShare employs a shared library approach for displaying shared files. Users simply add files to be shared directly from their host system to their Shared Library. LionShare also has implemented a few additional options for sharing.

The LionShare network topology is based on a hybrid model that mixes decentralized P2P with traditional client-server network architecture. The element of client-server in the LionShare architecture is known as a PeerServer, discussed in greater detail in the following section of this document. A PeerServer allows users to upload files to a server and share them without requiring users to maintain a connection to the LionShare network. A user does not have to choose one method of sharing over the others; in fact they can simultaneously share the same file on a PeerServer that is shared on their local Peer.

LionShare users can also specify which users or groups have access to specific files and folders. Access control is an important concept in the LionShare paradigm, enabling users to share files with other specific users and groups securely on the network.

1.3.4 Organize

Popular P2P file-sharing tools tend to focus solely on search and retrieval capabilities. LionShare aims to be a more generalized tool proficient at search and retrieval but also useful for organizing resources downloaded or imported from local storage. Media organizational capabilities of the LionShare Peer is another key component of the LionShare model.

Metadata is the descriptive information describing a resource. Within LionShare, metadata plays a vital role, not only in search and retrieval, but also for media organization. Metadata should not be viewed only for sharing a resource with the outside world; it also can serve as a mechanism for individuals organizing their own personal media collections, regardless if they intend to share them with others. LionShare organizational tools greatly enhance the metadata entry process so that average and novice users can easily and painlessly describe their own resources. A major emphasis of the LionShare strategy centers on providing tools for users to automate the metadata entry creation process.

Automated metadata creation is a core part of the LionShare strategy of simplifying the metadata entry process. All files located on a computer's file system have some very basic descriptors, such as file name, size, and type. The LionShare metadata entry process will automatically populate this information to the correct metadata fields.

Furthermore, many different file formats have their own internal metadata located in the headers of each file. For example, the header metadata is stored with each picture taken by a digital camera. Each JPG file in a digital camera contains information about that file, including time and date taken, camera settings, and image information. LionShare harvests worthwhile information from these files and automatically populates this information into the appropriate metadata fields.

1.3.5 Collaborate

Collaboration has long been ignored by traditional file-sharing technologies, especially the popular file-sharing networks on the Web. Furthermore, it has been difficult or impossible for individuals to find specific files stored by other individuals on networks like Kazaa and Gnutella. Likewise, these file-sharing technologies provide little support for individuals, small groups, or large groups to share between one another. LionShare combines file-sharing capabilities with the ability to support file-sharing and collaboration within small groups and teams. On the LionShare network, it is as easy for a user to share one file with one person, as it is to share many files with all users on the network.

Part of LionShare's long-term collaboration strategy is supporting and extending collaboration with real-time tools for instant messaging (IM) and group chat capabilities. Users can message with other users by double clicking their userID to initiate an IM session. Buddy Lists will be supported to encourage collaboration so that users can keep track of other team members online to chat, browse, and collaborate. When adding group chat functionality with the IM and Buddy List tools, LionShare becomes a powerful tool for academic collaboration. Users can also join public chat rooms formed around academic interests. These will be the foundation for like-minded people to meet and share with others in their field. In addition, private chat rooms can be created by users themselves as an excellent option for group collaboration. Users can create a chat room, invite other users, and work together.

Chat rooms are really just an initial step. Although not included in the first generation of LionShare, there are a number of useful capabilities that the LionShare team feels would extend the collaborative abilities of the LionShare environment. The feasibility of including other real-time collaborative tools, such as digital white boards, is currently being discussed by LionShare developers. Digital white boards assist collaboration, further promoting the group interactions within LionShare. The concept of a digital white-boarding has been around for quite some time, allowing users to have a real-time shared collaborative space for writing, drawing, or pasting whatever comes to mind. Having such a tool built-in to LionShare would act as a conduit for team-based activities on the network. Additionally, there is the possibility of including some form of audio and video conferencing capabilities. Both conferencing and white board features are not included in the first release of LionShare, but their feasibility will be evaluated as the project progresses.

1.4 LionShare PeerServer

1.4.1 Introduction

Although P2P systems make efficient use of the substantial storage available on personal hard drives located on the edge of the network, such personal nodes tend to be transient. For instance, laptops are constantly leaving and joining the network from a variety of IP addresses. The lack of persistence on P2P networks is not a problem when there is sufficient redundancy of information on the network. For example, in the case of music swapping, several copies of a given recording can be available at any given time. However, persistence does present a problem when information is more specialized or communities are small. Since LionShare is expected to be used in very specialized academic disciplines, a lack of persistence could potentially be a serious handicap. LionShare handles this problem through the use of a resource called a PeerServer.

The PeerServer is a major component of the LionShare network topology. The PeerServer provides the capability for users to share files on the LionShare network without having to maintain a constant connection. This ability for easy, persistent resource sharing is one of the fundamental concepts that differentiate LionShare from other P2P efforts. The authorization technologies embedded in LionShare allow users to designate the files they placed on a PeerServer as being available for sharing with others or simply as a backup of important resources unavailable to others and accessible only to the owner.

LionShare PeerServers can perform other functions for the network beyond persistent file sharing. For instance, PeerServers can be used as a base of operations for group chat and other collaborative tools. Likewise, for the network administrators, the PeerServer functions as a central location for administrating the Lionshare network.

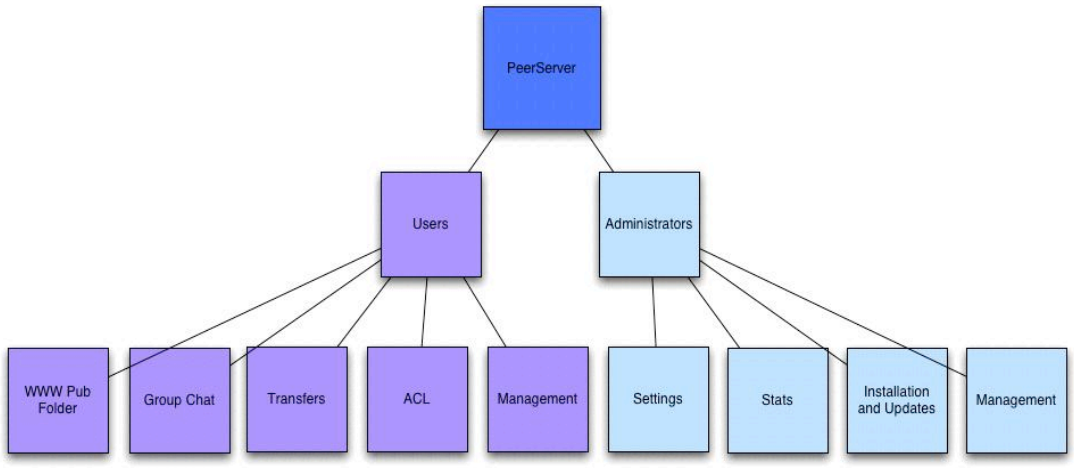

Figure 3: PeerServer Functionality

1.4.2 Users

LionShare users can easily connect to available PeerServers using the LionShare Peer. LionShare Peers sends out an advertisement to other Peers on the network. A listing of PeerServers on the network is returned to the Peer and the user can easily switch between available PeerServers, assuming they have access permissions.

To upload a file from a Peer to a PeerServer, the user simply selects the file(s) or directories to upload, and publishes the resource to the PeerServer. A progress bar displays details on the upload/download process including the transfer rates and time remaining. Once finished, the newly uploaded files are displayed on the PeerServer. Users then have the opportunity to manage their files, assign access control rules, delete the files from the PeerServer, or move files to the Public Folder located on the PeerServer.

The Public Folder addresses two potential complications involved in LionShare file-sharing. These concerns are: 1) the requirement to download the Peer in order to connect to the network, and 2) the necessity of authenticating with a participating institution. The Public Folder is designed to alleviate these concerns by creating a space for files deemed suitable for general public access. Any member of the public will be able download any file in the Public Folder on a PeerServer by using a standard Web browser. The only files available for public download are those placed in a Public Folder by the file owner. The owner will distribute the URL through e-mail, IM, or by any other means to the people who need access to the shared resource. Users can still protect files inside the network by simply not placing them in the Public Folder.

1.4.3 Administrators

For administrators, the PeerServer is a centralized location for network configuration and administration. The PeerServer keeps activity logs of traffic through the LionShare network. Administrators can examine the LionShare network and diagnose and debug problems using administrative tools on the PeerServer. It can also record these network usage statistics in order to provide useful information for research and understanding use patterns. Administrators can view PeerServer logs and change settings using a Web-based administration interface. The PeerServer can also be used to provide timely information to network users. The Web-based administrative interface provides tools for monitoring quotas, users and bandwidth. Because of the decentralized nature of the LionShare network, the best way for administrators to change network behavior is by sending updates directly to the LionShare Peers throughout the network.

Periodically, the LionShare Peer will check the closest PeerServer for an update and notify the user if an update is available.

1.5 LionShare Peer/PeerServer Conclusion

As mentioned previously, LionShare Peers will utilize the basic search and retrieval features from the Limewire 4.0 implementation of the Gnutella P2P protocol. In order to offer its expanded feature set, the LionShare application will need to perform many tasks beyond simple file search and retrieval. These include functions such as obtaining authentication related credentials, requesting attributes about users, or querying independent digital repositories. Each of these tasks requires its own protocol and these are being added on top of the Gnutella P2P protocol. The following sections of this report detail the concepts and approaches for these other protocols.

Section 2: LionShare Security Model

2.1 Overview

Trust is the cornerstone of sharing. It is a human trait established between individuals or organizations and lies outside of the technological solution. Technology merely serves to certify and verify trust relationships. The basis of trust is either direct reputation, between parties known to each other, or indirect, when the parties involved have a trust relationship with a mutually known third party. If trust is not present, then cooperation and sharing are difficult to achieve. In short, trust is very closely allied with notions of security, authority to hold or to view content, and identification of users. To date, P2P systems for learning object exchange have had only rudimentary security features to address these concerns.

In order to create a legitimate and collaboration-friendly environment, the LionShare project has designed its security model based on the following design requirements:

- Individuals should never be able to share files anonymously to prevent file-sharing abuses
- Individuals should be able to search the LionShare network anonymously due to privacy concerns
- File owners must have the ability to control who can access their files based on certain verifiable attributes. Files can be shared with specific individuals, or people possessing certain attributes (e.g. members of a department).

The LionShare security model is designed to prevent users from impersonating others on the network by stealing their private keys, by obtaining certificates or attributes of another user, or by generating fraudulent attributes. There will also be measures in place to prevent a protected file from being intercepted during transfer over the network.

The following sections briefly describe LionShare's processes for authentication, authorization, access control, and the use requirements that have been established concerning network security and privacy. Please see Appendix C for more detailed security technical details.

2.2 Authentication and Authorization

LionShare's security model revolves around the issues of authentication and authorization. Authentication is the process of identifying an individual and ensuring that the individual is who he/she claims to be. Once authenticated, authorization determines the level of resources and network services that the user is permitted to access. Traditional P2P applications do not contain such features.

To use the LionShare network, users must authenticate with their home institution to establish their identity. As LionShare is an inter-institutional network, it is important that users authenticated at one institution be able to assert their identity to users at other institutions. These remote users can then decide to release files based on trusted attributes from the requesting user's institution. This model is called federated authentication. LionShare supports federated authentication by leveraging components of Shibboleth, an open-source middleware package developed by the Internet2 Middleware Group.

A LionShare network will likely involve users from many different institutions, each of which may use different technologies to authenticate users. Given this reality, LionShare needed a common identification mechanism which could work across institutional boundaries. Public key infrastructure (PKI) was selected as the most flexible solution. The LionShare PKI can use existing educational federations, such as Internet2's InQueue and InCommon federations.

LionShare places a strong emphasis on the user's privacy and will only reveal the user's identity or attributes when required (and when the user has authorized such a release). To facilitate this privacy protection, each LionShare Peer is provided two distinct certificates: a client certificate and a server certificate. The contents of the certificates are different in order to withhold or share the user's identity when in the different roles of either searching for or sharing files.

Both the client and server certificates contain information about the authenticating institution. Further, the certificates act as a digitally signed assertion from the user's institution that the user has authenticated and is allowed to both share and search for files. The server certificate allows the user to share files on a network and contains the user's identity. The client certificate allows the user to request files from other Peers on a network, but does not contain identifying information.

2.3 Access Control

Files on a LionShare peer may be shared publicly or they may have access restrictions placed on them. This decision can be made on a file-by-file basis. These restrictions take the form of attributes a requester must hold, such as belonging to a particular class or being a faculty member at a particular institution. Multiple such restrictions may be placed on an individual file for fine-grained control. The list of restrictions is called an Access Control List (ACL).

ACLs are private in LionShare. When a user creates an ACL, it is stored only on the local LionShare Peer or on a PeerServer; it is never sent to other users, as doing so could leak sensitive information. For example, a user may want to share a file with only three other users, but he may not wish to reveal to everyone the identities of those three users.

To facilitate this privacy requirement, the ACL is stored in two parts. The complete ACL contains the list of required attributes and their required values. This is stored internally on the LionShare peer sharing the file. Publicly, only the list of required attributes is revealed; this list is stored in the file's metadata, which is public. This two-part ACL allows a user requesting a protected file to retrieve the appropriate attribute assertions from his home institution and protects the privacy of the user sharing the file.

Section 3: LionShare Federated Repositories

3.1 Introduction

P2P networks have architectures that traditionally constrain information exchanges to nodes within the network. However, the LionShare project will expand its boundaries to include resources, such as centralized learning repositories, that are outside of the LionShare network through the use of the eduSource Communications Layer (ECL) Connector. Many valuable learning resources and services lay out of reach behind protective network security measures. The ECL Connector will make it possible for LionShare to conduct federated searches of these disparate learning object repositories by interfacing with these various security measures. Federation implies that a transaction initiated in one system can be seamlessly transmitted to and completed in another system.

As mentioned in the preceding section of this report, one goal of LionShare is to provide authenticated P2P communication between its Peers. It is important to note the differences and interconnection between the authenticated P2P communication and the ECL connector. The authenticated Peer connections will allow for trusted and secure connections between LionShare Peers throughout the LionShare network regardless of where the user authenticated. The ECL connector will sustain this authentication and expand it beyond the LionShare network to other learning object repositories.

The recent SPLASH project has demonstrated the feasibility of this approach. For this eduSource Canada Project, the SPLASH team developed the eduSource communications layer (ECL) as a means of linking, or "federating," a wide variety of learning object repositories within the SPLASH P2P system. Hatala and Richards (2002) demonstrated how SPLASH could affiliate with centralized servers (PONDS) to form a hybrid network of personal and community collections. In a follow-up project, eduSource Canada extended this concept to build connectors and gateways that could link disparate communities into a global network capable of searching and exchanging learning objects. For a more detailed background of the ECL Connector please see Appendix D.

LionShare will advance this to the next logical step in this process by providing authenticated access to those resources. To reach this goal, additional work has to be done on the existing ECL Connector to enable a broader range of security transactions across the ECL. Currently, the ECL can pass along simple identity tokens and passwords but it is far from being a secure system.

To reach the goals of the LionShare project, the ECL protocol will need to be extended to include a security layer that will allow the gateway to support mapping between security mechanisms used in each of the connected networks. The Secure ECL Connector will implement such security extensions to allow for quick, secure connections of the repositories into the LionShare network.

The goal of the Simon Fraser University (SFU) Surrey team is to extend the LionShare community to other networks by integrating the ECL into the LionShare architecture. This means extending not only the relaying of messages to LionShare, but also ensuring that the security principles of LionShare – the secure environment for file sharing – are maintained.

3.2 Integration of ECL in LionShare

The SFU Surrey team has four proposed tasks in integrating the ECL into a seamless P2P trust environment:

3.2.1 Secure ECL – Currently the ECL is completely open, so there is no well defined security layer

and anyone can search any site that is connected via a gateway or a connector. There is only a primitive provision for a system that connects to the ECL that wants to place a restriction on which search requests it will honor, via a user login-password mechanism. The user has to have login permission from the destination repository in advance rather than during a search transaction. To enable other security aspects, such as privacy and trust, the ECL has to be redesigned. A system, through its own resources, may restrict functionality to a limited set of functions – e.g. search is allowed but only metadata will be returned, not the actual object. More flexibility is desired so that a repository may selectively respond to requests and determine which level of service it will offer to which users. To make the ECL secure the protocol must first be redesigned with the explicit focus on the security. A security mechanism must be designed and have it integrated with the SOAP protocol.

3.2.2 Secure ECL Connector (SHEC) – In fulfilling the security discussed above, the connector middleware needs to be extended to support the selected security. The goal is to provide an API for the connector implementers that will hide the complexity of the authentication that is required for the secure exchange of messages.

3.2.3 Secure ECL Gateway (SHEG) – In addition to mapping between the content part of protocols, it may be necessary to have a security crosstalk so that a user request substantiated using one security mechanism can be relayed to a destination that uses another security mechanism.

3.2.4 Secure OKI/ECL Plug-in – The OKI OSIDs approach to interoperability is to provide a plug-in component that connects user parts of systems via standardized APIs with backend or networked parts of the system. Three OSIDs are particularly relevant to the work: a) Digital Repository (DR) OSID, b) Authentication OSID and c) Authorization OSID. In this part of the proposed work a plug-in will be built that will integrate all three OSIDs for communication using secure ECL protocol. By using the plug-in, any OKI OSID enabled system will be able to participate in the eduSource/LionShare network.

3.3 Conclusion

The development and integration of the ECL connector into the LionsShare client will enable secure communications across the LionShare community and enable access to secure resources and services. As the interconnection of learning object repositories and networks increases the number and kind of resources visible to learners also increases. The enabling of secure interconnections should facilitate smoother access to learning resources and services that have been protected by secure-use policies. The extended ability to authenticate beyond the LionShare network will facilitate the extension of human trust networks and enable community treaties for exchange and access to learning resources.

Section 4: LionShare Assessment

4.1 Introduction

LionShare emerged from the Visual Image User Study (VIUS) – a 26-month Mellon-funded project ending September 2003 that was geared toward exploring how digital images are used for teaching, research, and outreach in academe.

The most important connection between VIUS and LionShare is that LionShare evolved from a prototype developed by the VIUS team. VIUS was both a technology development project and an assessment project. The VIUS study included extensive interviews surveys, protocol analysis, and focus groups with stakeholders (students, faculty, and collections managers) along with analyses of transaction logs. That assessment was geared towards exploring how individuals and groups were using images for scholarly

purposes, where they were finding images, and what would serve them well in terms of digital image systems.

The VIUS experience provided a foundation for LionShare by providing substantive knowledge and documentation about the kinds of features and concerns that stakeholders have and by demonstrating how integrated, systematic assessment can contribute to the success of projects. Drawing from that experience, the LionShare team will conduct a variety of assessment activities that will inform development of software to meet the needs of stakeholders.

4.2 Two Assessment Dimensions

There are two very different, though related, sets of challenges that the assessment process of LionShare addresses: first, system architecture and second, the "economics of sharing."

4.2.1 System Architecture

System architecture assessment addresses the many technology-related questions important to the functionality of LionShare. This dimension entails questions about scalability, stability, functionality, authentication, and so on. The system's design and programming challenges are substantial, and evaluating the ability of LionShare to serve the needs of users is important and necessary. Nonetheless, in terms of assessment, this is a relatively straightforward task – determine what features LionShare should include and how well the features work?

4.2.2 The Economics of Sharing

A more problematic assessment need arose from some of the thorniest issues identified by the VIUS project – namely, the scenarios under which people are, and are not, willing to share files. Throughout VIUS, users consistently raised a host of questions about motivations and conditions for sharing, ownership, ethical and legal responsibilities, intellectual property and licensing. Those issues are referred to here under the umbrella term, the "economics of sharing."

P2P networks in academe exist in a special environment. The most successful P2P systems to date have been characterized by very large numbers of users with a strong common interest (usually popular music.) But how will this technology be used in the academic environment for scholarly purposes, where interests are typically more specialized and the potential user populations are smaller? Who will be the most intensive users? What will motivate people to share? With what restrictions? What tools will be the right ones to foster sharing?

Overlaid on questions about the incentives for people to share are questions about intellectual property. How LionShare should address these matters is not entirely obvious at this point to members of the LionShare development team. Neither is a clear answer necessarily available from informed experts in this field.

In short, LionShare – in decisions about the system, and in user behavior – needs to play by the rules, but at present it is not completely clear what those rules are or what they should mean for the design of the LionShare P2P network. The economics of sharing will be a front-and-center issue for the assessment aspects of LionShare.

4.3 Milestones of LionShare Assessment

4.3.1 "Economics of Sharing" Web Survey

LionShare's first Web survey will focus on the issues that this report has called the "economics of sharing." As noted, we expected these issues to be fundamental to some aspects of LionShare's design and implementation. Therefore, the "economics of sharing" Web survey was an early data-

gathering effort conducted during summer 2004. The assessment team has already developed and implemented the first instrument designed to explore the sharing of an individual's repository resources.

4.3.2 Multi-Pronged Assessment of the Application User Interface

For each Beta release of the LionShare client application, there will be a three-pronged assessment approach: focus groups, interviews, and Web surveys. The Assessment Team's plan is to fully implement this approach for all Beta releases beginning with the first one in January 2005 and continuing up to the final release at the end of the project. Additionally, for the non-public, Alpha release on September 30, 2004, the assessment team will conduct as many of these activities as is feasible. The release schedule is detailed in the timeline later in this document. These assessment activities will include both Penn State and non-Penn State faculty and staff, as well as students. In addition, any available network-use log data from PeerServers will be analyzed; these logs will compile data on file type, file size, and frequency of use (most popular file, type of file, number of users, and so on).

4.3.2.1 Focus Groups/Protocol Studies

For each operable interface, a small set of about six to ten users will be invited to a demonstration and focus group discussion. The VIUS project team will also use protocol analysis for some of its user-software evaluations. Protocol analysis may be utilized, depending on the stage of development of the LionShare software. This would give the participants the opportunity to try out the interface, to manage and publish their own multimedia files, and to access other objects and metadata.

4.3.2.2 Administrator Interviews

As noted in other LionShare documentation, a major goal for this project is to federate P2P networks among several institutions. Clearly, there will be significant technical obstacles as LionShare is extended. It is anticipated that by January 2005, the first release of an application for intercampus installation should be available. At about that time, members of the LionShare assessment team will conduct structured one-on-one interviews with administrators from three to five other universities to explore matters such as the ease of installation, user interface, tools, whether and how well setup and configuration work, what is not understandable and what is missing.

4.3.2.3 Web Surveys

Once it is possible for users to download and install the application to their own computer, there will be Web surveys designed in order to measure the experience of these users.

4.4 Final Assessment Summary

Assessment results will be incorporated into the final report of the LionShare project in September 2005. This final report will summarize all the assessment activities described above and will incorporate any transaction log-type analysis. The final evaluation will not only summarize the previous assessment but will focus on "gap analysis" – that is, to examine what the project and software have been able to accomplish, and to what extent users needs have and have not been met.

Whitepaper Conclusion

When released, LionShare should be a novel form of P2P networking that will provide unprecedented search, retrieval, collaboration, and security features. The preceding white paper represents the goals and methods that the various LionShare teams are working towards at the time of writing. Over the course of events, various technical insights, improved methods, assessment analyses, and Beta testing feedback may result in a refinement of these approaches. LionShare's development is still in a fluid state so as to continually improve up until the final release in Fall 2005. Prior to the final release, there will be a couple of public, Beta test releases. These are currently expected to occur in January and May 2005.

Appendix A

Glossary
LionShare Application and PeerServer

Bittorrent – A file sharing protocol specializing in the distribution of large files, Bittorent is innovative for supporting advanced swarm and partial file downloading techniques.

Federation – A group of independent authentication realms agreeing to interoperate under certain terms.

Gnutella – A decentralized peer-to-peer protocol for search and retrival.

Hash URN Gntulla Extensions – HUGE is a collection of Gnutella extensions that allow for files to be identified and located by uniform resource names..

LionShare – A federated P2P network for collaboration among faculty, students, and staff on Academic networks.

OSIDS – Open Source Interface Definitions, a generalized API to create a level of abstraction to encourage interoperability among applications and projects in higher education.

Peer – An individual user on a P2P network or the name of the "client" application running on an individual's computer.

PeerServers– Always-on machines on the LionShare network that let users share files persistently, allowing them to keep sharing files when logged off the network.

Peer-to-Peer (P2P) – P2P is a communications model in which each party has the same capabilities and either partty can initiate a communications session.

Servent – Gnutella terminology for a Peer, the names client and server are bound together to form "servent."

Shibboleth – An internet2 middleware technology for enabling federated authentication and authorization between multiple intuitions using a variety of authentication methods.

Swarm Downloading – The ability to download one file in sections from multiple users at the same time in order to speed up downloads hindered by slow upstream bandwidth.

Appendix B

Gnutella History
LionShare's Peering Protocol

Introduction
The LionShare protocol is based on the widely used Gnutella protocol developed by Nullsoft Inc's Justin Frankel. The first public version of the Gnutella protocol was released in March of 2000. At the time, Justin Frankel was employee of a recent AOL acquisition called Nullsoft. The Gnutella release happened without any approval from the AOL corporate hierarchy and the project was promptly pulled from the Nullsoft website. During the short window of time that the original version of Gnutella was available, thousands of people downloaded a copy. Within a few weeks, the protocol was well documented, the source code was released, and several Gnutella compatible applications were released to the public.

Gnutella Summary
The original version of the Gnutella protocol was simplistic but innovative. The Gnutella protocol took the Napster file sharing concept one step further by making the process completely decentralized. Gnutella was the first file sharing application that was not dependent on servers or any one organization. The first Gnutella servants were innovative, but certainly not in the running for any usability awards. The early Gnutella clients required users to go to a website called a "host tracker" and copy IP addresses in to their Gnutella client in order to connect to the network. Searches were very crude, unreliable, and prone to many problems. The connection scheme for the original Gnutella network was not aware of bandwidth differentials. Users on direct high speed Internet connections ran the risk of being routed to the Gnutella network through high latency modem connections.

The first release of the Gnutella network was an unfinished work that developers attempted to use as a production protocol. This led to a bad reputation among users for many Gnutella applications and the protocol in general. Forgotten was the fact that the original protocol spec was a pre-release version that was not intended for wide use but rather as a foundation for further development.

Since the original release of the Gnutella four years ago, the protocol has been further refined and developed resulting in greater networking efficiency. The Gnutella Developers Forum (GDF), a group of various developers and P2P companies, handles extensions to the Gnutella protocol in an open, democratic process. The current Gnutella protocol is at version 0.6 but most popular Gnutella servants use a variety of extensions to the protocol. The Limewire core, the base for the LionShare protocol, has implemented and created many of the available Gnutella extensions. Using the Limewire core as the base for Gnutella messaging, LionShare takes advantage of all the latest working extensions to the Gnutella protocol, along with additional modifications to the protocol when needed.

Currently, Gnutella supports many features that were noticeably absent with the first release in the Spring of 2000. One of the common user complaints of file sharing networks was that queries had to match the exact filename. Descriptive rich queries are now supported on the Gnutella network thanks to support for XML schemas. Users now have the ability to add metadata to their shared files and search for specific descriptive information in metadata fields.

As mentioned before, connecting to the Gnutella network was once a laborious process but the advent of the Gnutella webcache and Ultrapeer specifications have improved the protocol tremendously. Now, Gnutella servants connect automatically to the network and low-bandwidth modem connections are

routed through the Gnutella network via high bandwidth Ultrapeers. In today's Gnutella network, queries are not constrained by low speed connections thanks to bandwidth detection and prioritization code.

Another common user complaint with early Gnutella servants was the fact that a popular resource may be shared under multiple filenames confusing the user and leading to redundant downloads. A single file being shared under multiple names on a Gnutella network is no longer a problem thanks to hash file identification. Most Gnutella servants now support the Hash URN Gnutella Extension (HUGE).

There are quite a few Gnutella enhancements related to transfer and browsing of files. The browse host extension for Gnutella, allows a user to browse another user's complete shared collection. This was a widely used feature of the original Napster software that was noticeably missing from the early Gnutella servants.

Besides searching, connecting, and browsing, file transfer is another aspect of Gnutella that has greatly improved. Gnutella servants now have the ability to download the same file from multiple users at the same time using a feature called “Swarm Downloading”, which can speed up download times tremendously. Users can even download from servants that only of have a portion of a file available. Both swarm downloading and partial file downloading give advanced Gnutella servants features similar to BitTorrent. BitTorrent is a popular file sharing protocol specializing in the distribution of large files.

Appendix C

LionShare Peer-To-Peer Security Model
Technical Details

Obtaining Credentials

[0]LionShare will use the federation trust frameworks pioneered by Shibboleth to establish trust between requesting and publishing peers. The current implementation uses PKI technology to identify parties and establish trust. In addition, LionShare is using SAML assertions in ways not intended by the SAML designers. Consequently, LionShare will map the user-supplied credentials to a set of temporary PKI credentials, and then use the PKI credentials to prove that the SAML assertions refer to the user.

The LionShare protocol uses X.509 certificates for authentication between peers. Each peer has two certificates, a "server" certificate which allows the user to share files on a network, and a "client" certificate which allows the user to request files from other peers on a network. The "client" certificate is also used to obtain credentials about the user from a modified Shibboleth Attribute Authority (AA).

The X.509 certificates are obtained from a SASL-CA -- a certificate authority which issues temporary "junk" certificates to LionShare users. Since each institution in the federation may be using a different local authentication system, a common system was needed for use between LionShare peers who may be at different member institutions. X.509 and SSL were chosen as this common authentication mechanism, since the LionShare protocol's file transfer protocol is HTTP-based.

LionShare uses certificates very differently from traditional X.509 applications. Its certificates are created when the LionShare application starts and are destroyed when the application is shutdown. The "client" certificate has a very short lifetime (8-10 hours), and private keys are never committed to permanent storage. Certificates are also specific to each instance of the LionShare peer on the network. For example, if a user is running LionShare on two computers, there will two sets of "junk certificates" for the user.

To obtain these "junk certificates," the user authenticates to the SASL-CA by using the SASL (Simple Authentication and Security Layer) protocol[1]. SASL allows two peers on a network to negotiate at run-time which authentication mechanisms they support[2]. The SASL-CA is very similar to the NMI KCA[3] but adds support for multiple authentication mechanisms. Once authenticated, the user generates two RSA keypairs on his workstation, one each for the "client" and "server" certificates (this is performed automatically by the LionShare application). The public keys from these keypairs are then sent to the SASL-CA, which generates the appropriate certificates. The SASL-CA may query an enterprise directory service, such as LDAP, to populate fields in the certificates, such as DN. Each host institution may set policies for acceptable authentication mechanisms, certificate lifetimes and acceptable public key algorithms and lengths.

Verifying Credentials

Since the file transfer component of the LionShare protocol is HTTP-based, much of the credential verification may be performed by standard SSL libraries available for Java. However, an X.509 certificate must be rooted in a trusted CA. To validate these certificate paths, each LionShare peer must have a copy of each federation member's SASL-CA signing certificate in its trusted CA bundle. This bundle file on each peer must be periodically updated, possibly through an automatic mechanism from a secured web site. Generating and maintaining this bundle is presumed to be covered by the federation's policies and is out of scope for the LionShare project. The LionShare application will support multiple certificate bundles, so that an individual peer may belong to multiple federations and users may import their own

certificates.

LionShare does make one change from canonical X.509 certificate path validation: It does not check if the peer's certificates are revoked. As these certificates are only valid for a very brief time, it was decided that revocation checking was not necessary. Further, a scalable mechanism of revocation checking was not identified.

LionShare also handles expired credentials in an atypical way. Please see the section on Metadata Signature Verification for more information.

Certificate Contents
Each LionShare peer has two credentials, a "client" certificate and a "server" certificate. The contents of these certificates are different in order to protect the user's privacy. LionShare's use cases stipulate that it should not be possible to share files anonymously. To search for files, one must authenticate and obtain certificates, but it should be possible, however, to search for files without giving away one's identity. This is the "pseudononymous" user ability adopted from Shibboleth[4].

Both the "client" and "server" certificates contain information about which institution issued the certificate (encoded in the Issuer field). In this regard, the "client" certificate acts like a SAML authentication assertion -- it is a digitally signed assertion that the user has authenticated, but it does not reveal the user's identity. The "server" certificate does contain the users's identity; its Subject field contains the user's userid (such as an eduPersonPrincipalName) in the CN field. The client certificate contains an opaque handle for the user. The client certificate is used to open an SSL session with the Attribute Authority, which uses the opaque handle to identify the user.

Metadata Signature Verification
All files shared on a LionShare network have an associated metadata structure. This structure contains, at a minimum, a cryptographic hash of the file[5], the "server" certificate of the last user to alter this file, as well as the certificate chain for this "server" certificate. Most files' metadata structures will also have "traditional" metadata (such as ID3 tags) encoded as a stream of XML. The format of this metadata (Dublin Core, IEEE-LOM, etc) is not specified by LionShare. Crucially, this entire metadata structure contains a digital signature of itself; this signature is generated by the peer's "server" certificate.

These metadata structures are stored in a relational database inside the LionShare application. When a user alters a file's metadata structure, either by editing the structure itself, or by altering the file (which would change the file's hash), the metadata structure must be re-signed. This is done using the user's current "server" certificate. No attempt is made to track revisions to a metadata structure.

This scenario does present a problem. As discussed in the Obtaining Credentials section, a user's "server" certificate is destroyed when the user closes the LionShare application, but the metadata structures (and their digital signatures) are not destroyed at shutdown -- they are persisted into an embedded relational database. When the user next launches LionShare, a new "server" certificate will be obtained from the SASL-CA. However, the metadata structures are not re-signed with the new "server" certificate, since re-signing large filesets would be prohibitive. Instead, the metadata structures (and their digital signatures) are retrieved from the database and made available for other peers to query.

Therefore, it is possible that a user will receive a metadata structure signed with an expired certificate. In traditional X.509 certificate processing, this would raise an error. In LionShare, expired certificates are allowed. If the metadata's digital signature is valid, then the included "server" certificate must have been used to sign it. Since a valid certificate was used to sign the metadata, it is assumed that the metadata is legitimate.

Policy Management
Files on a LionShare peer may be publicly available, or they may be shared only to certain other users. All files on a LionShare peer have an associated metadata structure. As discussed above, this metadata often includes an access control list (ACL) which details the attributes a requester must possess to obtain the file. This ACL is expressed in XACML[6]. It contains both the required attributes and the permissible values for those attributes. This policy should not be released to other peers on the network; doing so would leak sensitive information. To prevent this, a peer will translate the XACML policy into a SAML attribute request. The latter format only lists which attributes a peer requires, but not the values of those attributes. Ideally, a "server" peer will include the SAML "version" of the ACL in a QueryHit message so that a "client" peer can extract it from the QueryHit stream and forward it to its Attribute Authority. See the section on "LionShare Protocol Flow" below for more information.

Policies are not transferred with a file. When a user downloads a file from another Lionshare peer, he receives only that file -- he does not receive the XACML policy for sharing that file. This is to prevent information leakage. Once a file is downloaded, a LionShare peer will automatically reshare that file if it was public; if is was protected, LionShare will not automatically reshare it.

Authorization
Authorization decisions in LionShare are made using XACML. We have tentatively decided to use Sun's open-source SunXACML engine[7]. XACML engines work by comparing several attributes against a policy document (the ACL). In LionShare, we are using SAML signed attribute assertions to provide attributes to our XACML engine. In order to remain portable, we needed to limit the number of possible attributes that users could use to build an ACL. For the initial release of LionShare, we are only going to support the eduPersonPrincipalName (EPPN) and eduPersonScopedAffiliation attributes, as these are the only attributes in wide deployment across institutions. Additionally, we may allow users to use Department (OU), but this attribute would only work within an institution.

Further, we need to extend the Shibboleth Attribute Authority to sign attributes using the holder-of-key method. This is a technique by which a signed attribute assertion is bound to the holder of an X.509 certificate. This will prevent attribute assertions from being spoofed by other peers on a LionShare network. The attributes will be bound to a LionShare peer's client certificate.

LionShare Protocol Flow
The numbered points below refer to the numbered lines in the figure. Steps 1-3 are performed by each peer on startup, but for simplicity’s sake, only the interactions with one peer are shown. Steps 4-7 are performed for each query on the network, and steps 8-12 for each file retrieval.

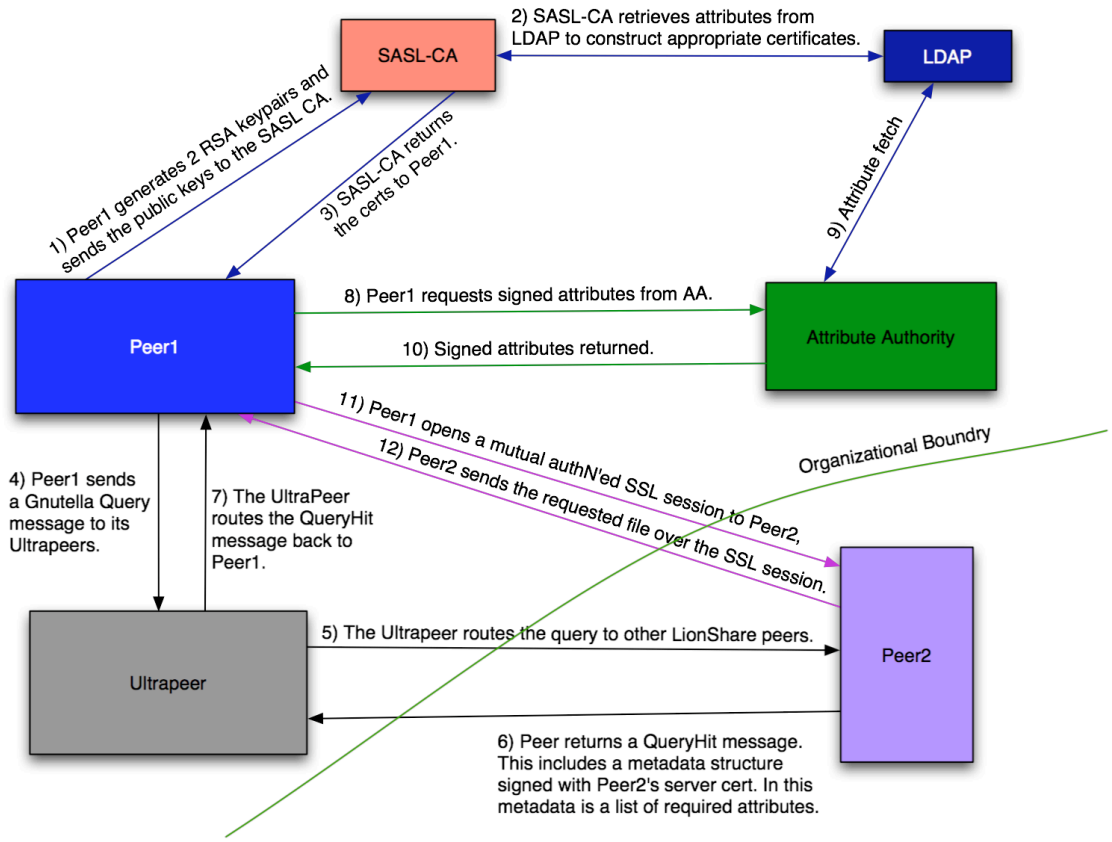

Figure 4: LionShare Security Protocol Flow

1. Each Peer on the network must obtain its client and server credentials, as detailed in the "Obtaining Credentials" section above. During this step, each peer opens a connection to the SASL-CA and performs a "SASL handshake" – that is, it negotiates which authentication mechanism to use, and then authenticates itself to the SASL-CA. The peers then generate the RSA keypairs and send the public keys.

2. The SASL-CA, having authorized Peer1 to obtain a certificate, retrieves any necessary attributes from the institution's LDAP server. These attributes may be used to populate the DN field of an X.509 certificate.

3. The SASL-CA generates the requested certificates and sends them to Peer1. The connection to the SASL-CA is then closed.

4. If Peer1 wishes to search for a file, it will send a request to each of its Ultrapeers. To simplify the diagram, only one Ultrapeer is shown here; normally, each regular peer would query 1-5 Ultrapeers.

5. Each Ultrapeer maintains a list of many regular peers on the network. It forwards search queries to these regular peers on behalf of other peers.

6. If Peer2 has a matching file, it will generate a QueryHit message. This message includes the signed metadata for the file (see "Metadata Signature Verification" above), as well as its server certificate.

7. The Ultrapeer routes the QueryHit message back to the originating peer.

8. Peer1 collects QueryHit messages from multiple peers. It displays a list of these QueryHits to the user, who then selects one or more files to download. As part of each QueryHit message, Peer2 included a list of required attributes, encoded as a SAML attribute request. Peer1 then extracts this SAML attribute request and forwards it to its Attribute Authority.

9. The Attribute Authority must fetch the attributes from its institution's LDAP directory. In reality, there may be multiple backend datasources; here, we have shown only a single LDAP server for simplicity.

10. The AA then signs the attributes and returns them to Peer1.

11. Peer1 then opens a mutually authenticated SSL session with Peer2. This uses Peer1's client certificate and Peer2's server certificate. Peer1 sends an HTTP POST to Peer2 requesting the appropriate file. This POST includes the signed attributes obtained from the Attribute Authority.

12. Peer2 checks the ACL on the file, using OpenXACML and the supplied attributes. If the ACL is satisfied, it will send the file over the SSL session to Peer1. Otherwise, it will end the connection.

Appendix D

Background to the eduSource Communications Layer

In the Canarie POOL Project, Hatala and Richards (2002) demonstrated how a P2P system (SPLASH) could affiliate with centralized servers (PONDS) to form a hybrid network of personal and community collections. In a follow-on project, eduSource Canada, they extended this concept to build connectors and gateways that could link disparate communities into a global network capable of searching and exchanging learning objects (Hatala, Richards, Eap and Willms, 2004).

In a similar method to that of Lionshare, the Splash P2P system also has community nodes ("PONDs") – which is usually Splash software with a more robust database mounted on a web server. The relationship of the community nodes to the single user Splash nodes is not as well defined as in LionShare. Splash uses a proprietary peering protocol that was developed to minimize pinging yet provide bi-directional transport of files across firewalls. In the recent eduSource Canada Project, the Splash team developed the eduSource communications layer (ECL) as a means of linking or "federating" a wide variety of learning object repositories. Federation implies that a transaction initiated in one system can be seamlessly transmitted to and completed in another system. Unlike the monolithic peering protocols, the eduSource Communication Layer in addition to its own protocol provides means for handling other transport mechanisms, protocols or metadata.

In POOL there were 2 basic design parameters:

- To cross firewalls that protected traditional academic bastions from file-swapping, and
- To integrate POND sized repositories by installing a SPLASH repository as its connector.

In eduSource the interoperability was focused on more, rather than the repository. It was inevitable that other repositories would evolve, and that they would likely have significant portions of their meta-data in local jargon, so it became important to make it relatively simple for a repository to join the network. This meant ideally a couple of hours for a good programmer to download and install a connecting middleware. The eduSource Communications Layer provided:

- Definition of a protocol for communication between digital repositories and between repositories and tools,
- Connecting middleware to enable the repositories and tools to join the eduSource network,
- ECL Gateway that provided a framework for building bridges between the eduSource world and other networks sharing digital objects. The ECL is ready to accommodate an ontological layer for future connections searching via ontological mappings.

Together, these tools provide a simple solution for remapping search requests in a highly flexible way. A programmer for a well-formed repository simply needs to download the appropriate middleware and implement the mappings for local metadata and protocols with the eduSource framework. For LionShare sites, the ECL offers an effective method of bi-directionally extending services to other repositories or networks of repositories as it can encompass a wider variety of protocols and document types than are available in Gnutella. While the ECL provides federated searches across a variety of systems, there is more to e-learning than searching. A potential plethora of learning resources and services await similar federation, however, many of these are available only to authorized users. For the ECL to connect P2P users with these services it must be able to guarantee the security of the transactions that it facilitates.

By integrating the ECL connector into the LionShare client Lionshare users will be able to access repositories outside of the LionShare network in a secure fashion using the ECL protocol. To provide authenticated access to those resources, the ECL protocol needs to be extended with a security layer. The Secure ECL connector will implement such security extensions to allow for quick connections of the repositories into the ECL (and LionShare) network. Similarly, when communicating with other networks via ECL gateway mechanism, the gateway has to be able to support mapping between security mechanisms used in each of the connected networks.

Appendix E

Simon Fraser University's Design Ideas Learned from "POOL, POND & SPLASH"

- Every repository should have the opportunity to interoperate with every other repository.

- Any repository should be able to spawn a community by creating a "preferred search list" which may or may not be made available to others. For example the fourth grade math teachers may decide to create a preferred search list for fourth grade math objects. Part of the self-regulation of the list is that it only contains sites with material suitable for grade four students.

- Any community or site should be able to develop an extension of the basic metadata to suit the needs of the community. The local schema should be posted where it can be referenced by other repositories. The middleware should be metadata-agnostic and built with the idea that metadata will migrate. Therefore the application should be reconfigurable.

- Security is the responsibility of every repository. A minimum of 3 levels of file security are required: public access, metadata only, and private. Some repositories may wish to create more sophisticated access permission structures. In LionShare much of the proposed work will be to explore authentication schemas to extend trust fabrics between organizations to individual users.

- The repository should be user-scalable – an institution with good technical support may wish to have a sophisticated database such as Oracle, while a smaller ad-hoc community might settle for the freeware MySQL, and an individual may opt for a smaller java data base engine. The power of the database is a separate issue from the need for common metadata and interface to the search tools.

- It is unreasonable to expect all users to download and install a common software package in order to access the system and its resources. A web interface should be provided to demonstrate the resources available, and the benefits of membership. This runs contrary to the normal requirement of P2P networks.

- Ideally the application should be cross-platform (Mac, Linux and MS).

Appendix F

Economies of Sharing Web Survey
Preliminary Results

Population
The target population was an interested group of, mainly, faculty who tended to be heavy digital learning resource users. This targeting was intentional as it was thought that this group would be better able to comment on what factors would influence their usage of such a system. While the opinions of people who are not heavy digital resource users would undoubtedly provide some interesting insight, it was believed that their lower usage and knowledge of digital image delivery systems would limit the usefulness of their responses. Invitational emails and two follow up reminders were sent to 170 faculty members that were identified during the VIUS study as being particularly interested and heavy digital users. To date we have received 34 response for a nearly 20% response rate. This return rate was lower than hoped for however this survey was began during the summer of 2004 while many faculty are away. An additional email will be sent at the beginning of the Fall semester in the hopes of improving this. Until the survey is closed off to new responses these should only be considered preliminary results.

The responses supported the contention that the sample was comprised mainly of faculty members who are heavy digital learning resource users. For instance, over 90% were faculty and the remaining were staff. Three-quarters searched for new digital learning resources for professional purposes on at least a weekly basis while almost half did so daily.

Usage, Knowledge and Attitudes Towards Peer-to-Peer
While the respondents frequently searched for digital resources they tend not to use P2P applications to find them. Only about a quarter used a P2P application at least weekly for **any** purpose (professional or otherwise) while half never use P2P at all. Half were somewhat familiar with the P2P concept while another third said they were quite familiar with P2P. Only 17% did not know what P2P was. Despite the lower usage of P2P applications, there was no strong bias against using a P2P application for academic purposes. Only about 10% had an unfavorable impression of P2P being used academically. Fortunately, while P2P is not frequently used by intense digital resource users in the academic environment, there does not appear to be massive ignorance about P2P or a bias against using P2P in such a setting.

Important LionShare features.
Perhaps the loudest call for a specific feature was made by 94% of the respondents who said that a thumbnail, sound bite, or other such representation of the file would be important to them. Conversely, this feature was unimportant to almost no users at all.

File management and integration of various collections ranked highly. For about 70% of the respondents, management of their personal collection would be an important feature. About half said it should be able to handle a collection of 1000 objects. Over 85% indicated that being able to work both with their own personal collection as well as external collections is an important capability. About three-quarters indicated that controlling access to their files was a concern.

How the system handles copyright issues would influence the usage of two-thirds of the respondents. At the same time, over 80% of the respondents believe they have an adequate understanding of copyright issues including fair use. It seems that heavy digital resource users will be both knowledgeable about copyright issues and look at LionShare critically in terms of how it handles these issues.

Metadata

There is no consensus as to how much metadata would be useful. Responses were pretty evenly split from the bare minimum to the maximum listed. Further, "I'm not sure" captured the highest percent at 32%. This is a difficult issue to pin down.

Appendix G

CETIS Interview:
LionShare peer-to-peer project kicks off

LionShare peer-to-peer project kicks off
http://www.cetis.ac.uk/content2/20031125012153
Wilbert Kraan, CETIS staff
November 25, 2003

When the Visual User Image Study at Penn State looked at the use of images by students and staff they discovered a good many hidden treasure troves. How, then, to share these and other digital assets without taking control away from the owners? The LionShare peer-to-peer project aims to reconcile the two. We talk to the project's lead, Penn State's Mike Halm.

In both further and higher education, a good many teaching & learning and research material are either home grown or 'home tweaked' out of material shared with others. Most of these things sit on PC hard disks. Nothing wrong with that, but a good many people have tried to make that process a bit easier and more efficient- especially since the growth in the amount of eLearning material.

But the proposed solutions are nearly always a version of getting people's content off their PCs and into silos that belong to 'them'. 'Them' could be your friendly local techy, the librarians, the college administrators, a company- whatever, but not 'us', or, most importantly, 'me'. And that may be a problem, because I know best with whom I want to share my articles, learning objects or other digital assets.

The idea behind LionShare is to give people that control by providing them with a program that runs on their own machine, and allows them to create shared communities. That can be a local department, the whole college or a group of researchers in the same field, wherever they may be.

Such peer to peer (P2P) technology, however, tends to put the fear of Napster (or, more accurately, the lawyers of the RIAA) into administrators. Not just that, in a conventional P2P network such as Napster or its descendants, once a resource has been put out there, there is no way of finding out where it goes, or what someone does with it. For that reason, LionShare integrates authenticated access into the client, so that users are never anonymous on the network.

One other practical consideration for any resource network is the availability of a decent collection of resources right from the start. A library without books is not very useful, after all. To this end, LionShare will incorporate the technology to interoperate with major existing learning object networks and digital libraries, much like the conceptually very similar SPLASH tool from Simon Fraser university interoperates with the rest of Canada's eduSource network. LionShare networks will also be able to talk to SPLASH networks, as its makers are involved in the development of LionShare.

To find out a little more about LionShare and the ideas behind it, we talked to Mike Halm, chief architect of the project.

The outline of the project says that "if they [regular users] could find these resources, it is not clear whether they could use the materials or how they would get permission to do so." Why not solve that problem by using a Digital Rights Expression Language (DREL) and some access control and authorisation function on a dedicated repository?

Mike

> Because we want to focus on the individual, to let them make the decision with whom they want to share, and make it easy, without the overhead of a whole digital rights infrastructure.
>
> What we want to do is get rid of the anonymity [of P2P networks, WK]- the LionShare application started as a kind of personal information manager. We found that a lot of people had trouble keeping track of all their stuff. The trick was to make making metadata as easy as possible; you authenticate, the program harvests lots of metadata about you from the LDAP server, and from the files that are being organised, templates for specific kinds of objects and from re-useable profiles. That means that we'll only need to collect a couple of fields direct from the user.
>
> We want to foster communities- hence the peer server. Access is controlled via the peer server- stuff is persisted there. So if you take a laptop with a LionShare peer home, authorised people will still be able to get at the resources. A department can have a server, and only grant access to that department. There would be a Shibboleth [common network access function, WK] 'lite' authorisation engine on the server.
>
> The idea is that wide spread communities can still be properly integrated- via federated access control.

How will you be going about the problem of searching a number of different collections with one single search?

Mike

> We will make use of eduSource's work on IMS DRI [Digital Repository Interoperability], which enables us to talk the Z39.50 and OAI [Open Archive Initiative] protocols that the big libraries use. They can also get back in to the P2P network- if the LionShare peer allows it.
>
> We're also looking at OKI [Open Knowledge Initiative] interfaces for authorisation and federating searches among multiple Learning Object Networks- that's were our real focus is. The LionShare network will have an OSID [OKI interface, WK] that will help do that.

> Another important thing is to build a development community right from the start. With every release of the tools we'll have a requirements bucket. If its out of the scope of our project, developers can build it with support from the open source community, unlike, for example, OKI.

Which main protocols are you looking at for LionShare itself?

Mike

> The P2P protocol we have used so far is Gnutella, but we've had to alter it so that you have to authenticate first. Dartmouth is also doing a PKI [Public Key Infrastructure] implementation of the Gnutella protocol- that might be worth pursuing. We'll evaluate other ones like JXTA, which has some security already built in.

With that kind of control, it could even be feasible to use LionShare for authoring and controlling an ePortfolio. The UK's learner information profile, for example, envisages that the Personal Development Record, at least, be controlled by the learner herself. Which is fine, but leaves the problem of where to store these things in a predictable way. A peer to peer client with the whole UK Learner Profile on it could be quite attractive. Particularly if it comes complete with a means to check the authenticity of the transcript and check the authorisation of, say, a particular employer to look at a relevant portion of the profile.

For now, though, the project is mostly concerned with sharing learning objects, images and other multimedia material among communities that you define. More information on the project can be found on a LionShare web site (lionshare.its.psu.edu). To find out more about Splash, go to the EduSplash site (www.edusplash.net).