A Public Keys Based Architecture for P2P Identification, Content Authenticity and Reputation

Neander L. Brisola[1], Altair O. Santin[1], Lau C. Lung[2], Heverson B. Ribeiro[1], Marcelo H. Vithoft[1]
[1]Pontifical Catholic University of Paraná
Graduate Program on Computer Science
Curitiba – Paraná – Brazil
[2]Federal University of Santa Catarina
Department of Computer Science
Florianopolis – Santa Catarina – Brazil
{neander, santin, heverson, vithoft}@ppgia.pucpr.br, lau.lung@inf.ufsc.br

Abstract *— In the classic use of P2P, e.g. file sharing, there is no concern about persistent peer identification, peer and content reputation and content authenticity. Security proposals currently found in technical literature try to adapt techniques from client-server architecture to P2P environments, which it is not the most appropriate approach. This work proposes applying public keys to identify peers. It allows creating a persistent identification scheme, without losing anonymity, even in a self-managed environment as P2P. Also, it applies digital signature to provide authenticity to the P2P content and to guarantee non-repudiation in the content transfer. In order to provide credibility to the non-certified content and public keys a reputation mechanism is applied. We have developed a prototype to show the benefits of this approach.*

Keywords: Security for P2P, P2P persistent identification, content authenticity, peer and content reputation.

I. INTRODUCTION

Peer-to-Peer networks (P2P) allow end-to-end communication regardless of the underlying network (in general based on *IP*).

Currently, P2P networks are presenting themselves as an alternative to client-server architecture. In such a case, a peer (nodes of P2P network) can be a *serv*er and a cli*ent* at same time, so called *servent.*

In general in P2P networks there is no concern with authenticity of content, i.e., any user can modify content and republish it with the same original description (index keys). A user searching for certain content would find the authentic and modified content through keywords, but there is no criterion to choose among them. In some cases, the P2P front-ends offer as searching parameters (choice's criteria) bandwidth, availability and other features. If a user chooses the modified content, he/she would not even know that the original content had been modified and published with the same description. In addition to being misled the user would share this polluted content with other peers, becoming a passive replicator of P2P junk [1].

A worst case happens when a peer publishes a content using an attractive description (keywords), but in fact there is no relationship between keywords and file content (content pollution). In 2005 the polluted content already achieved 50% of P2P traffic, in some providers it implied in 60% of the Internet traffic [1].

In order to solve the aforementioned problem of lack of content and publication authenticity a reputation mechanism could be considered [2]. Reputation based on authority relies on a node which signs all documents, assuming that a subsequent succeed signature verification gives authenticity to the content [3]. However, a corrupted content may have an authentic signature.

In P2P, authenticity of content is obtained using digital signature without apply a certificate authority, i.e. certificates are auto-signed. In fact, auto-signed certificates are suitable only to protect a message at communication channel level [4].

The negative point of auto-signed certificates is that there is no well-known entity to provide the endorsement of a certificate, as a PKI certification authority. In this case, peers with positive reputation in previous access can be considered as *credible* [3]. The problem is that peers change their identification often to support anonymity [5]. Therefore, it is necessary to create persistent mechanisms for peer identification, considering that the network is driven by content and it is not client-server architecture.

In any system involving security it is worth providing resources which prevent false denial of participation in a content exchange to support non-repudiation. In the case of networks driven by content (P2P network) non-repudiation must be accomplished without compromising anonymity [5].

As in P2P network the content is distributed over *servent*s, there is not a fixed server for content, thus where the content can be found? Which is the path (URI) to the content?

SDSI/SPKI [6] is a simple PKI, without certification authority. The identification mechanism on SDSI/SPKI is public key-based and the name certificates links the identification to local names. All operations on SDSI/SPKI are digitally signed. One important concern in SDSI/SPKI is the storage and retrieve of name certificates [7].

In this paper we propose a scheme for persistent peer identification, supporting anonymity, assuring non-repudiation, and providing credibility to SDSI/SPKI public keys through a reputation mechanism. SDSI/SPKI keys will be used to generate digital signatures which will assure the authenticity of P2P messages, e.g. publications, request/response, content downloading, etc. The reputation

This work has been partially supported by Brazilian National Research Council (CNPq), grant nº 550962/2007-7.

978-0-7695-3639-2/09 $25.00 © 2009 IEEE
DOI 10.1109/WAINA.2009.139

mechanism also aims at providing content authenticity and helping the selection of content source.

This work is structured as follows: Section II presents peer-to-peer technologies. Section III addresses SPKI/SDSKI. Section IV details the proposal. Section V shows the related work. Section VI presents our Conclusion.

II. PEER-TO-PEER NETWORKS

There are several ways of implementing P2P networks [8]. However, there are two models which are different regarding connection control, brokered and pure. In the first case, a peer which searches for certain content connects itself to a server peer to receive *servent*s' address.

In the pure model, nodes communicate between themselves through a direct connection for resource sharing as well as to obtain the *servent*s' address. There is no centralized node to mediate connections in pure P2P networks. JXTA is P2P infrastructure that employs connection controls in a similar way. Indexing and searching of content are made by flooding messages addressed to special neighbor nodes on the JXTA network to make this activity easier.

A. *Indexing P2P content*

By its distributed nature, P2P depends greatly on indexing services to facilitate the content searching. The simplest implementation technique to address search is flooding all the peers with query message. Other most efficient search technique is applying DHT (Distributed Hash Table).

In DHT all data stored in the table pass through a hash function (e.g. MD5 or SHA-1) before being inserted in the table [9]. After this procedure an ordered pair (key, value), is stored in the DHT. For instance, when indexing content in distributed systems, the key represents an index for the value field and the value stores the attributes that lead to the content.

B. *JXTA*

JXTA [10] is a set of protocols based on XML created in order to provide typical functionalities of P2P networks. It approach is independent of platform or programming language, offering a general-purpose architecture for creation of P2P application.

JXTA creates an abstraction of underlying network, hiding the communication complexity between devices of heterogeneous networks. It also can transpose Firewalls.

As identifier JXTA applies UUID, a 128-bit address data to refer an entity (a peer, an advertisement, a service, etc.). Once a peer gets an UUID, its can communicate with other peers through the JXTA protocols, for instance, to find advertisements, peers, peer group, and so on.

III. SIMPLE DISTRIBUTED SECURITY INFRASTRUCTURE (SDSI) AND SIMPLE PUBLIC KEY INFRASTRUCTURE (SPKI)

SDSI / SPKI is a simple PKI, totally decentralized and independent of technology, allowing the storage of certificates in any type of repository [6]. SDSI/SPKI is client-oriented and does not need server infrastructure for its operation. SDSI/SPKI supports anonymity through the usage of a public key for principal identification. SDSI/SPKI guarantees authenticity based on digital signature. Furthermore, it may be utilized as a part of non-repudiation mechanism, since all the exchanged messages need to be digitally signed.

The SDSI names are always local, corresponding to the names space of the principal issuing the certificate. The issuer of a certificate is always identified by its public key, which concatenated to a local name forms a unique global identifier [7]. In SPKI/SDSI any principal may create a pair of keys (private and public) and then link the public key to a name in its local name space and publish it through a certificate.

One disadvantage of SPKI/SDSI auto-signed certificates is the lack of entity that endorses the issuer of a certificate.

IV. THE PROPOSED ARCHITECTURE

Aiming to provide an easy access to the security features, an intermediate layer between the application layer and the P2P infrastructure is proposed in this work.

The objective of the proposal (Fig. 1) is to assure at an application level, some security properties such as authenticity, integrity, and non-repudiation in the sharing of content on P2P network.

Servent software deployed for use in P2P network is made available in the application layer. Interposing the application layer and the P2P infrastructure it is a security layer based on public keys cryptosystem, which is not transparent for the *servent* applications. It is assumed that *servent*s use the distributed index repository, implemented on top of DHT, to share common information among them, regarding the security, identification, addressing of content, etc.

Initially it is assumed that all peers detain a public key to their own identification in the P2P *servent* layer. Such identification is persistent and independent of the *peer id* that is used in the P2P infrastructure layer for routing.

The *peer id* changes constantly (normally on each initialization of the peer), but in such a case the peer publishes in the index repository the new mapping from the *peer id* to the public key. The publication cannot be forged given it is digitally signed by the public key that made the publication. Thus, a peer holding a *peer id* cannot impersonate a public key by a fake publication. The anonymity of the peer is preserved because a public key does not necessarily identify a principal (*servent*) in the real world.

A peer must also keep safety the private key correspondent to the public key identification to be able to make digital signatures.

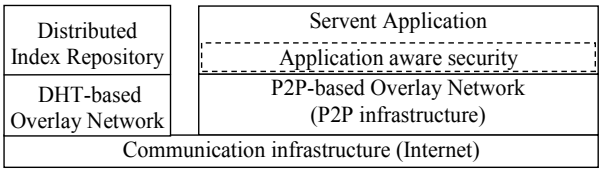

Figure 1. Overview of Proposed Architecture

In the proposal everything must be signed (publication, content, search, reply, certificates, etc.), thus the

non-repudiation mechanism is facilitated, given that exist a strong link between the peer request (signed by its private key) and its identification (the public key).

The P2P infrastructure layer offers resources for storage and transportation of content documents, hiding the infrastructure of communication from the higher layers.

Every time a *servent* intends to share a document on the P2P network it signs the content, that is stored in the local peer repository, and publishes the keywords describing its content in the index repository. Thereby, the peer that queries the index repository can know if the peer that is doing the publishing is a trustworthy *servent*. The authenticity of publication (digital signature verification) can provide a hint of content authenticity. However, an evaluation done based on the reputation of the public key that is publishing, for example, can be more accurate on such rating.

Peers may choose to download content only from other peers who already know the public key (based in their peer reputation). The goal is to reduce the probability of getting false content. Therefore, the credibility of the public key making a publishing is based on its positive reputation with the client peer. The positive reputation of a key is built based on providing authentic content, because a peer may make authentic publishing (with verified digital signature) but with false content. In our proposal we assume that the evaluation of content authenticity can only be done by a human [2].

Positive reputation is obtained from a historical-based relationship between the P2P server and P2P client, as aforementioned. However, to facilitate the reputation building, a client peer may associate a good degree of credibility for a public key without having a history for positive reputation of it. In such a case the server public key must be recommended by a peer that already has a good reputation with that client. Therefore, it is assumed the P2P server is trustworthy, according transitivity property. When it is not the case the following reputation scheme should be adopted. The content authored by a *servent*, namely *document identification*, is defined by the format: *PublicKeyAuthor@documentName.*

A peer keeping a copy of content published by a peer author will announce its identification through a *P2P Uniform Resource Identification (PURI),* following the format: *@PublicKeyAuthor@documentName*.

In classic P2P network, normally, the content address (URI) is linked to a *peer id*, and therefore it changes each time the peer startups. The PURI maintains permanent the document identification, while it server identification may change. In PURI, the server (*PublicKeyServer*) is a secondary document identification qualifier.

It is easy for a peer client to differentiate the peer authoring a content from a server replicating it, i.e., the *PublicKeyServer* is same of *PublicKeyAuthor* in the PURI, when a peer is authoring a content.

The proposed reputation scheme is based on qualification of both, author and server of a content. When a peer requests content (document) for downloading from a P2P server, it sends back to the client a qualification request. That request must be signed by the client and returned to the server in order to obtain the document (content). After, the server publishes the signed qualification request on the index repository and provides the document to the client.

Supposing a client peer wishes to know who is the holder of a public key performing a content publication. The client may search for the name certificate associated to the peer in the index repository. If the certificate is found, the client peer can identify the author of a publication. Otherwise, the publication is anonymous, which does not mean that the anonymous publication cannot be authentic. A public key can produce authentic content and make authentic publications; however, their author prefers not to be identified.

After downloading the document the peer client evaluates the content and assigns a grade for its author and server. The grades attributed to both are recorded on the index repository. The grade can be neutral, positive or negative, ranging from neutral to highest positive/negative value. It is assumed that a positive qualification of an author means the content produced by it is recognized by the peer client as developed with good quality in all sense. Analogously, a positive qualification of the P2P server is understood as an approval to server's good service in providing contents.

On the index repository a qualification request is answered by its respective qualification (voting) expressing the grade assigned by the client to each one, author and server of content. Each vote should answer (counteract) its respective qualification's request, i.e., after the voting action, a qualification *pendency* (request) will not be valid anymore.

When a peer client attributes a positive grade to the author of a content, it must share such content with others publishing it on the index repository. Thus, the client itself becomes a replicating P2P server; otherwise the replication is not recommended in order to avoid junk content distribution.

The sharing of all publications through the index repository can also be employed to keep a chronological authenticity of publication, preventing already published content from being illegally republished as new by a malicious author peer.

Prevention against denied of service or other attacks on the network layer such as exploits and other types of *malware* (*malicious software*) is not the goals of this work.

A. Scenario

Considering a news agency where all articles are made available online using the Internet. Such agency desires to avoid the costs of high availability systems, the unique point of failures using a central server and dependency of a *web designer*. Therefore, the agency chooses to use the P2P network. P2P allows quick availability of news in this competitive area, since being the first journalist to publish important news imply in a well succeeded career. Thereby, the journalists make the news available on their own computers.

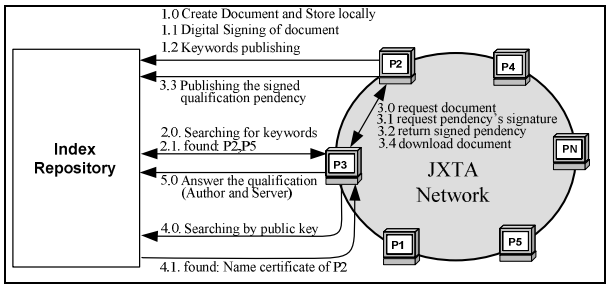

Figure 2. News Agency Scenario

A journalist can be in the most remote area when producing news, however, when he/she makes the news available, allows immediate reading without sending it to a news center to be edited and designed, and afterwards published on the news agency web page.

Initially, it should be considered in the context above that all journalists wishing to publish their news should obligatorily have a pair of keys and preferably a SPKI/SDSI name certificate published in the index repository – the certificate facilitates the identification of the journalist.

From the implemented browser in the prototype [11] it is possible to edit HTML pages (the editor is simple, however it allows the evaluation of the prototype). After editing the news in the HTML editor, a plug-in is triggered to digitally sign the article. After digitally signing the news the journalist store it into a web server shared directory. Also, he/she creates and signs the keywords and document identification, and publishes them on the index repository (Fig. 2, events 1.1, 1.2 and 1.3).

At a given time peer P3, for example, searched for some keywords and found various publications (Fig. 2, event 2.1 and 2.1), among them it is the one previously published by P2. As P3 needs to choose among various publications returned (from index search), let us assume that P3 has already obtained content from P2 previously; therefore P2 is on the P3 reputation historical.

P3 decides to request the news from the web server running in P2 through the JXTA (event 3.0). Then, P2 sends a qualification *pendency* to P3, which signs it and sends it back to P2 (events 3.1 and 3.2). Thus, P2 publishes the P3 signed qualification *pendency* on the index repository (event 3.3).

P2 web server records the public key of the peer P3 to avoid non-repudiation. The anonymity of P3 is not being violated because if P3 has not published a name certificate, there will not be a name linking the peer to its identification (public key). P2 releases the request content document to P3 downloading it (Fig. 2, event 3.4).

After downloading the news via http, P3 records its source and verifies the digital signature of the document. The recording of document source is applied to update the reputation historical and for non-repudiation purposes. It is important that P3 logs the public key of document source because whether the document is replicated, the peer server of copied document will be lost. In other words, P3 will publish oneself as *PublicKeyServer* (in the PURI) for the downloaded content, in substitution to the last server – P2.

After reading and evaluating the downloaded news, P3 may wish to know to whom the public key signing the document belongs to. In such a case, P3 must search on the index repository again in order to retrieve the name certificate correspondent to the P2 public key (Fig. 3, event 4.0). In this scenario as P2 (journalist) published a name certificate, P3 can know who is authoring the news downloaded.

B. Implementation Issues

The prototype architecture (Fig. 3) is composed by various technologies which jointly implement the proposal. P2P infrastructure of JXTA was used to achieve a platform and network environment independence, as well as to provide a transportation mean for P2P content. The security features are based on SPKI/SDSI, aiming security in a *serverless* based Public Key Infrastructure. The Bamboo implementation of DHT [09] has been used to deploy index repository and search engine for name certificates, *servent* identification, content publication and addressing, and author and content qualifications.

The *servent* application, *web2peer* [11], implement an Internet/P2P browser (P2P client), a *HTTP serve*r (P2P server) written in Java, and an embedded *DHT client*.

The Apache web server *Jetty* was used as the P2P server in the application layer; all the documents/contents stored locally in *DocRoot* are available for download through the *http server*. When the peer acts as P2P client, the P2P *http client* connects to an http server in another *servent*. *P2P initiator* requests the *insertion task* to announce the JXTA *peer id* for itself on the index repository. The *p2psockets* adapts the classic Java socket implementation to be use under JXTA infrastructure.

DHT client aims to make publications of content authored by a *servent* and make searches on DHT (index repository). The *keyword parser* extracts from the html content (page) the keywords to be published on the index repository. All the messages exchanged between *servent* and DHT is done through *XMLRPC*.

The client gets access to *servent* application (web/P2P browser) through the GUI. Build in Java, the GUI is instantiated by *screen loader* and the user actions is treated by *event listener*. The GUI enables the user to get access to http page from Internet in the traditional way (through the *Internet standard browser features*), gets content from P2P network (through *P2P content searching*) or creates and publishes content through the *html editor*.

The SDSI/SPKI offers facilities for digital signature, key generation, hash functions, and name certificates handling based on SDSI (Criptx32) libraries. The browser accesses SDSI/SPKI security facilities through *plugins* to check/make digital *signatures* and compute cryptography *hash* functions, and to obtain a digest or verify integrity of a content.

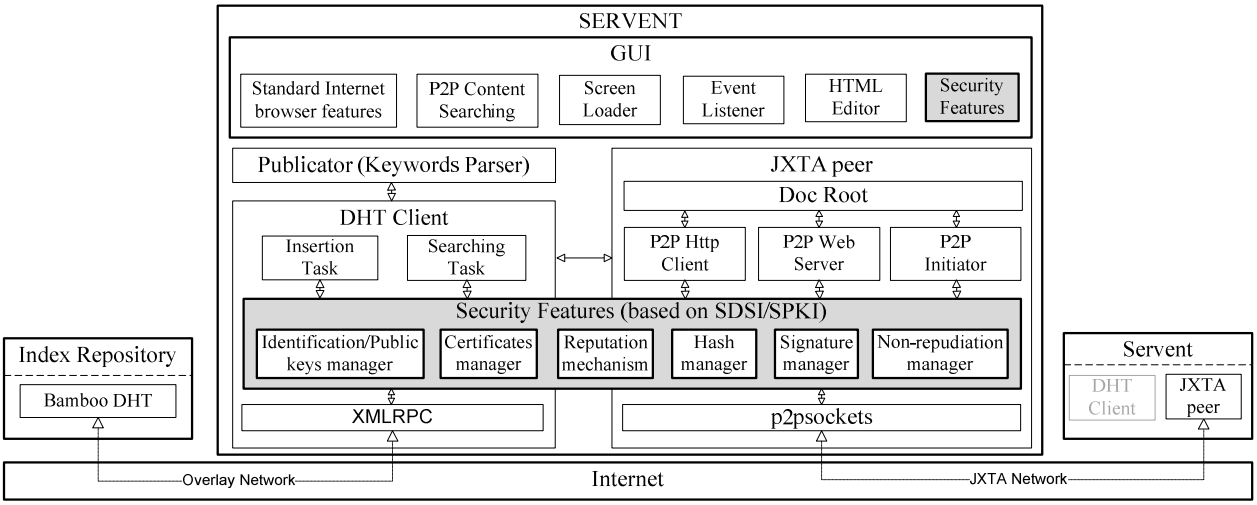

Figure 3. Prototype Architecture

The proposal relies on public key to identify a *servent* and therefore obtain a persistent peer identification that has been employed on the reputation mechanism. Thus, the correct creation, store and handling of public key are very important. *Java keystore* was used to protect the private key. All issues regarding keys are treated by *keys manager* in the prototype. Each time the P2P initiator invokes the *insertion task* automatically it associates the public key of the *servent* to *peer id* and publishes it on the index repository.

The *certificates manager* can publish/revoke a name certificate for the *servent* and store known certificates of other *servent*s. Also, in this module the credibility of auto-signed certificates is checked against the reputation of a given public key to be sure it is trustworthy.

The *non-repudiation manager* takes care of logging public keys downloading content and requires digital signature verification to guarantee that a P2P content server is authentic.

The qualification records stored on the index repository serves to provide information for the reputation mechanism. This prototype module works asking for qualifications when a content is requested to the server side of *servent* or querying the index repository through the *searching task* to obtain the reputation of a P2P server.

The reputation mechanism builds local scores about peers that it knows. When there is no local knowledge about a P2P server the reputation mechanism requests all the records regarding *pendency* (positive, negative and neutral qualifications) from index repository for computing the local scores. The period for updating scores depends of each peer. The scores are applied also by *certificates manage* to obtain the credibility of an auto-signed certificate that is presented by P2P content servers.

The prototype was implemented using the aforementioned technologies; however the proposal is not limited to it, and can deploy any feasible scenario.

C. Proposal Evaluation

The evaluation was done on the implementing the prototype (Fig. 3). The measurement done on our essays considered the overhead for the prototype performance, caused by the usage of digital signatures and by the adoption of the scheme to request qualifications. The trigger of signing plug-in in the HTML editor introduces one overhead of 12% on the time to publishing keywords in the index repository and for storing the content on *DocRoot directory*, when compared to the same operation without the usage of digital signature. The keywords signing verifications took 7% more time than the operation without applying signature.

The content authenticity verification spent about 19% more time than the same operation without signature, for files bigger than 500 KB. The time increment for files below 500 KB reduces proportionally; for files above 500 KB the increment is not significantly proportional. The sending of qualification and its return back to content server, followed by their publication on the index repository, in a LAN, not taking into account the human intervention, increased about 23% the time in a downloading operation. Measurements were taken all the time in average after 40 repetitions, the coefficient of variation observed were under 5% .

V. Related Works

In the technical literature, many works are focused on techniques to assure security proprieties to distributed content in the P2P networks.

Authenticity from digital signatures is obtained from name certificates, issued by a CA (Certification Authority), or auto-signed certificates in the Poblano project for JXTA

platform [12]. Poblano can also apply certificate based on a trustable network therefore providing recommendation chain, similar to a Web of Trust in PGP (Pretty Good Privacy).

Identity Crisis assumes that identities never changes. That model uses self-signed certificates, which permit honest peers build trust relationships during a series of disconnections and reconnections from distinct IP addresses [5].

EigenTrust project [13] uses a transitive trust for peers and content reputation – a peer taking into account the opinions of peers it trusts. EigenTrust computes a local trust score for all peers that have provided authentic or fake content to it. The scores are produced based on the peer satisfaction about the downloaded content.

Xrep project is a Gnutella protocol extension [14], which considers peer and content reputation in a fully distributed way. A peer requesting content from overlay receives from all peers the associate keywords matching the file digest. A new query asking for reputation of the previous return peers and their files are made to others peers to select the proper peer for downloading the content. After downloading the file from the selected peer, its integrity is checked against the file signed digest. Afterward, the client peer will update its local repositories with its opinion about the downloaded content and its server peer.

In a nutshell, the proposals found in technical literature deals with the identification of peer and authenticity of content using public keys obtained by auto-signed certificates. But, none of them consider the credibility (endorsement) of auto-signed certificates. Also, it is not found a proposal that offers an easy way to find the mapping from peer public key to underling network addressing in a decentralized P2P network. The persistent content identification is not addressed.

Our proposal addresses all the literature limitations mentioned above, providing a persistent identification and non-repudiation, while assures anonymity. Additionally, in our proposal we present a scheme to give creditability to auto-signed certificates and apply a PKI (SDSI/SKPI) that is more suitable to P2P features, mainly, due to its ability to manage name certificates without requiring a TTP.

VI. CONCLUSION

The goal of this proposal was present a scheme that supports publishing and checking publication's authenticity on index repository. Moreover, it has been proposed a reputation mechanism that provides credibility to the public keys that sign the publications.

The public key was applied also to persistently identify peers, content and servers content, and for logging non-repudiation records. However, the usage of public key as identifier in the proposal was done respecting the free choice of each peer staying anonymous with no losses for the proposal goals. Besides, reputation scheme allowed to easily highlighting distinct content author and server for client peer.

The prototype showed that the scenario with the P2P based news agency is advantageous in comparison to the conventional one. The main advantages are immediate content availability without the need of intermediation of web designers, and mostly by authentic content availability even outside of the agency site. Moreover, prototype allowed us to evaluate that PURI replaces efficiency the URI.

REFERENCES

[1] R. Kumar, D. Yao, A. Bagchi, K. Ross, D. Rubenstein, "Fluid Modeling of Pollution Proliferation in P2P Networks," Proc. ACM Sigmetrics, 2006, pp. 335-346.

[2] N. Daswani, H. Garcia-Molina, B. Yang, "Open problems in data-sharing peer-to-peer systems," Proc. 9th ICDT, LNCS vol. 2572, Springer, 2003, pp. 1-15.

[3] P. Resnick, R. Zeckhauser, E. Friedman, K. Kuwabara, "Reputation systems," ACM CACM, 2000, pg. 45-48.

[4] X. Zhang, S. Chen, R. Sandhu, "Enhancing Data Authenticity and Integrity in P2P Systems," IEEE Internet Computing, vol. 9, nov. 2005, pp. 42-49.

[5] S. Marti, H. Garcia-Molina, "Identity crisis: Anonymity vs. reputation in p2p systems," Proc. IEEE P2P, IEEE Computer Society, 2003, pp. 134- 141.

[6] C. Ellison, B. Frantz, B. Lampson, R. Rivest, B. Thomas, and T. Ylonen, "SPKI Certificate Theory," IETF RFC 2693, Sep. 1999.

[7] A. O. Santin, J. S. Fraga, C. Maziero, "Extending the SDSI / SPKI model through federation webs," Proc. 7th IFIP TC-6 TC-11 CMS, LNCS 2828, 2003, pp. 132-145.

[8] S. Androutsellis-Theotokis, D. Spinellis, "A survey of peer-to-peer content distribution technologies ," *ACM Comput. Surv.*, Vol. 36, Dec. 2004, pp. 335-371.

[9] S. Rhea, B. Godfrey, B. Karp, J. Kubiatowicz, S. Ratnasamy, S. Shenker, I. Stoica, and H. Yu, "OpenDHT: a public DHT service and its uses," Proc. Conf. on Appl., Tech., Arch., and Prot. for Computer Communications, 2005, pp. 73-84, doi: 10.1145/1080091.1080102.

[10] B. Traversat, A. Arora, M. Abdelaziz, M. Duigou, C. Haywood, J. Hugly, E. Pouyoul, B. Yeager, "JXTA 2.0 Super-Peer Virtual Network," 2009, Available in: http://www.jxta.org/project/www/docs/JXTA2.0protocols1.pdf.

[11] H. Ribeiro, L. C. Lung, A. O. Santin, N. Brisola, "Implementing a Peer-to-Peer Web Browser for Publishing and Searching Web Pages on Internet," Proc. IEEE 21th AINA, IEEE CS, 2007, p. 754-761.

[12] R. Chen, W. Yeager, Sun Microsystems, "Poblano: A distributed trust model for peer-to-peer networks," 2001, Available: http://gnunet.org/papers/jxtatrust.pdf.

[13] E. Friedman, P. Resnick, "The Social Cost of Cheap Pseudonyms," Journal of Economics and Management Strategy, vol. 10, 2001, pp. 173-199.

[14] K. Walsh, E. Sirer, "Experience with an object reputation system for peer-to-peer filesharing," Proc. 3th NSDI, May 2006, Available: http://www.truststc.org/pubs/173.html.