Trust Management in Wireless Networks

Eduardo Cardoce and Muthu Pitchaimani
Department of Electric Engineering and Computer Science
University of Kansas
1520 West 15th Street
Lawrence, KS, 66045-7621
{ecardoce, muthu}@ku.edu

ABSTRACT

The open and anonymous nature of wireless networks makes it an ideal medium for attackers to spread malicious content. This paper addresses our proposed approach to provide security in wireless networks based on a static access control for a peer to peer overlay network using a trust management system and applying digital signatures. Thus we implemented the KeyNote trust management system on top of the Chord overlay network. The performance of our algorithm in terms of look up and delay is demonstrated by experimental results

Key Words

Peer-to-Peer (p2p), Mobile Ad-Hoc Networks (MANET), trust management system (TMS) and digital signatures.

1. INTRODUCTION

Security in Mobile Ad-Hoc Networks (MANET) has been an area of research in the past several years. A MANET is an autonomous collection of mobile users that communicate over relatively bandwidth constrained wireless links [1]. However, due to the absence of a centralized server, the implementation of a security system becomes a complicated task. Furthermore, due to the fact that the majority of the nodes are mobile, the network topology may change rapidly and unpredictably over time which makes the process even harder.

Similar to MANETs, a peer-to-peer (p2p) overlay network consists of a dynamically changing set of nodes connected via the Internet [2]. P2p overlays and MANETs share several characteristics of self-organization and decentralization like frequent changing topology and hop by hop connection establishment. These common characteristics shared by p2p overlays and MANETs also dictate that both networks are faced with the same challenge, that is, to provide security in a decentralized and changing environment.

Several approaches have been proposed to regulate some malicious attacks on overlay networks, like Sybil and Eclipse attacks. For example, in [3], the authors identify the Eclipse attack as one of the security problems in structured overlay networks. They introduce a defense mechanism that prevents these types of attacks by bounding the degree of overlay nodes. Similarly, in [4], Castro et al use strong structural constraints on the overlay to defend against the same attack.

The use of different techniques to defend against malicious attacks is not the best approach to implement security in peer-to-peer networks because the variety could be extensive and the network will still be vulnerable towards unconsidered or innovative attacks. For this reason, researches have proposed different methods to provide security. In [5], an identification of dishonest peers by means of a complaint based system was introduced. This is certainly a much suitable security application to work with peer-to-peer networks because nodes are classified as trustworthy or untrustworthy based on negative feedback provided by other peers.

In addition to negative feedback, the introduction of positive feedback to distinguish malicious responses from benign responses was proposed in [6]. This concept of reputation presents the advantage that peers are not only punished for their bad behavior but they are also rewarded for their good deeds. Currently, there are several groups, including MIT, Berkeley, University of Maryland, among others. That are studying the use of trust management systems to provide security in p2p networks, however, there have been some difficulties with initial trust assignment and therefore overload considerations have delayed their research.

December 12, 2006, Lawrence, Kansas, USA.

1.1 Our focus

The goal of this paper is to use a trust management system based on digital signatures to implement security in overlay networks. Thus we propose the use of the KeyNote [10] trust management system in junction with a modification of the Chord [9] overlay network to simulate a real environment of secure overlay network. More specifically, our contributions on the subject are the following:

- We propose a modification of the Chord daemon so that the overlay network could be able to work with the implementation of digital signature based trust system using KeyNote.
- We make use of a trust server which provides access control to joining nodes. This server will be modified to a trust and evidence server in the near future to assign a trust value to each node in the network
- We implemented our system in 8 real nodes using the 2060 Lab located in Eaton Hall, at the University of Kansas. In addition to the results of the proposed algorithm we present an analysis of the delay and lookup times introduced by the usage of KeyNote.

The rest of the paper is organized as follows: section 2 presents some related work. The proposed approach, including a brief description of the KeyNote TMS and the Chord overlay network are described in section 3. The algorithm's results are discussed in section 4. Finally future work and conclusions are presented in sections 5 and 6 respectively.

2. RELATED WORK

The topic of security in wireless and peer-to-peer networks has been an area of intense research in the past years. There have been several different proposed techniques to deal with malicious attacks and malicious nodes. However, the absence of a centralized server and the changing topology due to frequent node joining and leaving has challenge researches to exploit their knowledge on the subject and come up with innovative solutions to the problem. Some of the proposed approaches are discussed in this section.

2.1. Secure routing on peer-to-peer overlay networks.

The first proposed approaches to include some sort of security in peer-to-peer networks consisted in analyzing different attacks to the network and proposing a defense mechanism against them. Examples of such methods are presented in [3,4] and [8] which include the Eclipse, and the Sybil attack.

Castro et all in their paper, *"Secure routing for structured peer-to-peer overlay networks"*, argue that the Eclipse attack is more general than the Sybil attack and that therefore, this represents a severe threat that the system should defend against. Furthermore, they mention how an attacker can use a Sybil attack to launch an Eclipse attack by creating a large number of seemingly distinct overlay nodes to populate the neighbor sets of correct nodes.

Because of the severity of this attack, the authors have proposed a defense mechanism for overlay network protection. The idea behind their algorithm is that the indegree of attacker nodes must be higher than the average indegree of nodes in the overlay during an Eclipse attack. Therefore, benign nodes can bind the indegree of malicious nodes by choosing their neighbors from the subset of overlay nodes that are below a defined threshold.

These defense mechanisms to malicious attacks can effectively detect and prevent attacks, however, due to the extensive number of different malicious intentions, programming an algorithm that can provide security to every existing attack is virtually impossible. For these and other reasons, researches have explored other techniques to deal with malicious behavior, like the use of feedback (reputation) provided by other peers regarding previous behaviors. Some of the proposed approaches are briefly discussed below.

2.2. Reputation based trust management

A. Negative feedback

The first cases of reputation based trust management were based on the classification of peers as trustworthy or untrustworthy according to the feedback left by others. An example of such algorithm is presented in [5]. The goal of the approach is to identify dishonest peers using a complaint-based system. The algorithm analyzes earlier transactions of agents and deriving from that the reputation of an agent. The reputation is essentially an assessment of the probability that an agent will cheat. The data necessary for performing the analysis is provided by a decentralized storage method.

One of the disadvantages presented in the papers mentioned above is that it uses only negative feedback to promote the reputation of a peer. This could become an adversity to the system because there is no incentive to other peers to perform the actions they are supposed to do. Therefore, some scientists have proposed the use of positive feedback as well as negative feedback to reinforce and encourage benign behavior.

B. Positive and negative feedback

In EigenRep [7] each peer rates another peer from whom it tries to download files by rating each download as either posi-

tive or negative. Each peer maintains a sum of all his transactions with other peers in a local trust value vector. In order to form a global vector, the local trust values are aggregated around the network and normalized so that malicious peers will not be able to assign arbitrarily high trust values to other malicious peers. Normalizing a peer's global trust value in this way ensures that all values will lie between 0 and 1.

Global reputation of each peer is given by local trust values assigned to by other peers. These normalized local trust values are aggregated in a distributed environment by asking for opinions about other peers and placing them in a trust vector. In this way, the peer having the highest trust value will be selected for download.

A similar approach was proposed in [6]. In their paper, the authors designed a protocol for Peer-2-Peer systems in which a peer looking to download a file uses reputation from itself and others to help determine which host to download from. Meanwhile, the authors define four types of malicious resources such as Naive, Hypocritical, Collaborative, and Pseudo Spoofing. The advantage of this method is the use of positive and negative feedback to determine the reputation of a certain peer which rewards nodes for their benign behavior.

Even though this type of reputation based trust management systems has the advantage of assigning positive feedback to peers, there are currently several issues with initial trust assignment and therefore there should be considerations to avoid overload problems.

2.3. Policy based trust management

Policy based management is a different kind of approach to implement security on overlay networks. Recently developed policy specification languages aim to provide a common framework for specifying security and distributed systems management policies. Some common approaches include PolicyMaker [11], Keynote [10], and REFEREE among others. In this type of systems, trust is managed via a public key infrastructure (PKI) of some sort.

The focus of these approaches is on trust management mechanisms employing different policy languages and engines for specifying and reasoning on rules for trust establishment. The goal is to determine whether or not an unknown user can be trusted, based on a set of credentials and a set of policies. In addition, it is possible to formalize trust and risk within rule-based policy languages, where different peers have different access to the network data according to their trust value.

The greatest advantage of this type of algorithms is that they don't have to deal with initial trust assignment. In this paper we present an approach that uses the KeyNote trust management system to provide security to overlay networks via digital signatures. Furthermore, in our future work we propose an extension of our security system to work with evidence and reputation of peers.

3. PROPOSED APPROACH

In this section we discuss our proposed approach to implement trust management in overlay networks using digital signatures. Our method uses a modification of the Chord overlay network using a distributed hash (for storage) in junction with the KeyNote trust management system to provide access control to joining nodes.

A.. Chord

The *Chord protocol* [9] supports just one operation: given a key, it maps the key onto a node. Depending on the application using Chord, that node might be responsible for storing a value associated with the key. Chord uses consistent hashing

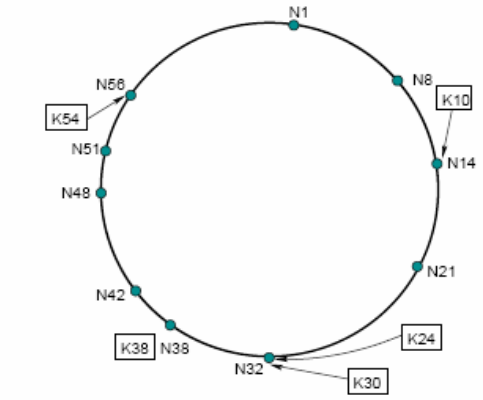

Fig.1. An identifier circle (ring) consisting of 10 nodes storing five keys.

to assign keys to Chord nodes. Consistent hashing tends to balance load, since each node receives roughly the same number of keys, and requires relatively little movement of keys when nodes join and leave the system.

Chord maps nodes into an m-bit circular space In particular, each key is assigned to the first node whose identifier is equal to or follows the identifier of the key in the space. This node is called the successor node. To implement the successor function, all nodes maintain an m-entry routing table called finger table. This table stores information about other nodes in the system. Each entry contains a node identifier and its network address, which consists of an IP address and a port number.

Figure 1 shows an example of a Chord ring with a value of m=6. The Chord ring is composed of 10 nodes and stores five keys. For this specific example, the successor of identifier 10 is node 14, this means that the 10^{th} key would be located at node 14. Similarly, keys 24 and 30 would be located at node 32, key 38 at node 38, and key 54 at node 56.

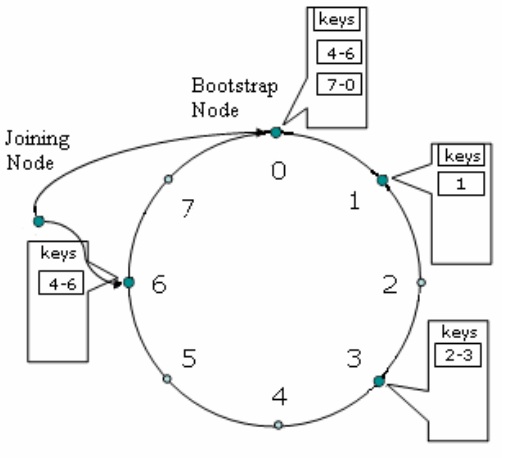

Fig.2. Example of joining node contacting bootstrap node. BN assigns location 6 and gives the node the sixth key as well as the keys of the empty positions before it (4 and 5).

When a new node wants to join the network it must initialize its finger table, which means that existing nodes must also update their tables to reflect the existence of the new node. If the system is in a stable state, a new node can initialize its finger table by querying an existing node for the respective successors of the lower endpoints of the intervals in the node's table. Furthermore, to obtain its location inside the Chord ring, the new node contacts Chord's bootstrap node, which provides the information of the location and keys that will be assigned to the new node, see fig. 2.

The problem with the current Chord protocol is that it grants access to any node that wants to join the network, which might conclude in the addition of a malicious node to the network. This security downside could be controlled with the requirement of a predefined credential to join the network. Next, we present a brief description of the KeyNote trust management system which will be used as a security authority to verify required credentials for a node to join the network.

B. KeyNote

The KeyNote trust management system was introduced in [10] and aims to provide a common, application-independent mechanism, used with specific credentials and policies. When using KeyNote, each application will develop its own set of attributes, with several credentials and policies created to operate on them.

The KeyNote language is designed to make it easy to express and evaluate the different types of policies. The basic element of KeyNote programming is the assertion. Assertions are the mechanism by which a key (or collection of keys) is authorized to perform various trusted actions. Assertions are used to specify local policy. Assertions are designed to be easy for humans and computers to write and understand therefore they are ASCII-based. The function of an assertion is to allow an entity to authorize another entity to perform specific trusted actions. Assertions are designed to require only minimal computation to evaluate them; a more in depth analysis will be shown later.

```
/* At each point the application decides that someone's requesting an
   action, do the following: */
requester          = requesting principal's identifier;
action_description = data structure describing action;
policy             = data structure describing local policy,
                     typically read from a local file;
credentials        = data structure with any relevant credentials
                     sent along with the request by the principal;
PCV                = Call_KeyNote (requester, action_description,
                                   policy, credentials);
if (PCV = = "allowed")
    do the requested action
else
    tell principal that action isn't allowed
```

Fig.3. Pseudo code for using Keynote. It is composed of a requester, action description, policy, and credentials. The Policy Compliance Value (PCV) is returned by KeyNote system to the application

The Policy Compliance Value (PCV) is a result returned by the KeyNote system to the application. It describes whether an action requested conforms to an application's local policy. In figure 3 we present a template code to use the KeyNote trust management system in an application.

C. Trust Server

Since it's the application's job to notice when a node is requesting an action, we propose the use of a trust server to call the KeyNote system with a correct description of the action, the governing policy, and the relevant credentials. By doing this, we can restrict the access to join the Chord network to only those nodes that have the correct credentials. This will provide a defense mechanism against malicious nodes that are trying to access the network.

We modified the Chord daemon so that when a joining node contacts the bootstrap node to obtain a location in the Chord ring, the bootstrap node sends the credentials of the joining node to the trust server instead of directly assigning a position for the node in the circle. The trust server will then call the KeyNote system with the governing policy and the credentials for the joining node. The KeyNote will then return the PCV which will advice the bootstrap node either to grant the access requested by the node or deny it. Figure 4 illustrates the procedure followed by our proposed approach upon a request from a joining node to access the ring. In the first scenario, the node does not have enough credentials to join the network therefore the access is denied by KeyNote and the request is

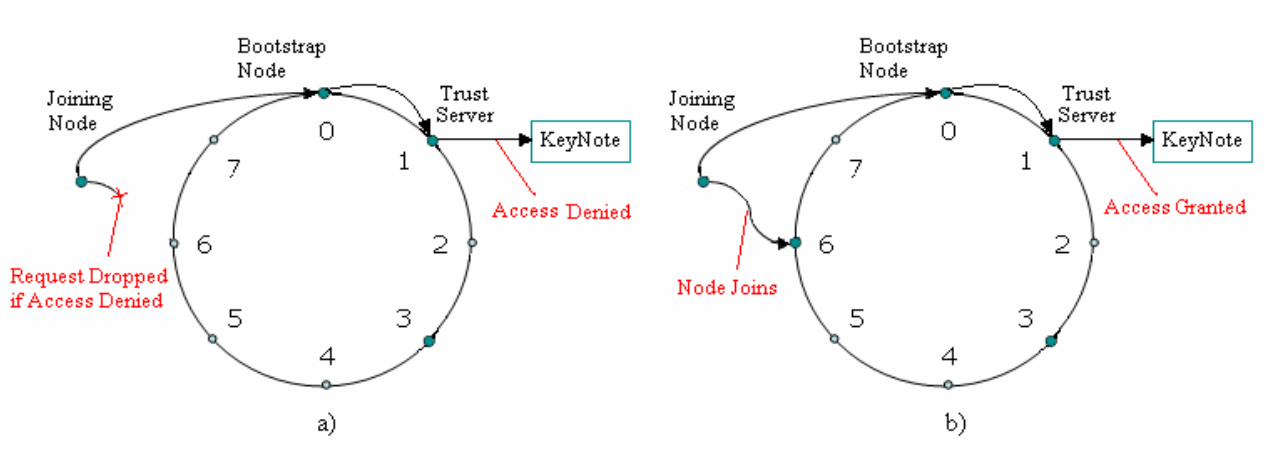

Figure 4: Joining node does not have credentials to join the Chord ring (a) and node has credentials to join the ring (b)

dropped by the Chord daemon. In the second case, the credentials are sufficient so the access is granted and the node successfully joins the ring.

Due to the fact that Chord does not store data and in order to overcome problems with traffic congestion at the node acting as a trust server, the following step in our algorithm is to implement a Distributed Hash (DHash). The DHash will provide storage for the overlay network which will enable every node in the ring to act as a trust server and eliminate the problem of congestion and possible overload.

```
Keynote-Version: 2
Authorizer: "POLICY"
Licensees:
"dsa-hex:3081de02402aa13340295f98111da43a71870847\
 9ce2c02b2583d258e7cbab420c88c54b98c22881937b6bbcb\
 45adc0ba6d146971bec6c45c2ff1c613db8b1054ec434f980\
 0241009e561aaf2db9e6dd22b607e6730cf2951fb81c9d9b9\
 a27ca6b0b7a69e62cdf14ce49b31ab57ef6ccf1cd2f2186fc\
 75a440bd6dad7b92e8030137c0b7af90c3db021500875d529\
 dbe5db7b02a93f85113a0df2f4a77162102406d980a196afa\
 5e290e9b3aa216a62aa7e344efcb0875b361fccd903b03714\
 80a04fd59e687e4f10c2076939211e822d48d6ac9b10e5ad3\
 4d8f57eda4621eb897"
Conditions: (  IP == "192.168.1.14") -> "Allow";
policy (END)
```

Fig.5. Policy for Chord access, if IP address is 192.168.1.14 access is granted, otherwise access is denied.

4. SIMULATION RESULTS

A. Evaluation

In order to analyze the correct functioning of our proposed approach we have implemented the algorithm in 8 real nodes using the 2060 Lab located in Eaton Hall at the University of Kansas. We first implemented the Chord overlay in each computer and the added a distributed hash to provide storage to the network. Later we modified the Chord daemon and implemented the Keynote trust management system

For simplicity, we have programmed our policy to be a predefined IP address (see Fig 5), so when a node intends to join the network, the KeyNote trust management system will verify that its IP address matches the one specified on the policy. We expect our approach to successfully determine if a certain node is authorized to join the ring or not. Furthermore, if a node's access is denied we expect the Chord daemon to drop the attempt of joining. We also discuss the delay and lookup introduced by the use of KeyNote and we expect these timings to be minimal compared with the regular performance of Chord.

B. Results

We tested our algorithm using the specified conditions and under different situations including different number of nodes composing the Chord ring. We obtained 100% accuracy of the KeyNote trust management system since every node with the correct credential was successfully added to the network and access to other nodes was denied. For example, using the previous policy, when a node with IP address 192.168.1.13 tried joining the Chord ring, the access was denied by the KeyNote trusts management system. However, when the requesting node's IP address was 192.168.1.14, the access was granted and the bootstrap node was able to successfully allocate its position in the ring and assign the corresponding keys.

C. Analysis

In order to further analyze the correct functioning of our algorithm we have measured the delay and lookup of the Chord overlay network with and without the implementation of KeyNote trust management. To ensure that we obtained the correct data, we tested our approach using different number of nodes (real nodes) and we obtained the delay and lookup for several tests, see Fig 6. We found the average delay of the Chord overlay without using KeyNote to be close to 480ms and using

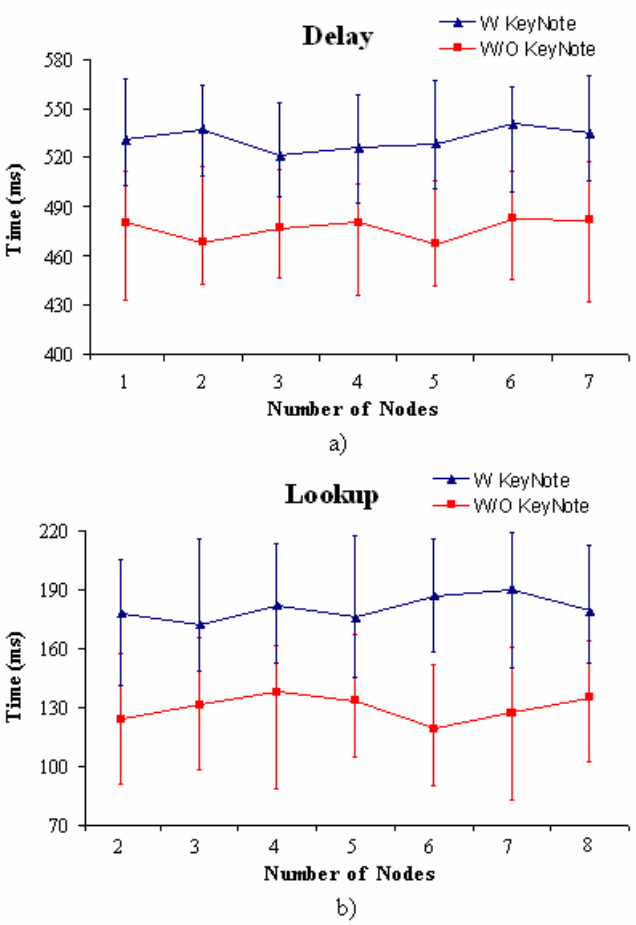

Fig.6 Delay (a) and Lookup (b) for the Chord overlay with and without using KeyNote trust management system.

KeyNote close to 530ms. This means that the introduction of security using digital signatures has a cost of approximately only 50ms.

Similarly, the average lookup timing found for the Chord overlay was 130ms using 2 to 8 nodes, and after the implementation of KeyNote it increased to near 180ms. Once again we obtained a 50ms increase. We consider the algorithm to be robust both lookup and delay wise since the time augment was minimal. Furthermore, based on the KeyNote description, the implementation of additional assertions to have a more structural network control (rather than simply allow or deny) will only affect the performance of the algorithm in the order of micro seconds.

5. CONCLUDING REMARKS

We have proposed a new approach for the use of policy based trust management in overlay networks. More specifically we have implemented the KeyNote trust management system to the Chord overlay network using a trust server and distributed hash to provide security against malicious nodes trying to join the network. We have demonstrated the correct function of our algorithm by testing our approach on 8 real nodes located in the 2060 Lab in Eaton Hall at KU.

For testing purposes, we have programmed our policy to be a specific IP address and we achieved 100% decision accuracy from the KeyNote system against nodes trying to access the Chord ring, based on their IP address. Furthermore, we have analyzed our system in terms of lookup and delayed and have found that the introduction of this kind of security is not very time consuming. The average delay found for the Chord overlay was 480ms for 1 to 7 nodes and 530ms using KeyNote. In a same way, the average lookup was 130ms without trust management and 180ms with KeyNote.

The use of distributed hash is used to provide network storage as and to implement a decentralized trust server, in which every node belonging to the ring could contact KeyNote and determine whether a node should be allowed to join or not.

6. FUTURE WORK

Our approach provides a robust security towards malicious nodes joining the Chord network. However, once a node joins the network it can still behave in not the optimal way. Therefore, we will study the concept of reputation and our goal is to develop a robust and reliable dynamic trust management system for wireless networks. This will be done by analyzing peers previous behaviors and therefore assigning a specific trust value to each node in the Chord ring.

In order to implement this dynamic system, we proposed the extension of our trust server to a trust and evidence server which will collect evidence from different peers about a specific node's behavior and increase or decrease its trust value. This concept of reputation will not only provide additional security to the network in the sense that nodes will be forced to keep a benign behavior but also, we could selectively determine what information can be shared to what node depending on its trust value.

Another possible extension to our project based on the concept explained above is the implementation of additional assertions to the KeyNote trust management system. In this way, we could have certain policies for nodes joining the network and other policies for accessing and uploading data. Furthermore, we could expand the policies for accessing the network based on the intentions of the joining node.

7. REFERENCES

[1] http://w3.antd.nist.gov/wahn_mahn.shtml

[2] Y. C. Hu, S. M. Das, and H. Pucha, "*Exploiting the synergy between peer-to-peer and mobile ad hoc networks*", In Proc. of HotOS-IX, May, 2003.

[3] A. Singh, M. Castro, P. Druschel, and A. Rowstron, "*Defending against Eclipse attacks on overlay networks*". In Proceedings of the 11th ACM SIGOPS European Workshop, pages 115--120, Leuven, Belgium, Sept. 2004.

[4] M. Castro, P. Druschel, A. Ganesh, A. Rowstron, and D. S.Wallach. "*Secure routing for structured peer-to-peer overlay networks*". In *Proc. OSDI 2002*, Boston, MA, Dec.2002.

[5] K. Aberer and Z. Despotovic, "*Managing trust in a peer-2-peer information system*", In Proc. of the Tenth International Conference on Information and Knowledge Management CIKM, 2001.

[6] A. Selcuk, E. Uzun, and M. Pariente, "*A Reputation-Based Trust Management System for P2P Networks*", CCGRID2004: 4th IEEE/ACM International Symposium on Cluster Computing and the Grid, 2004.

[7] S. Kamvar, M. Schlosser, H. Garcia-Molina, "*EigenRep: Reputation Management in P2P Networks*", In Twelfth International World Wide Web Conference, 2003.

[8] E. Sit and R. Morris, "*Security Considerations for Peer-toPeer Distributed Hash Tables*", In Proceedings of the First International Peer To Peer Systems Workshop (IPTPS), pages 261--269, Cambridge, MA, USA, Mar. 2002.

[9] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "*Chord: A scalable peer-topeer lookup service for internet applications*," in SIGCOMM, 2001.

[10] Matt Blaze, Joan Feigenbaum, and Angelos D. Keromytis, "*KeyNote: Trust management for public-key infrastructures*" (position paper). Lecture Notes in Computer Science, 1550:59--63, 1999.

[11] M. Blaze, J. Feigenbaum, and M. Strauss, "*Compliance Checking in the PolicyMaker Trust Management System*," Financial Cryptography 1998: 254-274.