

Peer to Peer Security

Jochem van Vroonhoven

Department of Computer Science

Faculty of Electrical Engineering, Mathematics and Computer Science

University of Twente

j.vanvroomhoven@student.utwente.nl

ABSTRACT

With the increasing popularity of peer-to-peer networks, new problems arise. One of those problems is the security of such networks. In the area of peer-to-peer security there are five goals: availability, file authenticity, anonymity, access control and fair trading [RiMo04]. The solutions that try to achieve these goals, can be divided into three categories [RiMo04]: identity, trust & reputation and incentives.

In this paper a critical survey of the various solutions in the area of peer-to-peer security is given.

Keywords

peer-to-peer, security, trust, reputation, identity, incentive

1. INTRODUCTION

Over the last few years peer-to-peer systems have become very popular. Especially file-sharing peer-to-peer systems like BitTorrent [Bi05] and KaZaa [Ka05]. Those systems are used to download and share files with other users.

There are no dedicated servers and clients in peer-to-peer networks, every system in such a network acts like both a server and a client. Every system that supports the peer-to-peer aware protocols can join it and use the peer-to-peer network. This “open” structure has many advantages, (see [DaGa03]): adaptation, self-organization, load-balancing, fault-tolerance, availability through massive replication and the ability to pool together and harness large amounts of resources.

However there are also some disadvantages about this open structure, because it is just as easy for a malicious peer to join the system as it is for a normal (i.e. non-malicious) one. Malicious peers can abuse the network in many ways. They can for example use it to spread viruses or they can try to make resources unavailable by attempting a denial-of-service attack on peers [DaGa03].

To make these kinds of networks more secure, many solutions have been proposed. These solutions try to achieve several goals. [RiMo04] states **availability, file authenticity, anonymity, access control** and **fair trading**. In [DaGa03] definitions are given of availability, file authenticity, anonymity

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission.

4th Twente Student Conference on IT, Enschede 30 January, 2006

Copyright 2006, University of Twente, Faculty of Electrical Engineering, Mathematics and Computer Science

and access control.

Availability is defined as the availability of files and peers in the system. Each node should be able to communicate with other peers and be able to offer access to resources that it contributes.

File authenticity deals with finding the authentic responses to a query. The authenticity of a file can be defined in multiple ways. In [DaGa03] four definitions are given: *oldest document*, *expert-based*, *voting-based*, *reputation-based*. With the *oldest document* approach the oldest response to a query is believed to be authentic. With the *expert-based* approach “experts” give their opinion on the authenticity of a file. With the *voting-based* approach several experts give their opinion on the authenticity of a file and all those opinions together give information about the authenticity. The *reputation-based* approach is similar to the voting-based approach, with the difference that in this approach the reputation of experts is taken into account.

Anonymity is defined as: making it difficult for peers to find out who created a file, who stores a file, who accesses a file and which documents are stored on a peer.

Access control deals with restricting access to resources to peers that have the right to access those resources.

Fair trading means that a peer who benefits from the network also contributes to it.

[RiMo04] also categorizes the solutions into three categories: **identity, trust & reputation and incentives**. The research that is done in the identity category focuses on researching solutions that achieve the anonymity and access control goals. In the trust & reputation category the research focuses on trying to achieve availability and file authenticity using trust systems. The research that is categorized in the incentives category deals with trying to achieve fair trading and availability of peers by researching various ways to incite peers to contribute to the system.

In all these categories there has been a lot of research. The peer-to-peer research group of Internet Research Task Force (IRTF) [IR05] considers that there is a need for a critical survey on this matter. An evaluation of these solutions is necessary for the research community, because it will provide a way to show what has already been done, what has to be improved and what are the open issues. This paper will answer the question: “*How can the security of peer-to-peer systems be improved*”. It will focus on the areas of **trust & reputation and incentives** and the way solutions in those areas deal with the anonymity goal. This will be done by conducting a literature study.

The paper will start with describing work that has been done in the area of *reputation & trust* in section 2. The various solutions in this area are described and compared.

In section 3 this paper will focus on the work that has been done in the area of *incentives*. The systems that are proposed on this

matter will be described and the differences between the systems will be discussed.

Section 4 will deal with *identity* in peer-to-peer systems. This sections will try to answer if it is possible to implement and use the various solution proposed in the areas of *reputation & trust* and *incentives* without giving up on the anonymity goal of peer-to-peer systems.

The conclusions and the answer to the main research question will be described in section 5.

2. REPUTATION & TRUST

This paper will focus on reputation systems that are design for decentralized peer-to-peer systems, because most of the peer-to-peer systems that exist to day are such kind of systems. The most popular peer-to-peer systems are data-sharing systems. In such a system users can search for the files they want to download and download them if they are available within the system. In decentralized networks, such content retrieval processes involve two phases: a *content search* phase and *content download* phase. Malicious nodes can misbehave in both these phases.

In the *content search* phase a node generates a query and sends it to all peers that it is directly connected to. These nodes send the query to the nodes they are directly connected with and those nodes send the query to their “neighbors”, etc. This process repeats itself until the TTL (time-to-live) of the query is 0 (every time the query is send the TTL is decreased by one). If some of these nodes have the queried content they send a reply to the node that sent the query. If the querying node has received all the replies it can select one of the nodes to download the content from. Subsequently the *content download* phase starts.

In the *content download* phase the requested content is downloaded from one of the in the *content search* phase selected peers.

The process of content search and content download is a process that requires cooperation among the nodes of the system. Malicious nodes can disturb this cooperation. They can for example choose to not forward a query to the nodes they are connected with or they can choose to forward it after changing the content. Or they can send a reply to a query even if they do not have a proper copy of the queried content.

If such a malicious node is selected to be the downloading node, it can provide content that the querying node didn't ask for, e.g. a virus or adult content instead of the new Disney film.

These problems are serious security issues. To solve these problems various reputation systems have been proposed. These systems try to filter out the peers that misbehave by giving reputation to peers. If a peer misbehaves then it will get a bad reputation. A node with a bad reputation won't be selected to operate as a downloading peer.

This section will describe various reputation systems, the problems with reputation systems and the differences between the systems.

2.1 General problems

Reputation systems seem to be a nice solution to the problems described earlier, however when designing such a system, several problems arise (see [DaCa02]). This subsection describes these problems.

Pseudonyms

Most peer-to-peer systems use pseudonyms to identify users. This is to ensure some anonymity. If a reputation system uses this same technique, then several problems arise. Because it is easy to get a new identity by creating a new pseudonym, it could be easy for peers to start over again, once they got a very bad reputation. A solution to this problem could be to design a system where the reputation of a peer is never lower than the reputation of new users. This solution has a problem: cold-start. The cold-start problem is described later in this paragraph.

Pseudospoofing

The pseudonym mechanism to identify users has another problem. A user can easily create multiple pseudonyms and run them for example to create fake witnesses for one of their other pseudonyms. In this way it can be easy to get a high reputation.

Shilling

Shilling is a problem similar to *pseudospoofing*. The difference is that with shilling the multiple identities are created with different real IP addresses.

Cold-start

When there are several peers in a reputation system with a high reputation it is likely that those systems are selected for downloading content. If a new user joins the system he has a low reputation or no reputation at all. This means that he won't get selected for downloads as much as other peers with good reputation (or not at all). This way it is very hard to build up reputation for new users.

Load problems

If several files are shared by peers with a high reputation and peers with a low reputation, the peers with a high reputation will be selected to download the content. This means that resources of peers with high reputation will be used a lot more than that of low reputation peers. A load-balancing technique could be useful to avoid this kind of problems.

2.2 Reputation systems

Depending on the value of a specific file to a user, the user weighs the risks of downloading that file. If a user wants a file very badly he will take more risks getting it. The less he trusts a peer the higher the risk is for him to download the file from that peer. But on what ground can a user decide to trust a peer or not? In current peer-to-peer systems that depends on the requesting user, reputation systems can provide a way to base trust on the actual behavior of a peer. In this paragraph various reputation systems will be described and discussed. [KaSc03] states four important issues that should be addressed by reputation systems. We will use these as criteria to compare the systems. The issues are:

Self-policing: a peer should be able to apply to the reputation system without the help or supervising of a central authority

Anonymity: the reputation system should use indirect identities such as pseudonyms of usernames, and in no circumstances use IP addresses etc. to identify users

Overhead: the systems should have a minimal overhead in computation, infrastructure, storage and message complexity

Malicious collectives: The system should be robust to collectives of peers that know each other and will attempt to collectively subvert the system

The results of this comparison are explained in the following subsection and are shown in table 2.1. Because it is very hard to

draw hard conclusion about the overhead of the various reputation system, this paper will only discuss possible overhead problems. It is not possible to derive conclusions about which system will have the lowest overhead. Therefore, this is not included in table 2.1.

2.2.1 Reputation system by [GuJu03]

In [GuJu03] such a reputation system is described. This system uses two reputation schemes: *debit-credit reputation computation* (DCRC) and *credit only reputation computation* (CORC). Under both mechanisms, a peer can choose not to have his reputation tracked, in which case it will always have a reputation score of 0, i.e., the minimum reputation score allowed by the system.

The DCRC-mechanism credits peer reputation scores for serving content and debits for downloading.

The CORC-mechanism credits peer reputation scores for serving content, but offers no debits for downloading content. Instead expiration on the scores serves as a debit.

Both mechanisms offer extra credits for query processing and forwarding and staying online.

The reputation scores of a peer are stored locally. To ensure trustworthy reputation scores this system uses a central *reputation computation agent* (RCA) to credit reputation points. So this system is not fully decentralized.

Peers that want to enroll in the reputation system generate a (public, private) key pair and registers it with the central RCA. The RCA itself has a (public, private) key pair. The public key of the RCA is known to all enrolled peers and all peers can obtain the public keys of other peers.

The RCA is contacted periodically by the peers to get credits for their contributions to the system. For every action the peer makes he saves a *proof of processing* (PP). A PP is a parameter that stores the combination of the identity of the interacted peer, time stamp and additional information about the interaction. These PPs are used by the RCA to compute the reputation for that peer. The RCA encrypts the reputation score with his private key and the peer that requested the reputation score saves it locally. The system has various mechanisms to ensure that peers can not tamper their PPs, the precise working of these mechanisms is beyond the scope of this paper.

The reputation of a peer can be requested and when given decrypted with the public key of the RCA.

Discussion

Self-policing

Investigating this system, we see that it doesn't address all the issues described in [KaSc03]. This system uses a central reputation agent and is thus not self-policing.

Anonymity

In the system peers are identified by a pseudonym, so the anonymity of the peers is high enough.

Overhead

The trust value of a peer is stored locally and can be retrieved very fast, so the querying overhead is low. This trust value doesn't have to be calculated at the querying moment, but it is calculated periodically by contacting the RCA. This calculation is accomplished based on the PPs concept, that requires a certain amount of storage space. The calculation isn't performed by the peer itself, so it won't affect the performance of the peer. Because all the trust information that is sent over

the network is encrypted using keys, there is some overhead in terms of computation. It is hard to say how much overhead there will be, because this is not yet researched.

Malicious collectives

The system can use two schemes: CORC and DCRC. In the CORC scheme there is no debit for downloading content. Due to this peers can help each other get a high reputation by continuously transferring files to each other. The system is protected to this abuse by analyzing the PPs with some simple algorithms

DCRC has debits for downloading. To get a high reputation and keeping it high, peers can create multiple identities and use one for sharing files and others for downloading. There is no solution to this problem in the system

Evaluation

To get a high reputation a peer only has to share files with others and forward every message that it is asked to forward. However there is no check in the system that prevents sharing malicious files, because, the downloaded files aren't rated, so even if the file that was downloaded was malicious it will still generate reputation. This is why this system will never prevent the abuses of peer-to-peer systems that reputation systems should prevent. So actually this system is useless as a trust system. It can be useful as the basis of an incentives system.

2.2.2 Xrep by [DaCa02]

Xrep, a reputation system described in [DaCa02] uses, unlike previous described systems, not only peer reputation, but also file reputation. It is a fully distributed reputation system.

The system extends the typical two phase search previously described to a 5 phase protocol. In the first phase, i.e., the search phase, a peer sends a query. Peers that have the requested file respond to the query. This response includes a digest of the file that matched the keywords included in the query.

The second phase is used for the selection of the proper peer for downloading. To find out the reputation of the peers and the files that were submitted in the responses to the query, the peer broadcasts to other peers. They respond with their IP address and with their view on both the offerers and the files.

In the third phase these votes are evaluated, to judge the reputation of the resources and the reputation of the downloading nodes, i.e., offerers. To do this the votes are checked. The peer contacts the peer that voted via the IP address that was included in the vote. This is to prevent shilling.

In the fourth phase the querying peer contacts the most reliable downloading peer and checks it to see if it really exports the queried resource.

In the last phase the file is downloaded from the selected peer. When the downloading is completed the file is checked against its digest to ensure the integrity and the peer will update its repositories with its opinion on the downloaded resource and its offerers.

Discussion

Self-policing

There is no third party that supervises the system. Because other peers are asked for their opinion there is no need for a supervisor, so the system is self-policing.

Anonymity

Peers are identified using pseudonyms, however the system has one shortcoming in this area. When a peer sends his opinion about another peer it includes its IP address. And so the peer that sends a report is not anonymous. This has some huge disadvantages. Malicious peers can for example query their own trust values and attack the peers that report a bad value.

Overhead

For every file a peer wants to download the peer has to poll other peers and calculate the trustworthiness of the offerers. This polling process takes a lot of time and takes a lot of resources in computation and infrastructure.

Malicious collectives

The system has a good defense against malicious collectives working together. The system has two mechanisms to avoid this.

The first is that the reports that are received upon querying a trust query are clustered by their IP address. This prevents a user to make multiple pseudonyms on the same IP address and let them report positive values for another pseudonym it controls.

The second mechanism makes it impossible to run multiple fake witnesses on with fake IP addresses, because every witness has to send its IP address and is checked by that IP address.

Also the resource reputation counters the problem. Because resources with a bad reputation won't be selected for download even if the offerer has a high reputation.

Evaluation

The combination of servant reputation and resource reputation used in this system seems to work very well. It tackles several problems where other reputation systems have such as: cold-start, pseudonym problems (changing identity in case of bad reputation) and load balancing (see section 2.1) problems.

The system doesn't suffer from cold-start problems with new users as other systems, because they can immediately participate in the system if they offer well known resources. Usually when a system doesn't suffer from a cold-start it is very easy to spread malicious content for malicious nodes. Because they can easily switch to a new identity and when they have gained a high reputation again they can repeatedly spread the same malicious content. This isn't the case with xrep, because the resource based reputation solves this problem. This resource based reputation has another advantage: it makes load balancing possible. A resource with a good reputation can be safely downloaded from every peer and not only peers that have a high reputation.

However, the disadvantage described earlier in the anonymity paragraph makes the system a lot less interesting. If that issue is solved, then the usability of xrep could significantly increase.

2.2.3 Privacy enhanced reputation system by [KiPe03]

The privacy enhanced reputation system described in [KiPe03] works similar to xrep. It generates, similar to xrep, a reputation for a peer based on the experiences of other peers with that peer. The paper does not describe the precise working of the reputation system or how it can be implemented on top of existing peer-to-peer systems. Instead [KiPe03] focuses on the way in which *trusted computing* (TC) can improve the security

and the reliability of such a reputation system. TC can provide a way to ensure that recommendations are correct and that the recommender as well as the software of the requester can not tamper them. This is because the TC ensures that the reputation software can not be altered.

In their trust model they use *trust values* and *confidence vectors* for a number of categories. They use multiple categories, because according to them, general trust is not available.

The *trust values* are in the range from 0..1. A trust value of a peer of 0 means that the recommender(s) have had bad experiences with that peer or that they haven't had any experiences with it. This is because it's very easy to obtain a new pseudonym.

The *confidence vector* is used to store meta-information. This is used to judge the quality of the trust values.

Discussion

Self-policing

The peers can run the system without the help or supervision of a third party, so the system is self-policing.

Anonymity

Peers are identified by pseudonyms, so the anonymity of the peers in the system is high enough.

Overhead

A peer that wants to download a file, tries to calculate the trust values for the offerers. It sends a trust query to other peers if it thinks that its own information about a peer isn't enough to calculate a good trust value. This is a better solution than the xrep approach, because it reduces the amount of queries that are sent over the network. But this won't improve the performance of new peers, because they don't have enough information about other peers in the beginning.

Also the reputation updates after the download is finished could take a lot of calculations.

Malicious collectives

This issue isn't addressed in the paper about this system. The article focuses on the privacy of peers and protection of recommendations using TC. It can be that they didn't implement any protection against malicious collectives or that they didn't describe it in their paper because the focus was on TC.

Evaluation

TC can contribute to reputation systems. It can ensure that the software that runs the system is reliable and so all the information that is given by peers can be trusted in the sense that it isn't tampered with by other peers. But TC requires extra hardware and that's a huge drawback, since computers used by consumers don't possess this hardware as a standard. Maybe this will be no problem in the future.

2.2.4 Reputation-based trust management by [SeUz03]

In [SeUz03] a reputation system is introduced that uses *trust vectors*. These are binary-vectors of n bits. A peer stores, for each peer that it has interacted with, a trust vector. In this trust vector a 1 represents a honest transaction and a 0 represents a dishonest one. New results are written at the most significant bit, shifting the present bits to the right. This trust vector is used to calculate a trust rating and a distrust rating. This is done by dividing the trust vector by 2^m for the trust rating and by

dividing the complement of the trust vector by 2^m for the distrust rating (m is the number of significant bits, this is stored for every vector).

The peers that respond to a query are grouped according to how the file hashes are provided. Furthermore, for every group a *trust coefficient* is calculated. The group(s) with the highest *trust coefficient* will be the group(s) that best fit the query and are relatively safe to download. The *trust coefficient* of a group is calculated as the average of the (local) trust ratings of the top n most trusted peers in that group. If there are less than n trust ratings locally available a trust query is issued for randomly selected peers in that group. The number of randomly selected peers will be n minus the number of *available trust ratings*. The responses to this query include both the trust and the distrust ratings. The values are then weighted by the *credibility rating* of the peer that provided them. The *credibility rating* is calculated using *credibility vectors*. *Credibility vectors* work similar to *trust vectors*, where a 1 represents a successful recommendation and a 0 an unsuccessful one.

The *trust vectors* are updated after the download by judging the downloaded file. If it is a successful download a 1 will be added and a 0 otherwise.

The *credibility vectors* are also updated after the download. Recommendations are a success if they gave a value that is in line with the way the *trust vector* was updated, else they are unsuccessful. So a negative recommendation will be a success if the download wasn't a success and a positive recommendation will be a success if the download was a success.

Discussion

Self-policing

The system is self-policing; there is no need for a third party to initialize or run the system.

Anonymity

The peers in the system are identified by pseudonyms, so the anonymity of the system is relatively high.

Overhead

The amount of trust queries that have to be sent depends on the amount of peers that are required to base a reliable trust value. If it's a high amount there will be a huge overload in sending messages and processing the responses to those messages. But the reliability of the calculated trust value will be high. With a low amount this value is less reliable, but the overhead is reduced. Of course this is all theoretic. Simulations have shown that if the amount of peers that are required for reliable trust values is set to 2 it is very effective. The gain from values higher than 2 is negligible.

A peer only stores vectors for every peer it interacted with. The peer doesn't store the actual trust value of peers. Instead every time a trust value is needed, it is calculated out of the vectors. The overhead of this system is higher than a system that stores trust values in terms of calculation.

Malicious collectives

The system is protected against the collaboration of malicious nodes in when it comes to voting. This is done by weighing the trust value that a peers reports about another peer by the credibility of that peer. This avoids malicious peers giving each other high values and others low values, because the malicious peers won't have a high credibility.

Evaluation

The simulations run by the authors of the article show that this is a very effective system and a solution is provided for all of the four issues as described by [KaSc03]. But I think there is a problem: the system hasn't got a solution for the cold-start problem like xrep has provided. Thus in this system it is very hard for new peers to get a high reputation.

2.2.5 TrustMe by [SiLi03]

In [SiLi03] a protocol for anonymous trust management is described. The article focuses on a technique to store and access trust values in an anonymous secure way.

They do this by assigning trust holding agents (THA) to a peer when it joins the network. This is done by a bootstrap server and these assignments are unknown to all peers, even to the new peer. The THA keeps track of the trust value for the new peer. To make all the communications with the THA secure their system uses keysets to encrypt/decrypt messages. These keys are generated by the bootstrap server.

When a node wants to know the trust value of a peer he broadcasts a trust query. The THA of that peer will receive the message and will respond with a broadcast that includes the requested information by the querying node. Because there is no direct connection between the querying node and the THA they both stay anonymous. The encryption of the response messages ensures that it came from a THA.

After the node has finished its download, it can file a report to the THA by broadcasting Proof-of-Interaction. The article doesn't describe how the trust values will be altered by the THA after receiving the Proof-of-Interaction.

Discussion

Self-policing

The peers can run the system without the help or supervision of a third party, so the system is self-policing.

Anonymity

Peers are identified by pseudonyms. The THA for a peer is unknown to all peers and due to the broadcast communication it will always be, because there is no direct communication. Therefore, it can be derived that the anonymity is very good in this system.

Overhead

Due to the fact that all the trust information that is sent over the network is encrypted using keys, there is some overhead in terms of computation. It is hard to say how much overhead there will be, because this is not yet researched. But this could however, be a bottleneck. The overhead of the overall system is relatively low compared to other systems, such as xrep. This is because a peer only has to send and receive one message in order to get the reputation of a peer.

Malicious collectives

In this system peers can collaborate to boost each others reputation. This can be done, because there the reputation of peers is stored on THA's. Due to this fact, the peers can exchange files and send positive reports to the THA. These report affect the reputation of a peer much more than in a system like xrep, where the reputation of a peer is based on the way multiple peers think about one peer. In this system the opinion of a malicious peer has the same credibility as other opinions. Therefore, it is not very hard for two or more peers to

boost each others reputation by repeatedly exchanging files and filing positive reports.

Evaluation

The system focuses on the anonymity of peers. It uses trust holding agents (THA) to store the reputation of peers. The THA's of a peer are unknown and all the communication is through broadcasts. This way the THA stays anonym and it can not be attacked this way. But the system also has some drawbacks. The system relies on bootstrap servers and can not be used in peer-to-peer systems that don't support them. Furthermore, the system has some weaknesses for malicious collectives. Peers can collaborate to boost each others reputation.

2.2.6 EigenTrust by [KaSc03]

In [KaSc03] the EigenTrust reputation system is described. EigenTrust uses the concept of transitive trust. This means that a peer puts more trust in the opinions of peers it trusts. Thus the trust values that peers give about another peer B to peer A, are scaled by the trust peer A puts in them.

The EigenTrust reputation system relies on a Distributed Hash Table (DHT) design. In the system each peer has a set of peers that holds its trust values, called score managers. These score managers are assigned by hashing the peers identity (using different algorithms to obtain multiple score managers). Peers are identified by their IP address. The trust values the score managers manage are global trust values and they are calculated using the local trust values that peers put in the score managers' *daughter peer*. This calculation process uses the concept of transitive trust.

If a peer wants to know the trust value of a peer it simply hashes its name and gets the IDs of the score managers. Then it can query them for the requested values.

Discussion

Self-policing

The peers can run the system without the help or supervision of a third party, so the system is self-policing.

Anonymity

In this system the IP address is used to identify a peer. By hashing the IP address, the identity of the score managers of the peer are found. Thus in this reputation system peers and score managers are not anonym.

Overhead

In this system the reputation score of a peer is stored on a limited amount of places. When a peer wants to know the reputation score of a peer he can easily contact the score managers of the peer. Thus in this system the overhead on the overall system is lower than in systems like xrep in which a lot of messages have to be sent.

Malicious collectives

The eigentrust reputation system stores reputation scores in a limited amount of places. Because of this the system could suffer from the same problem as the TrustMe reputation system. But this is not the case. In the Eigentrust reputation system the opinion of a malicious peer is less valuable then the opinion of a normal peer (transitive trust). Thus it is hard for collaborating peer to boost each others reputation by exchanging files and filing positive reports to the score managers.

Evaluation

The Eigentrust system works very well. It can reduce the amount of active malicious peers in a peer-to-peer network. But the system has one huge disadvantage: it ensures no anonymity to the peers. Because of this the system itself is very vulnerable to malicious peers. Malicious peers can easily find their score managers and try to take them out by a DoS attack for example. But anonymity is also a design goal of a peer-to-peer network, thus the Eigentrust system is not suitable to be used for reputation management in peer-to-peer systems.

Table 2. 1: Reputation systems comparison:

	Self-policing	anonymity	Robust against malicious collectives
[GuJu03]	No	Pseudonym based	CORC: yes DCRC: little bit
[DaCa02]	Yes	Pseudonym based, with shortcomings	yes
[KiPe03]	yes	Pseudonym based	No / unknown
[SeUz03]	yes	Pseudonym based	yes
[SiLi03]	yes	Very good Pseudonym based	yes
[KaSc03]	Yes	No	yes

3. INCENTIVES

Besides the reputation and trust problems, peer-to-peer systems face other problems, such as free riding. Most of the existing peer-to-peer networks assume that peers are cooperative. If they download data from other peers, it is assumed that they share some data themselves and contribute to the network. According to [SaGu02] this isn't the case. They found that many users are consumers; they make use of the system without contributing to it. This is called free riding. Free riding is a serious threat to peer-to-peer system. In [RaLi03] a little thought experiment is described that makes clear that free riding is a threat:

They consider a peer-to-peer system of N peers. The majority of these peers is a free rider and all the files that are available within the system are shared by a small collective. The peers that share files will lose their interest in the system when there are no interesting files for them to download. Because there aren't many peers that share it will not take long for some peers to reach the point where there a no interesting files for them and they will leave the system. If such a leaving peer is a sharing peer, the amount of interesting files decreases and it will not take long before other peers start to lose interest. This process can continue until there are no sharing peers any more and soon the system will stop being operational.

Besides this problem there is another problem caused by free riding:

Say a system has N peers and N_f of those N peers share some files. For the simplicity all the sharing peers share an equal amount of files that are equally popular. If there are on average R download requests per time unit, that would mean

that the peers have to serve R/N_f download requests per time unit. The smaller N_f is, the higher the load is on the peers in that group. This can lead to a point where peers that share files leave the system, because of CPU overloads.

To counter these problems of interest and overload, a peer-to-peer system could use incentives mechanisms. There has been research on incentives mechanisms on peer-to-peer systems and they will be described in this section. [BuAg03] states two forms of incentives mechanisms: *monetary payments* and *differential service*. The first form will be discussed in section 3.1 and the second form in section 3.2.

3.1 Monetary payments

With this incentive mechanism peers are paid to contribute to the system and pay to consume. In [GuJu03] (reputation system 1 in section 2.2) such a system is described. That system pays peers for serving downloads and for forwarding broadcasts. They have to pay to download content from other peers in the DCRC scheme. The other scheme (i.e., CORC) uses an expiration on the credits a peer got paid instead. The system described by [GuJu03] is not a very good reputation system (see section 2.3) and it isn't a very good incentives system either, because there is no penalty on peers for having a low score. Peers that have 0 credits can still download content from the system. And peers that have a high score will have a greater chance on being selected for downloading, without considering the fact that this might cause an overload situation on those peers. If these two aspects of the system are changed, it could be a good incentives system. It will not only incite a peer to share files, but it will also incite a peer to share popular files and maybe even to bring in new files that could become popular, because a peers gets credits for serving downloads. Thus he will get more credits if he shares popular files.

There haven't been many incentives systems proposed that are based on the monetary payments form. The reason for this might be, that when designing such a system one is faced with the same problems as with reputation systems: The credits a peer has earned form a score, but it is hard to trust these values, because it is hard to check them. The scores are based on downloads and uploads of peers and it is easy for a peer to lie about them. To counter malicious scores there has to be some mechanism for storing it. For example the system proposed in [GuJu03] that uses a RCA or a system like Xrep where the scores are stored on other peers. This makes the design of incentives systems harder than it has to be. If the incentive score could be based on the amount of resources a peer contributes to a system, then it will be for a peer to lie about. This is because the amount of resources a peer shares can easily be checked. But in such a design the scores of a peer cannot be represented by a credit system, where peers pay and get paid. Instead the peers get a better quality of service for contributing to the system (differential service).

3.2 Differential service

In incentives systems based on the differential service peers get a better quality of service for their contribution to the system. These incentive system could still be based on some score that is calculated out of the amount of downloads a peer has served or the amount of broadcasts it has forwarded, like in the system described in [GuJu03]. But that would cause the same problems like with reputation systems (see section 3.1). Instead it is much easier to use some other technique, a technique that can be

checked by other peers in an easier way than with a reputation score.

The system proposed in [GuJu03] uses *cumulative disk space*: disk space contribution over a fixed period of time. If a peer wants to download a file from another peer it has to provide its *cumulative disk space* value. Every peer accepts a download request from another peer with a probability that is based on the *cumulative disk space* value. Thus the increase of the quality of service in this system is that a peer that has a high *cumulative disk space* will have a higher probability that its requests will be accepted. *Drawbacks* from this system are the related to fact that the *cumulative disk space* is the disk space over a fixed time period and it is hard to check. If the time period is, for example a week a peer can easily lie about the disk space it contributed to the system at the start of the week and get a high value. The other disadvantage of this system is that it doesn't incite peers to share popular files. In this system a peer can contribute a lot of disk space, but with files nobody wants. This way the peer will get a high probability that it will get the files it wants, without really contributing to the system. This will not cost the given peer a significant amount of CPU power.

In [RaLi03] three mechanisms are described to provide incentives. The article describes three mechanisms that can be used to reward peers that contribute significantly to the system and penalize the peers that contribute less.

The first mechanism uses the amount of files a peer shares to calculate a utility value. This utility value represents the contribution of the peer to the system. The utility value is calculated on fixed times. To counter the free riding the paper proposes to limit the maximum number of files a peer can download within one time unit of the utility value. The disadvantage of this mechanism is that it is hard to check the amount of files a peer has downloaded from the system. Another disadvantage is that it is profitable for a peer to share a lot of very small unpopular files. An advantage of this system is that it is possible to check the utility value, because a peer can be requested to report which files it shares. Those files can be checked to verify if the peer really shares them.

The second mechanism takes the files size of the different files in account. This way the utility value isn't only based on the amount of files, but also on the file sizes. This mechanism limits the amount of bytes a peer can download to the utility value. Now it isn't profitable for a peer to share a lot of files, because they have to be of a certain size to get a high utility. But it still does not incite peers to share popular files.

The third mechanism uses the amount of files shared, the sizes of the files and the popularity of the files. The popularity of the files is calculated by the amount of times it is downloaded. Whenever a peer wants to download a file the offering peer checks if the utility value of that peer is higher than the size of the files he attempts to download. This mechanism incites peers to share files that have a certain size and that are popular. However, it is easy for a peer to lie about the popularity of a file. Thus there has to be some control mechanism. The peer could for example be asked to give the names of the peers that downloaded a file to check that they really downloaded it. Or the popularity of a file could be verified by asking other peers that share the same file.

The system described in [AnGr04] uses an exchange system to incite people to contribute to the system. Request from peers that can participate in exchange transfers are given a higher priority to peer that can't. The exchange doesn't have to be directly to the offerer, it can for example be a n-way exchange

(see figure 3.1). Non-exchange transfers are only served if there is enough capacity and if no other exchange transfer is possible.

This means that if a peer wants to download a file from an other peer, it as better changes that its request will be granted if it has files that other peers are interested in. This will incite peers to share a certain amount of popular files. The workings of the system can easily be checked and it is hard for a peer to lie about the files it has to offer. In combination with a good reputation system (to ensure peers don't provide fake files) this incentive mechanism can work very well.

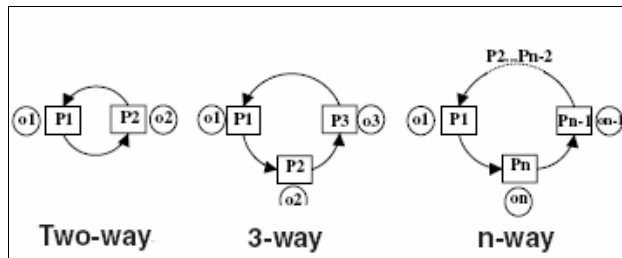


figure 3. 1: Two-way, 3-way and n-way exchanges

4. IDENTITY

In most of the peer-to-peer systems that are available today, users are identified by a pseudonym. The reason for this mechanism is to ensure a certain amount of anonymity. Anonymity can enable censorship resistance, freedom of speech without the fear of persecution, and privacy protection and it is one of the design goals of peer-to-peer system. In this section the following question will be answered: *Is it possible for reputation systems and incentive mechanisms to work without giving up the anonymity goal?*

4.1 Reputation systems

Various reputation systems have been discussed in paragraph 2. Most of these systems work with the pseudonym based identification. In this way users maintain their anonymity. But this system has one big problem: because in most peer-to-peer systems it is fairly easy to create new identities it is easy for malicious nodes to change their identity and continue to harm the system. Reputation systems try to counter this by designing a score system where users never get a score as low as the score of new participants. But this score system has also a disadvantage: in such a system it is very hard to build up reputation for new users. The system as proposed by [DaCa02] in the Xrep system could be a very nice solution to these problems. In the Xrep reputation system there exist two reputation mechanisms: node reputation and resource reputation. When using such a system it is easy for new users to gain reputation and it isn't as profitable for a malicious peer to switch identity as it is in other reputation system. We believe that if more research is done in systems based on the Xrep mechanism it could be possible to design a reputation system that works very well without losing the anonymity of the participants of the system.

4.2 Incentives

The incentive mechanisms that exist today can be divided into two categories: *monetary payments* and *differential service* mechanisms. Like described in section 3, the monetary payments mechanism use a credit system where nodes pay to download resources and get paid to contribute resources to the system. Whereas the differential service mechanisms provide better quality of service to the nodes that contribute more to the system. The monetary payments mechanisms can become

useful if some kind of identification is used. The identity of the node can then be used to update its credits when it downloaded something or contributed something. In the differential service mechanisms there isn't a need for an identification system, because there is no need to keep track of some kind of score. But even in the monetary payments mechanism it isn't profitable for a node to change its identity, thus a pseudonym based identification mechanism works well enough.

We therefore don't see any problem with the anonymity of peers in the various mechanisms proposed in the incentives category.

6. CONCLUSIONS AND FUTURE WORK

The security of peer-to-peer systems can be improved by using trust & reputation systems as well as good incentive mechanisms. Trust & reputation system provide a nice way to make it hard for malicious peers to abuse the system. Malicious peers can for example use peer-to-peer systems to spread viruses or by serving other unwanted content. Trust & reputation systems can locate these malicious nodes and can make it very hard for these nodes to continue their activities. However there are still some problems with the identity of nodes. It is fairly easy for nodes to switch identities. Reputation scores are linked to the identities of nodes. And therefore it is easy for nodes with a bad reputation to switch to a new identity in order to start all over again. The Xrep reputation system as described by [DaCa02] provides a nice solution to this problem. But the system has some shortcomings. Future research should focus on the idea beyond Xrep: node reputation combined with resource reputation.

In the area of incentives there has been a lot of research. Incentive mechanisms can provide a way to counter the free riding problem that most peer-to-peer systems face. We think that the focus on future work in this area has to be on solutions in the differential service form. Differential service incentive mechanisms give nodes that contribute to the system a better quality of service. Contribution to the system can be measured in the amount of resources a node contributes to the system and the quality of these resources. Because it is easy to check the quality and the amount of resources a node contributes to the system a node can not lie about his contribution. The other incentive mechanism that exists is based on monetary payments. In such a mechanism a peer gets paid for contributing to the system and has to pay to download from the system. Such a system is thus based on some kind of credit system. The credits a peer has reflects its behavior in the past. Because it is hard to keep track of the past behavior of a peer it is easier for a peer to lie about its contribution. Therefore, we think that it is easier and safer to design differential service mechanisms in peer-to-peer systems.

REFERENCES

[AnGr04] Anagnostakis, K, and M. Greenwald. "Exchange-based Incentive Mechanism for Peer-to-Peer File Sharing". Proceedings of the 24th International Conference on Distributed Computing Systems, ICDCS 2004.

[BaSu03] Bawa, M., Q. Sun, P. Vinograd, B. Yang, B. Cooper, A. Crespo, N. Daswani, P. Ganesan, H. Garcia-Molina, S. Kamvar, S. Marti and M. Schlosed. "Peer-to-Peer research at Stanford". ACM SIGMOD Record 32(3). 2003

- [BeEs04] Berket, K., A. Essiari and A. Muratas. "PKI-Based Security for Peer to Peer Information Sharing". Proceedings of the Fourth IEEE International Conference on Peer to Peer Computing. 2004
- [Bi05] BitTorrent internet site. <http://www.bittorrent.com/>. October 2005
- [BuAg03] Buragohain, C., D. Agrawal and S. Subhash. "A game theoretic framework for incentives in P2P systems". Proceedings of the Third International IEEE Conference on Peer to Peer Computing". 48-56, 2003.
- [CaWa03] Caronni, G. and M. Waldvogel. "Establishing trust in distributed storage providers". Proceedings of the Third International IEEE Conference on Peer to Peer Computing,. 128-133, 2003.
- [ClSa01] Clarke, I. O. Sandberg, B. Wiley and T. Hong. "Freenet: A Distributed Anonymous Information Storage and Retrieval System". International Workshop on Design Issues in Anonymity and Unobservability, H. Federrath, Springer. 2001.
- [CoNo03] Cox, L. And B. Noble. "Samsara: honor among thieves in peer to peer storage". Proceedings of the nineteenth ACM symposium on Operating System Principles, 120-132, 2003.
- [DaCa02] Damiani E., De Capitani di Vemercati S., Paraboschi, S., Samarati, P. and Violante, F. "A Reputation-Based Approach for Choosing Reliable Resources in Peer-to-Peer Networks". Proceedings of the 9th conference on computer and communications security, 207-216, 2002.
- [DaGa02] Daswani, N and H. Garcia-Molina. "Query-flood DoS attacks in grutella". Proceedings of the 9th ACM Conference on Computer and Communications Security; 181-192, 2002
- [DaGa03] Daswani, N., H. Garcia-Molina and B. Yang. "Open Problems in Data-Sharing Peer-to-Peer Systems". The 9th International Conference on Database Theory, 2003.
- [FiSa02] Fiat, A. and J. SAia. "Censorship resistant peer to peer content addressable networks". Proceedings of the 13th annual ACM-SIAM symposium on discrete algorithms, 94-103, 2002
- [FrMo03] Freedman, M. and R. Morris. "Tarzan: a peer-to-peer anonymizing network layer". Proceedings of the 2nd International Workshop on Peer-to-Peer systems, 2003
- [GuJu03] Gupta, M., P. Judge, M. Ammar. "A Reputation System for Peer-to-Peer Networks". 13th International Workshop on Network and Operating Systems Support for Digital Audio and Video, ACM, 2003.
- [HaWi02] Hazal, S. and B. Wiley. "A chord: A Variant of the Chord Lookup Service for Use in Censorship Resistant Peer-to-Peer Publishing Systems." Proceedings of the Second International Conference of Peer to Peer Computing, MIT Faculty Club, 2002.
- [IR05] IRTF peer-to-peer Work Group. <http://www.irtf.org/charter?gtype=rg&group=p2prg> October 2005
- [JoIs02] Josang A. and R. Ismail. "The Beta Reputation System". 15th Bled Electronic Commerce Conference, 1-14, 2002.
- [JoLo02] Joannidis J., S. Ioannidis, A. Keromytis, and V. Prevelakis. "Fileteller: Paying and Getting Paid for File Storage". In Proceedings of the Sixth International Conference on Financial Cryptography, 2002.
- [JoSi02] Josephson, W., E. Sizer and F. Schneider. "Peer to Peer Authentication with a Distributed Single Sign-On Service." The 3rd International Workshop on Peer-to-Peer Search, IEICE Trans. Commun. E86-B.1740-1753, 2002.
- [Ka05] KaZaa internet site. <http://www.kazaa.com>. October 2005
- [KaSc03] Kamvar, S., Schlosser, M., and Garcia-Molina, H. "The EigenTrust algorithm for reputation management in P2P networks". Proceedings of the Twelfth International World Wide Web Conference, 2003
- [KiPe03] Kinader, M. and S. Pearson. "A privacy-Enhanced Peer-to-Peer Reputation System". Proceedings of the 4th International Conference on Electronic Commerce and Web Technologies, volume 2378 LNCS, Springer, 2003.
- [MaGa04] Marti, S., P. Ganesan and H. Garcia-Molina. "DHT Routing using Social Links". The 3rd International Workshop on Peer-to-Peer Systems, 2004.
- [NgWa03] Ngan, T.-W., D. Wallach and P. Druschel. "Enforcing Fair Sharing of Peer-to-Peer Resources". Second International Workshop on Peer-to-Peer Resources, 2003.
- [OdVa04] O'Donnel, C. and V. Vaikuntantatham. "Information Leak in the Chord Lookup Protocol". Proceedings of the Fourth IEEE International Conference on Peer-to-Peer Computing, 2004.
- [PaSt04] Papaioannou, R. and G. Stamoulis. "Effective Use of Reputation in Peer-to-Peer Environments". Fourth International Workshop on Global and Peer-to-Peer Computing, 2004.
- [RaLi03] Ramaswamy, L. and L. Liu. "FreeRiding: A New Challenge for Peer-to-Peer File Sharing Systems". Proceedings of the 2003 Hawaii International Conference on System Sciences(P2P Track) HICSS2003, 2003.
- [Re05] Reputations Research Network. <http://databases.si.umich.edu/reputations/>. September 2005
- [ReLa04] Reldman, M., K. Lai, I. Stoica and J. Chuang. "Robust Incentive Techniques for Peer-to-Peer Networks". ACM E-Commerce Conference, 2004.
- [RiMo04] Risson, J. and T. Moors. "Survey of Research towards Robust Peer-to-Peer Networks: Search Methods". Technical Report UNSW-EE-P2P-1-1, University of New South Wales, Sydney, 2004.
- [SaGu02] S. Saroiu, P. K. Gummadi, and S. D. Gribble. "A measurement study of peer-to-peer file sharing systems". Proc. Of Multimedia Computing and Networking, 2002.
- [ScPa03] Schneidman, J. and D. Parkes. "Rationality and Self-Interest in Peer to Peer Networks". Second International Workshop on Peer to Peer systems, 2003.
- [Se02] Serjantov, A. "Anonymizing Censorship Resistant Systems". Proceedings of the Second International Conference on Peer to Peer Computing, MIT Faculty Club, 2002.
- [SeUz04] Selcuk, A., E. Uzun and M. R. Pariente. "A Reputation-based Trust Management System for P2P Networks". Fourth International Workshop on Global and Peer-to-Peer Computing, 2004.

- [SiKs04] Sieka, B., A. Kshemkalyani and M. Singhal. "*On the Security of Polling PRotocols in Peer-to-Peer systems*". Proceedings of the Fourth IEEE International Conference on Peer-to-Peer Computing, Zurich, 2004.
- [SiLi03] Singh, A. and L. Liu. "*TrustME: anonymous management of trust relationships in decentralized P2P systems*". Proceedings of the Third International IEEE Conference on Peer-toPeer Computing, 2003.
- [SiMo02] Sit, E. and R. Morris. "*Security Considerations for Peer-to-Peer Distributed Hash Tables*". First International Workshop on Peer-to-Peer Systems (IPTPS), 2002.
- [SuUp03] Surrudge, M. and C. Upstil. "*Grid security: lessons for peer-to-peer systems*". Proceedings of the Third International IEEE Conference on Peer-to-Peer Computing, 2-6, 2003.
- [ZaMo02] Zacharia, G., A. Moukas en P. Maes. "*Collaborative Reputation Mechanisms in Electronic Marketplaces*". proceedings of the 32nd Hawaii International Conference on System Sciences , 1-7, IEEE, 1999