A Reputation System for Peer-to-Peer Networks

Minaxi Gupta, Paul Judge, Mostafa Ammar
College of Computing, Georgia Institute of Technology
{minaxi, judge, ammar}@cc.gatech.edu

ABSTRACT

We investigate the design of a reputation system for decentralized unstructured P2P networks like Gnutella. Having reliable reputation information about peers can form the basis of an incentive system and can guide peers in their decision making (e.g., who to download a file from). The reputation system uses objective criteria to track each peer's contribution in the system and allows peers to store their reputations locally. Reputation are computed using either of the two schemes, *debit-credit reputation computation* (DCRC) and *credit-only reputation computation* (CORC). Using a *reputation computation agent* (RCA), we design a public key based mechanism that periodically updates the peer reputations in a secure, light-weight, and partially distributed manner. We evaluate using simulations the performance tradeoffs inherent in the design of our system.

1. INTRODUCTION

Peer-to-peer (P2P) networks have introduced a new paradigm in content distribution. Each peer is both a client and a server in these networks. Users are drawn to these networks due to the ability to locate a wide variety of multimedia content. The popularity of these networks can be judged by the fact that a recent study concluded that just the query-response traffic in 2000-2001 due to Gnutella comprised about 1.7% of the total traffic in Internet backbones in December 2000 [1].

P2P networks essentially come in three flavors: 1) centralized P2P networks like Napster, 2) decentralized unstructured networks like Gnutella and Kazaa, and 3) decentralized structured networks like CAN [2] and CHORD [3]. All are founded on the fundamental principle of cooperation among the peers. Getting content from a P2P network involves two phases: 1) *content search* and 2) *content download*. Content search in centralized P2P networks is facilitated by a central server.

In the unstructured and structured decentralized P2P networks, peers' willingness to share the content they have and forward the queries plays an important role during the content search process. Also, it is important that the peer that has been chosen to download from stays online during content download and serves good quality content. Using these *objective* criteria to track past peer behavior, we investigate the design of a *reputation system* in this paper.

Reliable peer reputations could be used in a variety of ways. They can help well-reputed peers find other peers with good reputations and hence help them in making decisions about who to serve content to and who to request content from. During the bootstrapping process for joining the P2P network, peers can potentially use reputations to decide who to directly connect to in the overlay topology.

A reliable mechanism to track reputations in P2P networks could act as a basis for an incentive system to motivate free-loaders[1] to be cooperative members of the P2P community. The presence of such peers is evidenced by several measurement studies of existing P2P systems like Napster and Gnutella. It has been reported [4] that at the time of the study, nearly 70% of Gnutella users shared no files, and nearly 50% of all responses were returned by the top 1% of sharing hosts. Also, the study in [5] quotes the free-loaders to be about 25% in Gnutella but much less in Napster.

Tracking peer reputations in a centralized P2P network like Napster is not difficult because the search for the content is facilitated by a central server. The lack of any central authority in the functioning of decentralized P2P networks makes the problem of accurate reputation tracking a challenging one. Unless *subjective* criteria of the type [6, 7, 8, 9, 10] are used, a certain amount of centralization is necessary to build a viable reputation system. Since unstructured P2P networks are the most prevalent today, the reputation system proposed in this paper mainly focuses on unstructured decentralized P2P networks like Gnutella, but it can easily be adapted for structured decentralized P2P systems also.

We propose two alternate computation mechanisms for a reputation system that objectively map each peer's activity in the P2P network to a dynamically updated *reputation score*. Both mechanisms essentially track the resources contributed to and used by the peers in the P2P network by means of a *non-negative* number of points representing a peer's reputation score. Unlike all the existing reputation systems where reputation inference requires on-demand processing, these schemes facilitate local storage of reputations for fast retrieval. The first mechanism, *debit-credit reputation computation* (DCRC) credits peer reputa-

[1]Free-loaders are peers who only download content but do not serve it to the other peers.

NOSSDAV'03, June 1–3, 2003, Monterey, California, USA.

tion scores for serving content and debits for downloading. The second mechanism, *credit-only reputation computation* (CORC) credits peer reputation scores for serving content but offers no debits. The expiration on the scores instead serves as a debit. Both DCRC and CORC offer additional credits for query processing and forwarding, and staying online. Under both mechanisms, a peer can choose not to have its reputation tracked, in which case it will always have a reputation score of 0, the minimum reputation score allowed by the system.

The reputation scores are intended to give a general idea of the peers' level of participation in the system. As a result, highly accurate reputation score computations are not necessary. However, since each peer stores its own reputation locally, for reputations to be reliable and effective, they have to be updated and stored securely to prevent malicious peers from thwarting the reputation system. An ideal solution will be light-weight, completely distributed, and compute trustworthy reputation scores. Since a fully distributed solution does not seem possible for objectively computed reputation scores, we introduce a partially distributed solution involving a reputation computation agent (RCA). This solution is light-weight and secure but trades some accuracy in the reputation score computations to keep the overheads to a minimum.

The rest of the paper is structured as follows. Section 2 details the proposed reputation system. Section 3 describes the security aspects of the reputation system. Sections 4 and 5 present the related work and the evaluation of various aspects of the proposed reputation system. Finally, section 6 concludes the paper.

2. DETAILS OF THE REPUTATION SYSTEM

In decentralized unstructured P2P networks like Gnutella, content retrieval involves a *content search* phase and a *content download* phase. To search for the desired content, a peer generates a query with appropriate *keywords* and sends it to all the peers that it is directly connected to in the Gnutella overlay topology. The peers who process this query reply back if they have the content in their shared directory and forward the request to the peers they are directly connected to depending on the TTL (time-to-live) of the query. This forwarding continues until the TTL specified by the querying peer is exhausted. Once the querying peer receives all the replies, it selects a peer to download the content from. At that point, the content download typically uses a HTTP or a TCP connection.

Cooperation among peers is required during both content search and content download. The success of the search phase requires that the other peers be online, agree to search for the content from their shared directory, and forward the query further depending on the hop count of the query. The success of the download phase requires that the chosen peer be online and serve the content when requested. Some additional factors come into play to create an overall experience for the peers in such systems. For successful content retrieval, the type, quality, and quantity of the content each peer places in the shared directory plays an important role. Further, the bandwidth at which the actual download occurs is also an important consideration. A high bandwidth querying peer is likely to have a better experience with the system if it downloads the content from another high bandwidth peer.

The above factors essentially differentiate the peers along the dimensions of *capability* and *behavior*. The capability of a peer depends on its processing capacity, memory, storage capacity, and bandwidth. The behavior of a peer is determined by the level of contribution offered by it for the common good of the P2P network. As the peers conduct content search and download functions, the proposed computation mechanisms map each contributing peer's *behavior* to form the first component of the reputation score for each contributing peer. The second component of the reputation score for each peer results from its *capability* (processing power, memory, bandwidth, and storage capacity). Figure 1 shows the various components comprising the reputation score.

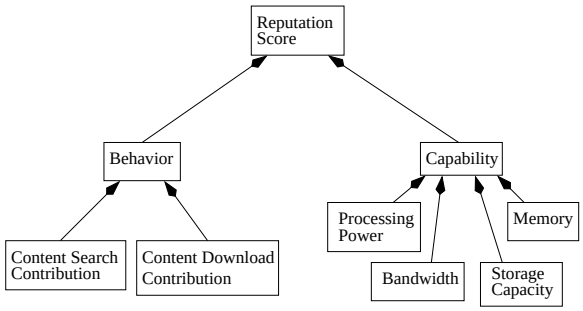


Figure 1: Components of the reputation score

Some peers may not want to get their reputation tracked for privacy reasons. Existing designs of P2P networks do not provide peer anonymity and our goal in this paper is not to propose alternate designs for Gnutella style P2P networks. As a result, the reputation tracking presented in this paper does not address anonymity issues in such tracking. Also, the reputation system involves additional overheads to keep the most up-to-date view of each peer's reputation which some peers may not want to incur. For these reasons, enrollment in the reputation computations is *voluntary*. Peers who choose not to enroll always maintain a default reputation score of 0.

Peers who enroll can enhance their scores by being good citizens of the P2P network. They can also save their reputation scores across sessions. Thus, a cooperative peer can maintain benefits of its participation in the system in spite of being off-line for a while. In a perfect world, each peer's local software can update and store its reputation score. However, this simple mechanism could be thwarted by the peers by altering the score computations to their benefit or by tampering with the value of the stored counter. The proposed solution to prevent such occurrences is discussed in section 3. The solution utilizes a *reputation computation agent* (RCA) for fair periodic updates to each enrolled peer's reputation, still ensuring that the reputation points for each peer are kept locally for fast retrieval. Note that the RCA is not involved during content search and retrieval functions and is therefore never a bottleneck for the normal operation of the P2P system. Figure 2 shows communication between the various entities in the system. The RCA is also used for enrolling peers who wish to enroll in reputation

computations.

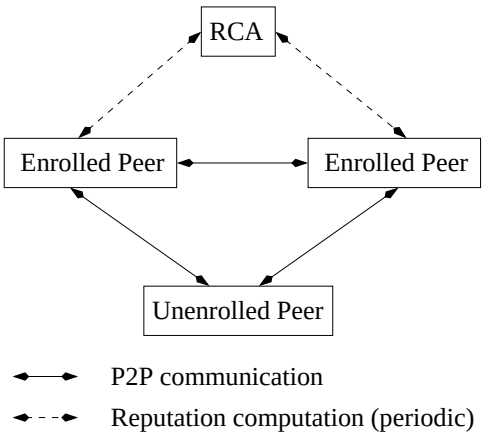


Figure 2: Components of the Reputation System.

Sections 2.1 and 2.2 describe DCRC and CORC schemes in detail. They assume that the updates to the reputation score take place locally. The actual mechanism to update the scores securely is discussed in section 3.

2.1 Debit-Credit Reputation Computation

DCRC uses three tunable system parameters: 1) *file size factor* f, $f \in integer$, 2) *bandwidth factor* b, $b \in real$, and 3) *time factor (in hours)* t, $t \in integer$. The file size factor, f determines how many MBytes of data transfer result in a unit increment to the reputation score. The bandwidth factor, b is used to classify peers into bandwidth quantums based on their bandwidths. Similarly, the time factor, t is used to determine the granularity at which peer cooperation by sharing and staying online is rewarded. These parameters are useful in ensuring that the reputation scores say within a certain range of non-negative values. They do so by tuning the system for larger file sizes, higher bandwidths, and longer stay in the system as the peer behavior and capabilities change over time.

Peers are likely to process many more query-response messages compared to the number of files they serve. Moreover, the size of most query-response messages is smaller than the size of the files served. As a result, DCRC updates reputation scores based on the size. Using the above parameters, a peer's total reputation score is computed using the following four components:

Query-Response Credit (QRC): DCRC uses average query-response message size to give credit to peers for being online and processing the query-response messages. If the average query response message size is denoted by QR, the number of points earned for each query and each response processed are given by:

$$QR$$

It has been reported [1] that the average query-response traffic in a Gnutella network is about $.75KB$ per second per connection. Also, most connections generated about 15 messages per second. This gives a rough estimate of the average query-response message size to be 0.00006MBytes. This value can be used to specify QR.

Upload Credit (UC): Each serving peers gets credit for serving content. A peer with bandwidth bw serving a file of size sMBytes earns points as:

$$\frac{s}{f} \times \frac{bw}{b}$$

Download Debit (DD): Every peer who downloads a file collects a debit for the download in DCRC. For a download of a file of size sMBytes, from a peer i with bandwidth bw_i, the peer earns debit points as:

$$\frac{s}{f} \times \frac{bw_i}{b}$$

Sharing Credit (SC): During the content search phase, in addition to peer availability, another important factor is the amount of content shared. Some peers may be sharing *hard-to-find* content. QRC, UC, and DC may not give any credit to such peers because by definition, such content may not be heavily accessed. SC is intended to capture this effect. As section 3 details, it is very hard to implement this correctly in a light-weight fashion. But assuming it can be implemented, if a peer shares n files where the size of jth file is given by s_j, at the elapse of each time factor, the number of points it will earn are given by:

$$\sum_{j=1}^{n} \frac{s_j}{f}$$

As section 3 describes, the periodicity at which debits and credits for QRC, UC, and DD are processed is chosen by the peers and is independent of the time factor that is used in processing SC. The total reputation score for a peer k who processes a query-response messages, facilitates b uploads, performs c downloads in d time factors is given by:

$$\text{Reputation Score}_k = (a \times QRC + \sum_{l} b \times UC_l - \sum_{m} c \times DD_l + d \times SC)$$

where UC_l and DD_m are the upload credit and download debit for files l and m respectively.

2.2 Credit-Only Reputation Computation

CORC updates the reputation scores essentially in the same manner as DCRC by using the computations described in section 2.1. The only difference is that the download debit (DD) component of DCRC is not used in CORC. This implies that the peer reputation scores only increase. To prevent peers from once gaining a good reputation score and then never contributing to the system, CORC time-stamps the reputation scores for expiration. Section 3 details various aspects of the DCRC and CORC schemes.

2.3 Other Issues

Some existing Gnutella-like P2P protocols facilitate parallel downloads [11]. Both DCRC and CORC can be used in such systems without any change. Both the schemes will give credit equivalent to sharing one file to each serving peer, but only according to the size of the download it facilitated.

Another issue is that of file popularity distribution. Both DCRC and CORC assume uniform file popularity distributions. They take into account file sizes but not their popularity distributions. In fact, several measurement studies

of Gnutella ([12, 13]) indicate that file popularity distributions are indeed non-uniform. We leave the investigation of how file popularities can be incorporated in debiting and crediting peer reputation scores for future research.

3. SECURE REPUTATION COMPUTATIONS

While describing the DCRC and CORC schemes, we assumed that the counter that keeps the most recent reputation score for each peer is updated and stored in the enrolled peer's local software. The local storage allows for fast retrieval of reputations. However, unsecured local updates and storage imply that the servent[2] software may be easily modified by malicious peers. As a result they can run a version of the software that thwarts the reputation computation schemes. This section describes a secure method to update and store reputation scores.

We assume that each peer interested in enrolling in the reputation computations generates a (public, private) key pair and registers it with the central reputation computation agent (RCA). The digest of the public key is used by the RCA to identify the peer. The lack of any infrastructure for secure key distribution simplifies deployment and overhead issues but leads to the problem of multiple identities. This issue is discussed in section 3.5.

The RCA does not effect the search and download functions of the P2P networks but is contacted periodically by the peers to get credits for their contribution in the system based on DCRC and CORC schemes. It can potentially be a central point of failure and needs to be replicated to make the system robust. The issues in the replication of RCA are beyond the current scope of this paper.

Several factors affect the accuracy of reputation scores. Some of these are: state maintenance at the RCA, network overheads, and loss of data while communicating with the RCA. However, since the reputation scores are intended to give a general idea of peers' level of participation in the system, certain amount of inaccuracy is acceptable.

We assume the RCA (public, private) key pair is denoted by $\{PK_{RCA}, SK_{RCA}\}$ and that each peer has access to RCA's public key and that the peers can obtain public keys of other peers in the system when needed. The assumption is that the RCA is not malicious but peers can be malicious and can collude with other peers in self-interest.

We now discuss how debits and credits for each of the components of DCRC and CORC schemes are securely computed by contacting the RCA.

3.1 Query-Response Credit (QRC) for DCRC and CORC Schemes

Enrolled peers receive credit for contributing to the system by being online and processing query-response messages. This credit is the same for both DCRC and CORC schemes. For every query-response message processed for a requester peer, a peer i with (public, private) key pair $\{PK_i, SK_i\}$ saves $\{$*requester_identity*, *query_keywords*, *query_size*, *time_stamp*, *self_identity*$\}_{SK_i}$ as the *proof of processing (PP)*. Periodically, the peer chooses to send these PPs to the RCA for receiving the credits. The RCA uses the PK_i to verify the PP indeed came from peer i. After processing the PPs, the RCA sends an encrypted reputation score $\{$*RCA_identity*, *time_stamp*, *reputation_score*, *peer*$_i$_identity$\}_{SK_{RCA}}$.

The above ensures that the stored reputation scores cannot be tampered with. However, it does not prevent peers from generating PPs without actually processing the query-responses. It also does not prevent peers from getting credit multiple times for the same receipts. Since the RCA is not involved in the search and download functionality of the P2P networks, it can not maintain any state by itself. As the peers contact the RCA to get credits to their reputation scores, the RCA can maintain *transaction state* of the form (*requester_identity*, *sender_identity*, *file_name*, *file_size*, *time_stamp*, *credit_processed_list*). The credit_processed_list is the list of peers who have already received the credit. It could have multiple entries for each peer depending on what type of credit it has already received. For example, the list could have one entry each for QRC, UC, and DD for each peer. The RCA uses this list to ensure that each peer receives each type of credit only once. The transaction state can also be used to prevent peers from generating false PPs.

To keep the state maintained by the RCA bounded, transaction state is maintained only for a certain duration. This implies that depending on the periodicity at which the peers request QRC, sometimes the state might be lost before they can get the credit. This impacts the accuracy of reputation scores, the extent of which is determined by the amount of state maintained and the periodicity of peer requests for QRC.

3.2 Upload Credit (UC), Download Debit (DD)

3.2.1 DCRC Scheme

The debit and credit to peer reputation scores in DCRC during the file transfer are also facilitated by periodic communication with the RCA. Let us denote the (public, private) key pair of the requester peers by $\{PK_r, SK_r\}$ and that of the sender peers by $\{PK_s, SK_s\}$. The following exchange takes place between the enrolled requester peer and the enrolled sender peer for DCRC at the time of the file download (if any of the peers are not enrolled, the following exchange does not take place):

- The requester peer sends a *requester portion of the receipt (RPR)* in the form of $\{$*requester_identity*, *file_name*, *file_size*, *time_stamp*, *other_info*$\}_{SK_r}$ to the sender peer. *other_info* might be the file popularity, if they are incorporated in processing UC and DC.
- The sender peer verifies the information using the requester's public key and stores $\{\{$*requester_identity*, *file_name*, *file_size*, *time_stamp*, *other_info*$\}_{SK_r}$ $\{$*sender_identity*, *sender_bandwidth*$\}\}_{SK_s}$ as a *receipt* of the transaction. At that point, it serves the content to the requester peer. The receipt is shown in figure 3.

The above sequence of receipt generation is important to ensure that the sender peer gets credit for its upload. Allowing the receiver to send the RPR after the transaction is not viable. This is because the requester peer may not send the RPR to avoid getting a debit to its reputation score. If the sender does not serve the content but generates the receipt to get the UC, the requester peer can report sender peer's malicious behavior to the RCA. Upon receiving many such complaints, the sender peer may be black-listed. Though the system offers no advantages for doing so, the misreporting is prone to peers colluding to malign the reputation of

[2]P2P software has both client and server functionalities built in, hence it is generally referred to as the *servent*.

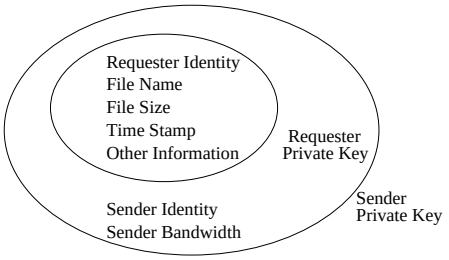


Figure 3: Receipt of the file transfer.

certain targeted peers and is a hard problem to solve in general. During the receipt generation, the senders also have the motivation to misreport their bandwidth to collect more credit. This is also not verifiable in a light-weight manner. These effect of these factors is to decrease the accuracy of reputation scores.

If any one of the peers involved in the transaction is not enrolled, the above receipt exchange does not take place. This is to ensure that peers who do not enroll in the program do not incur the overhead. In that case, if the sender is enrolled in the program, it would not be able to get credit for serving the file. This could potentially serve as a motivation for the senders to prefer transactions only with other enrolled peers, encouraging more peers to enroll in reputation computations.

Periodically, the sender peers send these receipts to get credits to their reputation scores. Using the transaction state, the RCA ensures a peer is not able to collect credit more than once for the same upload. After processing the receipts, the RCA sends an encrypted reputation score $\{RCA\_identity, time\_stamp, reputation\_score, sender\_identity\}_{SK_{RCA}}$ to the sender peer. Hence, the peers retain their own reputation scores locally for fast retrieval. At the same time, encryption by the RCA rules out tampering by the peers.

The credit processing also results in debits to the requester peers' accounts. To avoid having the requester peers drop negative reputation scores, the RCA retains the negative scores in the form of *debit state* with itself until those peers send some credits for processing. In order to limit the amount of debit state the RCA has to keep, the RCA will not keep debit state older than a certain duration. The duration for debit state will depend on the tradeoff of the state with the accuracy of reputation scores. This introduces some amount of inaccuracy in the reputation computation. However, as the reputation scores are intended to give a general idea about a peer's participation in the system, they can tolerate some inaccuracy. The reputation scores never go below zero. This is to ensure that an enrolled peer never has a lower reputation score than an unenrolled peer.

3.2.2 CORC Scheme

The receipt exchange in the case of CORC is almost identical to that of DCRC. The only difference being that the RPR is sent after receiving the content. This is to prevent the senders from collecting credits without actually serving the content. This receipt exchange is more fool proof compared to that in DCRC. This is because although malicious receivers in this scheme can choose not to send the receipt in spite of receiving the content, doing so is not advantageous to them because CORC offers no debits to requester peers. Further, reporting such requester peers to the RCA can prevent such occurrences. Figure 4 shows the receipt exchange in CORC scheme between the sender and the requester peers. It also shows the periodic credit processing for the sender peer by the RCA. The receipt processing is simpler for the RCA for CORC because it does not have to maintain debit state for requester peers.

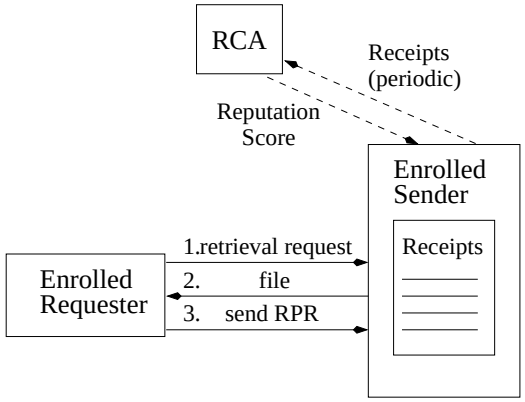


Figure 4: Communication for UC in CORC scheme.

After processing the receipts, the RCA sends the encrypted reputation scores for local storage to the sender peers, just as it does in the case of DCRC scheme.

3.3 Sharing Credit (SC) for DCRC and CORC

Both DCRC and CORC can optionally allow for credit to be given to the enrolled peers for staying online, based on the number of files they are sharing. There are two ways this can be done. Both the methods require extra work on the part of the RCA and induce some amount of error in the reputation score computations. The first method uses the transaction state maintained by the RCA to determine how long a particular peer was online. Then the RCA checks for the total amount of data shared by the peer by sending a special message to the peer. The inaccuracy of this method arises from the fact that the peer may have been online for longer and may not have processed any query-response messages. The second method involves periodic monitoring of the shared directories of peers by the RCA. This method is more work intensive for the RCA and is also more inaccurate because the credit depends on the monitoring frequency.

After processing the SC, the RCA sends the peer the encrypted reputation to the peer i in the form $\{RCA\_identity, time\_stamp, reputation\_score, peer_i\_identity\}_{SK_{RCA}}$.

3.4 Expiration and Consolidation of Reputation Scores

In describing how credits for query-response processing, file transfer, and staying online are processed for both DCRC and CORC schemes, we assumed that the time-stamped reputation scores are sent to the respective peers immediately (except for the debits in DCRC, that are retained at the RCA). In the case of CORC, the expiration duration for the reputation scores determines when the reputation scores expire. Too long a duration could mean that a peer has many

encrypted reputation scores from the RCA. This could potentially be a problem when the peers want to present their reputation to other peers in the network. However, a long expiration duration is beneficial for peers who go off-line for a long time. This is because a short expiration may not allow them to use their earned reputation when they come back online. In the case of DCRC, the time stamp is not important because of the allowance of debits to the reputation scores.

A straightforward consolidation of the reputation scores is not possible in the case of CORC because each score has its own expiration. However, in the case of DCRC scheme, the peers can periodically send their reputation scores to the RCA for consolidation and get one encrypted score back.

3.5 Multiple Identities

An issue that needs be carefully considered for reputation score computations is that of multiple identities. Because the reputation computations do not require any public key distribution infrastructure, peers can obtain multiple identities. Clearly, obtaining multiple identities is not useful if CORC is used because CORC has no debits for downloading. As a result, a single identity is more useful for a peer in increasing its reputation score. The peers in DCRC scheme can derive limited benefits by obtaining multiple identities. They could potentially earn a good reputation score for one identity by using it primarily for serving content. This identity can then be used scarcely for downloading content as well. Heavy load conditions may present one possible scenario for the use of the identity with a better reputation. When the load on the network is light, the peers could use the other identities for downloading content.

Because the enrollment in reputation computations is voluntary, there is no simple solution to this problem. Even sophisticated mechanisms for distributing (public, private) key pairs can be simply thwarted by generating another identity just for that does not enroll in reputation computations. However, if reputation scores are used in making decisions about who to connect to and who to do transactions with, an identity created just for downloading may not be very useful.

3.6 Collusion

Because the CORC scheme does not offer debits, peers can collude with each other to obtain a high reputation score. They can do so by transferring a set of files to each other continually. In the case of DCRC scheme, unless multiple identities are obtained, the debit component of DCRC prevents such an occurrence from happening.

Since the RCA maintains transaction state for some amount of time about transactions, it can run some simple algorithms to detect collusion among peers to increase their reputation score. This can be done by giving fewer credits if the same set of peers have many transactions with each other, especially if the files comprise a small set. However, care must be taken not to punish the peers who happen to perform many transactions with peers with whom they share common interests.

Table 1 summarizes the main differences between the DCRC and CORC schemes. If a scheme is better on some count, it is indicated in bold.

	CORC	DCRC
Relative safety of receipt exchange during file transfer	**More**	Less
RCA needs to maintain debit state	**No**	Yes
Possibility of collusion	More	**Less**
Consolidation of reputation scores possible	No	**Yes**
Short term misuse of reputation	Yes	**No**
Advantages from multiple identities	**No**	Yes

Table 1: Key differences between CORC and DCRC schemes.

4. RELATED WORK

Kazaa defines a *participation level* [14] for each peer based on the MBytes it transfers and the integrity of the files it serves. Each user rates the integrity of the files it downloads as *excellent*, *average*, *poor*, or *delete file*. Based on the ratio of Mbytes uploaded and downloaded and the integrity rating of the files, the peers are assigned to three categories: *low*, *medium*, and *high*. The participation level score varies between 0 and 1000. A new user starts at a *medium* participation level of 100. The participation level score is used in prioritizing among peers during periods of high demand. The security aspects in peers modifying their locally stored participation level values are not addressed.

Aberer and Despotovic [6] have proposed a *binary* trust model for P2P networks, i.e. a peer is either trustworthy or it is not. Assuming that maliciousness is an exception, the peers in this model only store information about their view of the malicious behavior of the peers they interact with. The overall trust is computed on the fly by querying appropriate peers. This system does not have any preventative measure against inserting arbitrary complaints about peers.

The proposal for tracking reputations in P2P networks by Demiani et. al. [7] involves keeping separate local repositories for resources and peers. They assume *binary* values for each of the repositories. Peers update their local repositories for the resources and their offerers upon finishing transactions. The criteria for such updates are subjective. To compute trust values for resources and the peers on the fly using votes, they enhance the 2 phase Gnutella search and download protocol into a 5 phase protocol called *XRep*. The first phase of this protocol enhances the resource searching to include sending a *digest* of the resource. Upon selecting a resource, in the second phase the querying peer broadcasts other peers to find out the reputations of the offerers and their view of the resource. The third phase involves the evaluation of votes to judge the reputation of the resource and their offerers. In the fourth phase, the querying peer explicitly checks the selected peer to counter any spoofing attacks. The final phase is similar to the download in Gnutella. While this work addresses many security considerations for both P2P networks and the reputation system, it offers no incentive to the peers to participate in XRep. Moreover, reputation inference in this proposal also involves on-demand computation.

EigenRep [8] is a reputation management system for P2P networks. Each peer locally stores its own view of the reputation of the peers it does transactions with. The global reputation of each peer is computed by using the local reputation values assigned to it by other peers, but weighted by

the global reputation of the assigning peers. This method of reputation inference rules out the possibility of malicious peers maligning the reputation of other peers.

NICE [9] is a platform for implementing cooperative distributed applications. Peers in NICE gain access to the remote resources by bartering local resources. The reputation in NICE is stored the form of a *cookie* which can take real values in the $[0, 1]$ interval and is based on a peer's subjective satisfaction from the transaction. As against all the above reputation systems where the locally stored reputation was an indication of that peer's view of the credibility of the peers it had had transactions with, the locally stored reputation in NICE is an indication of the satisfaction of others peers that it served. The system does not assume peers will store cookies that have low values. Since the peers store their own reputations, no cooperation is required from other peers for storage purposes. However, to compute reputations when needed, cooperation from the other peers in the system is a must, just like in the case of other reputation systems.

PeerTrust [10] is also a feedback based trust management system for peers to quantify and compare the reputation of other peers in P2P networks. The trust for a peer in this system is also a non-decreasing scalar and is subjective but differs from the other reputation systems in that it is computed based on the three factors: 1) the amount of satisfaction received by the other peers in the system, 2) the total number of interactions, and 3) a balancing factor to offset the impact of malicious peers that misreport other peers' service. Each peer is mapped to maintain a small database that stores a portion of the global trust data. Though this reputation storage scheme differs from that in other reputation systems, it still requires cooperation from the peers for storing the reputations. Maliciousness is countered by having multiple peers responsible for storing the same database. Voting can be used if these databases differ. Trust is computed on the fly through querying potentially multiple databases over the network.

5. EVALUATION

The focus of the evaluation is on the tradeoffs of the accuracy of reputation computations with state maintenance at the RCA, percentage enrollment in reputation computations, and the periodicity of processing debits and credits. The differences among the CORC and DCRC schemes are also evaluated.

Toward this goal, we generated 6 hour long synthetic P2P request logs with exponential inter-arrival time and an average request rate of 50 requests per second. The peer population in these logs varies from $10,000$ unique peers to $500,000$. The logs assume that all the peers stay in the system for the entire duration. The enrollment of peers in getting the reputations tracked varies from 25% to 100%. The peers access 100 unique files and the accesses to these files are are uniformly distributed. Apart from the information about the requester peers and the sender peers, the logs also simulate the peers that process the query-response messages for each transaction between the requester and sender peers.

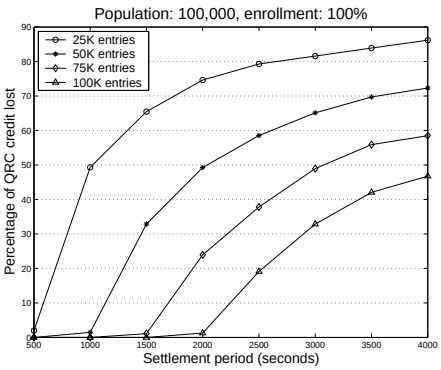
The accuracy of the reputation score computations is impacted by various factors in DCRC and CORC schemes. As section 3.1 pointed out, for both DCRC and CORC schemes, depending on the tradeoff between the transaction state maintained by the RCA and the periods at which the peers contact the RCA to get credit for query-response processing (QRC), some inaccuracy is introduced in the reputation scores. For the case when the enrollment is 100%, figure 5(a) shows the percentage of the QRC credit lost in a peer population of $100,000$ as a result of the time periods at which the peers contacted the RCA to get the credit (referred to as the *settlement period* in the figures). The reason for this loss of credit is the bound on the amount of state maintained by the RCA. In figure 5(a), the RCA maintains state about $25,000$, $50,000$, $75,000$, and $100,000$ file transfers. As can be seen from the figure, there is a clear tradeoff between the amount of transaction state maintained at the RCA and the settlement period. The trends for other population sizes were similar.

Since the participation in reputation computations is voluntary, the *receipts* involving upload credit (UC) and download debit (DD) in the DCRC scheme (just UC in CORC scheme) are only processed if both the peers involved in the file transfer are enrolled. As a result, when a file transfer takes place between an enrolled peer and an unenrolled peer, potential debits and credits are lost, leading to certain inaccuracy in reputation computations. Figure 5(b) shows the impact of percentage enrollment on the percentage of receipts that are not processed because one of the peers was not enrolled to get its reputations tracked. We show the results for the logs with a peer population of $500,000$, the results for other logs were similar. As expected, the percentage of unprocessed receipts peaks when the enrollment reaches 50%.

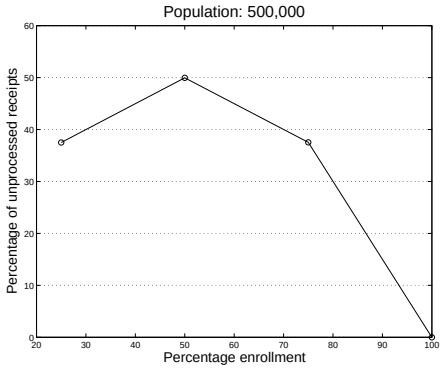
Figures 6(a) and 6(b) show the prominent tradeoffs of the CORC and DCRC schemes respectively. The time stamp on the reputation scores in the case of CORC scheme serves as an expiration. As the peers periodically get QRC and the credit for file transfer using receipts, they accumulate various denominations of reputation scores. This is an important consideration from the point of view of potential applications that require presenting the reputation scores. Figure 6(a) shows the average number of reputation denominations accumulated as a result of increasing expiration for a population size of $100,000$ when the enrollment is 100%. As expected, the longer the expiration, the more the denominations. A longer expiration duration is useful for peers who are off-line for a while but may pose problems when reputations are to be presented.

DCRC requires the RCA to maintain debit state for peers. This is necessary to ensure that peers do not drop the *negative* reputation scores. Thus, the amount of debit state maintained effects the accuracy of reputation computations. Figure 6(b) shows the effect of settlement period on the percentage of debit lost as a result of the amount of debit state maintained at the RCA. These results are also for the logs with peer population $100,000$ and an enrollment of 100%. Though the number of entries in the state maintained are different, the results of figure 5(a) and 6(b) show similar trends.

The periodicity at which peers contact the RCA for QRC and UC and DD receipt processing impact both the accuracy of reputation scores and the load seen by the RCA. For various enrollment percentages, figures 7(a), 7(b), 7(c), and 7(d) show the load on the RCA in terms of the average number of messages processed as a function of the periodicity at which the peers contact the RCA. Two things are noteworthy in these figures: 1) the QRC messages far exceed

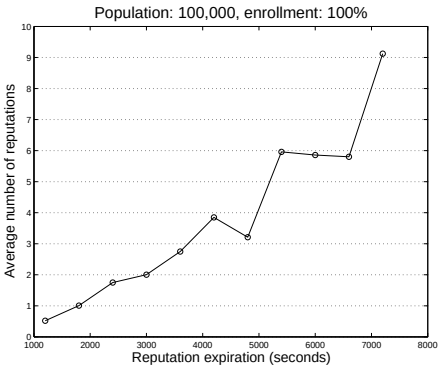


(a) Accuracy of QRC vs settlement time period.

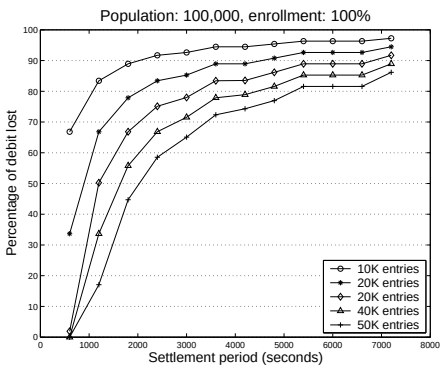


(b) Effect of enrollment on receipt processing.

Figure 5: Quantifying accuracy of reputation computations.



(a) Effect of expiration duration on the number of reputations in CORC.



(b) Accurate debit accounting vs settlement time period in DCRC.

Figure 6: Tradeoffs in CORC and DCRC Schemes.

the receipt messages, and 2) increase in enrollment implies more load on the RCA for both QRC and reciept processing.

6. CONCLUSION

This paper proposes a reputation system for decentralized unstructured P2P networks like Gnutella. The system comprises of two schemes (DCRC and CORC respectively) that utilize objective criteria for updating peer reputations. In order to secure the updates to reputations from peers who act in their self-interest, a partially distributed approach utilizing the RCA is proposed. Preliminary evaluation of the trade-offs of accuracy of reputation computations with overheads is presented.

7. REFERENCES

[1] M. Ripeanu, I. Foster, and A. Iamnitchi, "Mapping the gnutella network: Properties of large-scale peer-to-peer systems and implications for system design," *IEEE Internet Computing Journal*, vol. 6, no. 1, 2002.

[2] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, "A scalable content addressable network," in *ACM SIGCOMM*, Aug. 2001.

[3] I. Stoica, R. Morris, D. Karger, F. Kaashoek, and H. Balakrishnan, "Chord: A scalable Peer-To-Peer lookup service for internet applications," in *ACM SIGCOMM*, Aug. 2001, pp. 149–160.

[4] E. Adar and B. A. Huberman, "Free riding on Gnutella," Tech. Rep., Xerox PARC, 2000.

[5] S. Saroiu, P. K. Gummadi, and S. D. Gribble, "A measurement study of peer-to-peer file sharing systems," in *SPIE Conference on Multimedia Computing and Networking (MMCN)*, Jan. 2002.

[6] K. Aberer and Z. Despotovic, "Managing trust in a

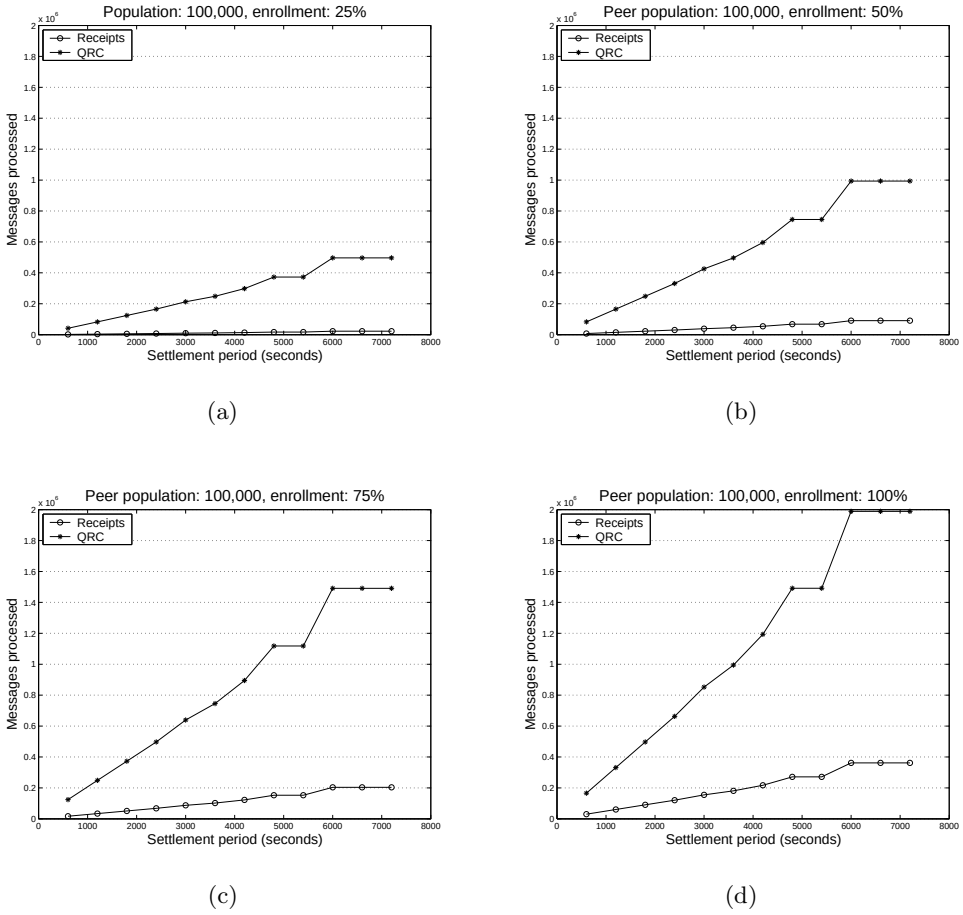


Figure 7: QRC and receipt processing load on the RCA.

peer-2-peer information system," in *Ninth International Conference on Information and Knowledge Management (CIKM)*, Nov. 2001.

[7] E. Damiani, S. De Capitani di Vimercati, S. Paraboschi, P. Samarati, and F. Violante, "A reputation-based approach for choosing reliable resources in peer-to-peer networks," in *9th ACM Conference on Computer and Communications Security*, Nov. 2002.

[8] S. D. Kamvar, M. Schlosser, and H. Garcia-Molina, "Eigenrep: Reputation management in p2p networks," Unpublished work, 2003.

[9] S. Lee, R. Sherwood, and B. Bhattacharjee, "Cooperative peer groups in nice," in *IEEE INFOCOM*, Apr. 2003.

[10] L. Xiong and L. Liu, "Building trust in decentralized peer-to-peer communities," in *International Conference on Electronic Commerce Research (ICECR-5)*, Oct. 2002.

[11] "Gnucleus home page," http://www.gnucleus.com/.

[12] K. Sripanidkulchai, "The popularity of gnutella queries and its implications on scalability," White Paper Featured on O'Reilly's website http://www.openp2p.com/, Feb. 2001.

[13] J. Chu, K. Labonte, and B. N. Levine, "Availability and locality measurements of peer-to-peer file systems," in *ITCom: Scalability and Traffic Control in IP Networks*. July 2002, vol. 4868 of *Proceedings of SPIE*, Proceedings of SPIE.

[14] "Kazaa participation level," http://www.kazaa.com/.