Reputation-Based Trust Management in Peer-to-Peer Systems: Taxonomy and Anatomy

Loubna Mekouar, Youssef Iraqi, and Raouf Boutaba

Abstract Trust is required in file sharing peer-to-peer (P2P) systems to achieve better cooperation among peers and reduce malicious uploads. In reputation-based P2P systems, reputation is used to build trust among peers based on their past transactions and feedbacks from other peers. In these systems, reputable peers will usually be selected to upload requested files, decreasing significantly malicious uploads in the system. This chapter surveys different reputation-based P2P systems. We will breakdown a typical reputation system into functional components. We will discuss each component and present proposed solutions from the literature. Different reputation-based systems will be described and analyzed. Each system presents a particular perspective in addressing peers' reputation.

1 Introduction and Motivation

The most important challenge of online environments is to assure positive interactions and satisfactory transactions. People will usually back up from dealing with strangers that they did not know before. Therefore, people minimize their interactions and transactions and tend to remain in their comfort zone. Peer-to-Peer file sharing systems provide a large collection of files available for download. In traditional systems, little information is given to the user to help in the peer-selection and file-selection processes. For example, if a user wants to download a file, the user is given a list of peers that have the requested file. The process of selecting the right peer with no a priori information is frustrating and risky. To foster positive

Loubna Mekouar
University of Waterloo, Waterloo, Canada, e-mail: `lmekouar@bbcr.uwaterloo.ca`

Youssef Iraqi
Dhofar University, Salalah, Oman, e-mail: `y_iraqi@du.edu.om`

Raouf Boutaba
University of Waterloo, Waterloo, Canada, e-mail: `rboutaba@bbcr.uwaterloo.ca`

interactions and reduce the risk involved in P2P file sharing systems, peers need to reason about trust and reputation systems are used to this end. Reputation systems are based on collecting information about peers' past transactions and computing a reputation value for these peers. The reputation values will be the basis for identifying trustworthy peers.

In a P2P system, peers communicate directly with each other to exchange information and share files. P2P systems can be divided into several categories. Centralized P2P systems (e.g., Napster [14]), use a centralized control server to manage the system while downloading a file is achieved directly between peers. These systems suffer from the single point of failure, scalability and censorship problems. Decentralized unstructured P2P systems try to distribute the control over several peers. They can be divided into completely-decentralized and partially-decentralized systems. Completely-decentralized systems (e.g., Gnutella [2]) have no hierarchical structure between the peers and all peers have equal role and responsibilities. Partially-decentralized P2P systems (e.g., KaZaa [4], Morpheus [5], and Gnutella2 [3]) occupy the middle ground between centralized and completely decentralized systems. These systems have been proposed to reduce the control overhead needed to run the P2P system by the use of "superpeers" or "supernodes". Supernodes are peers that have extra capabilities and assume more responsibilities than regular peers. A supernode act as a centralized server for the peers connected to it. Decentralized structured P2P systems like CAN [46] and CHORD [54] are based on Distributed Hash Tables (DHT) in that they use a hash function to deterministically map keys such as file names into points in a logical coordinate space. Peers are responsible for storing (key, value) pairs that are hashed into a point that is located within their region. Most P2P systems deployed on the Internet are unstructured.

In an open P2P system, peers often have to interact with unknown peers and need to manage the risks involved in these interactions. For example, if a user wants to download a file, the user is given a list of peers that can provide the requested file. The user has then to choose one peer from which the download will be performed. This process is frustrating to the user as no help is given on how to choose the right peer. After the download has finished, the user has to check the received file for malicious content and that it actually corresponds to the requested file (i.e., the requested content). If the file is corrupted, the user has to start the process again. In traditional P2P systems, little information is given to the user to help in the selection process. To solve this problem, peers need to be able to reason about trust in order to avoid untrustworthy peers and reduce risks. Trust management [13] is any mechanism that allows to establish mutual trust which allows peers to cooperate, and obtain in the long term an increased utility for the participating peers.

In [55], the authors classify trust management systems into three categories:

- Credential-based trust management systems: in these systems, service providers and the provided services are trusted, but service requesters are not. Service providers use credentials to evaluate the trustworthiness of service requesters and services may be granted or not.
- Reputation-based trust management systems: in these systems, service providers and provided services are not trusted. Service requesters select service providers

based on their reputation values. Reputable service providers are selected to provide the service.

- Social network-based trust management systems: these systems are based on social networks. Reputation is computed based on social relationships.

The focus in this work is on reputation-based trust management in P2P file sharing systems. Reputation-based P2P systems [13, 17, 20, 28, 29, 32, 36, 39, 42, 61] were introduced to build trust among peers. These systems try to evaluate the transactions performed by the peers and associate a reputation value to each peer. The reputation values will be used as selection criteria among peers.

In this chapter, we will investigate the existing research work proposed to address reputation in P2P systems. We will present existing reputation management schemes in centralized, completely decentralized and partially decentralized P2P systems. We will describe these schemes and identify key properties for each reputation system to summarize the efforts of researchers in addressing peers' reputation.

2 Traditional Systems Versus Reputation-Based Systems

The following is the life cycle of a transaction in a traditional P2P system (Fig. 1a):

1. Send a file request (either to the supernode or to other peers depending on the P2P system)
2. Receive a list of peers that have the requested file
3. Select a peer
4. Download the file

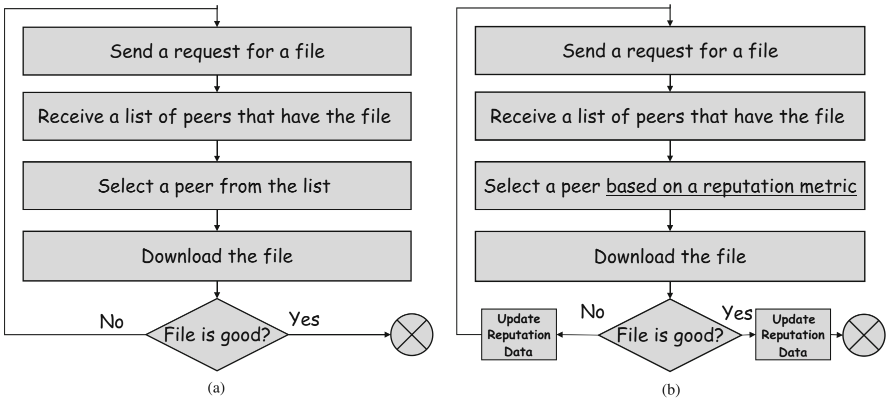
Send a request for a file
Receive a list of peers that have the file
Select a peer from the list
Download the file
No
File is good?
Yes
(a)
Send a request for a file
Receive a list of peers that have the file
Select a peer based on a reputation metric
Download the file
Update Reputation Data
No
File is good?
Yes
Update Reputation Data
(b)

Fig. 1 (**a**) Life cycle in a traditional P2P system, (**b**) Life cycle in a reputation-based P2P system

The following is the life cycle of a transaction in a reputation-based P2P system (Fig. 1b):

1. Send a file request
2. Receive a list of peers that have the requested file
3. Select a peer or a set of peers, based on a reputation metric
4. Download the file
5. Send feedback and update the reputation data

Most reputation management schemes try to achieve the following goals:

1. Make informed decisions about transaction partners and choose appropriate peers to download files
2. Isolate malicious peers from the network by downloading files only from reputable peers, hence reducing malicious uploads
3. Increase users' confidence and satisfaction
4. Help in the bootstrapping process to select peers to connect to in the overlay network.
5. Motivate peers to exhibit good behavior
6. Use the network resources more efficiently

3 Trust and Reputation

3.1 Trust Definition

Trust is as old as the existence of human beings on this earth. People were grouped in tribes and within the same tribe, they trusted each other. The concept of trust has a significant role in the surviving of human beings. We experience and rely on trust on daily basis. However, trust is difficult to define clearly and precisely.

Researchers from different fields such as psychology, sociology, philosophy, history, law, business and economics have tackled the concept of trust from different views. According to the Oxford dictionary [6], *trust is a firm belief in the reliability, truth, ability, or strength of someone or something.*

In 1973, Deutsch [21] has specified that *trust is the confidence that an individual will find what is desired from another, rather than what is feared.*

In 1990, Diego Gambetta [25] defined trust as follows: *Trust (or, symmetrically, distrust) is a particular level of the subjective probability with which an agent will perform a particular action, both before we can monitor such action (or independently of his capacity of ever to be able to monitor it) and in a context in which it affects our own action.*

In 1996, McKnight and Chervany defined trust as *Trust is the extent to which one party is willing to depend on something or somebody in a given situation with a feeling of relative security even though negative consequences are possible.*

In 2000, Grandison and Sloman [26] defined trust as *The firm belief in the competence of an entity to act dependably, securely and reliability within a specified context.*

In this chapter, we adopt the following definition proposed in 2006 by Chang et al. [15]. Trust is defined as *The belief the trusting agent has in the trusted agent's willingness and capability to deliver a mutually agreed service in a given context and in a given time slot.*

Based on our experience in the physical world, we extract the necessary information that can help us build trust in the virtual world to increase users' confidence and reduce the risk. Marsh [35] is one of the first researchers to give a formal model of trust that can be used in computer science. This model is based on social properties of trust taken from sociology.

3.2 Reputation Definition

Reputation has been widely used in different disciplines such as psychology, sociology, business and economics. From the Oxford dictionary, *reputation is the beliefs or opinions that are generally held about someone or something*. Abdul Rahman et al. [12] define reputation as "*an expectation about an agent's behavior based on information about its past behavior*". Sabater et al. [49] define it as an "*opinion or view of one about something*". Mui et al. [41] define it as "*the perception that an agent creates through past actions about its intentions and norms*". In service-oriented environments, Chang et al. [15] define reputation as *"an aggregation of the recommendations from all of the third-party recommendations agents and their first, second and third hand opinions as well as the trustworthiness of the recommendation agent in giving correct recommendations to the trusting agent about the quality of the trusted agent*".

3.3 Trust Properties

Trust is personal, subjective and it is based on various factors (i.e., endogenous and exogenous [15]). In [29], trust can be measured objectively by tracking peers' contribution to the system.

Trust is fuzzy since trust is imprecise and vague. Trust is dynamic since it is not stable and it changes as time goes by. Trust is also complex since different ways are possible for determining trust [15].

The trust relationship is usually asymmetric. The transaction between the trusting peer and the trusted peer results in a trust value assigned by the trusting peer to the trusted peer. This value shows the strength of the trust relationship.

The trust relationship can be transitive. In transitive trust [32, 33], if Alice trusts Bob and Bob trusts Claire and Alice wants to interact with Claire. Alice asks Bob. Bob will refer Claire to Alice. Alice derives a measure of trust in Claire based on Bob's trust to Claire and Alice's trust to Bob.

3.4 System Model

3.4.1 Modeling the Network

The success of file sharing applications has been behind the tremendous attention given to P2P technology. As a consequence, this technology has been applied to different distributed applications (e.g., Skype [8], PPLive [7], UUsee [9]). P2P applications have witnessed tremendous success among end users.

Centralized P2P systems use a centralized index for the files shared by each peer. It simplifies the direct exchange and the sharing of files between peers. However, it represents a single point of failure which reduces the reliability of the system. In completely decentralized P2P systems, a central authority for storing data and handling all the queries is not available. Interconnected peers are able to participate in transactions by interacting with each other and make local autonomous decisions to achieve their objectives. Peers are responsible for storing, sharing information and handling the queries. Peers act as clients and request services from other peers as well as servers and provide services to other peers. These systems provide improved robustness and enhanced scalability compared to centralized systems. In [55], the authors indicate the interplay between the degree of autonomy and the requirement of trust. In these systems, peers are autonomous which leads to loosing control over peers. Since no global view of the system is available, peers will get a partial and limited perspective of the system by receiving information from other peers. Trust is more needed to facilitate the interaction among peers. In partially decentralized P2P systems, some peers (named “superpeers”, “supernodes”) act as local central indexes for files shared by local peers. These systems provide lower discovery time as the discovery process involves only the supernodes.

In P2P systems, the following terms are used:

- Peers, users, nodes, agents: peers issue queries and reply to them
- Files, resources, items: the requested items from requester peers.
- Transactions: interactions and experiences between peers. In P2P systems, these interactions can be sharing a file, CPU cycles, or disk storage. A transaction in a P2P system involves:
  - The requester peer: the peer that requests the service.
  - The provider peer: the peer that has the requested service and is available to provide it to the requester peer.
  - Intermediary peer: the peer that is involved in a transaction. This peer participates to accomplish the transaction and to fulfill the request (e.g., forwarding messages). The intermediary peer is also referred as a third party peer or a mediator or a queried peer.
  - Recommender peer: peer that sends a feedback to the requester peer.
  - Feedback, opinion, rating, experience, appreciation: a peer sends a feedback as a result of a transaction. A rating is one trust value, while an opinion is

considered as an aggregation of all trust values based on previous transactions with a provider peer.

The following is a list of most important and representative reputation systems that have been proposed in the literature and will be used throughout the chapter: eBay [1], DistributedTrust [11], BinaryTrust [13], P2PBasic and P2PEnhanced [17], EigenTrust [32], Nice [33], DCRC and CORC [29], Travos [56], XRep [18], Regret [50], MDNT [15], MLE [20], LimitedReputation [36], CredibilityRecords [51], IDA and MDA [39], FineGrainedTrust [61], and PowerTrust [62].

3.4.2 Modeling the Nodes

In a P2P file sharing system, peers are expected to exhibit good behavior. Ideally, peers are expected to be trusted, that they will share good quality files, that they will upload requested files, and that they will send honest feedbacks. Unfortunately, real life P2P systems have proved that a mechanism is needed to measure explicitly trust in order to deal only with trustworthy peers.

There are two types of users' behavior: Good peers and malicious peers.

Good Peers
Those are the well-behaved peers that will usually send an authentic file. The following is the expected behavior from good peers:

- Availability: available to share a file, to forward a query, to reply to a query.
- Contribution: a peer contributes positively to the system by uploading authentic files.
- Credibility/Honesty: Upon receiving a reputation query, a recommender peer sends an honest feedback.

Malicious Peers
Peers that misbehave to impact badly the system. The adversary is another term to refer to malicious peers as used in [37]. The maliciousness of peers can appear in different ways and malicious peers can act individually or by forming collectives. Malicious peers can be one or more of the following:

- Senders of inauthentic data: these peers upload inauthentic and corrupted data and send spurious information to pollute the system.
- Liar peers: these peers lie in their feedbacks.
- Free riders: take advantage from the system without contributing to it. Although these peers do not harm directly other peers, they do have an impact on the performance of the system. In the literature, they are also referred to as selfish peers, free-loaders [29] or rogue peers [52].
- Traitors: peers that induce the milking phenomenon. These peers provide authentic files for a while to get a high reputation value before starting uploading inauthentic files. These peers abuse from this reputation by harming other peers, defecting on transactions and increasing the reputation of malicious peers.

- Colluding Peers: these peers know each other and work together to form a malicious collective. These malicious peers can upload inauthentic files, decrease the reputation of good peers (i.e., defaming), or increase the reputation of other malicious peers (i.e., flattering).
- Whitewashers: these peers join the system, act maliciously or misbehave then leave and re-join with new identities to get a new reputation value and eliminate the bad reputation received previously. In general, it is difficult to trace these peers and prevent the system from considering them as regular new comers. Systems with weak identities are vulnerable to whitewashing. Examples of these systems include KaZaA, Gnutella, and EigenTrust.
- Sybil Attacks: the malicious attacker creates multiple identities allowing the attacker to control a whole portion of the network.
- Denial of Service (DoS) attackers: these peers consume large amount of resources from the system to subvert the whole system and bring it down. In P2P systems, detection, management, and the prevention from these attacks is still an open problem.
- Unreliable: they can promise availability of specific services and do not deliver them.
- Imposter: they can pose as other peers.

Malicious peers act maliciously for different reasons, including personal characteristics, psychological, social, and economic factors. For example, movie and music industry pollutes P2P networks to minimize the use of these systems and to push users to buy their products rather than sharing them for free using P2P file sharing applications. Polluting these systems is achieved by distributing files with legitimate titles but with silence or random noise.

Peers Identity

To distinguish between peers, identity is required. Each peer in the system is identified by its identity that must be kept the same during its lifetime. As an example, users in Nice choose a PGP style identifier. The identifier includes a plain text identification string and a public key. Desirable characteristics of peers' identity related to trust include:

- Longevity: peers need to keep the same identifiers during their lifetime [47]. It should be difficult for peers to change their identity easily (i.e., whitewashing) and start over by obliterating their past behavior.
- Anonymity: according to the application used, peers' identity can belong to different levels of anonymity. Most P2P applications use a simple, user-generated pseudonyms. In some applications (e.g., freenet), onion routing is used to provide the anonymity of peers. However, since the peer identity is protected, it will be difficult to track the actions done by this peer. In general, there is a tradeoff between anonymity and trust.
- Unforgeability: Unforgeable identities are generated by a central trusted entity or a set of trusted entities and are given to new comers when they join the system. Each user is given only one identity. In TrustMe, for example, users are assigned

unforgeable identities. Unforgeable identity is used to protect from whitewashing (i.e., change of identity to eliminate peer's history) and sybil attacks (i.e., the use of multiple identities).

- Spoof-resistance: to prevent malicious peers from impersonating other peers identities.
- Sybil-attack resistance: peers identity should be resistant to sybil attacks. Some mechanisms have been proposed to slow the creation of new identities such as imposing an entry cost or a registration fee (e.g., Nice), requiring from users to solve a puzzle that can not be done by a computer, or read some text from a JPEG file.

New Comers Policy

When new comers join the system, these peers do not possess any local information regarding other peers in the system. In addition, all the peers in the system do not know yet the reputation of these new comers. Since new comers to the system are strangers and their behavior is not known yet, two policies can be adopted:

- The optimistic approach: new comers are trusted. However, other peers may be disappointed.
- The pessimistic approach: new comers are not trusted. However, in this case, new comers may not be able to build their reputation.

Reputation systems should welcome new comers and give them a chance to increase their reputation gradually. At the same time, these systems should protect peers from a new comer in case it turned out to be malicious.

New comers have to assume their responsibility in establishing trust with other peers by uploading authentic files. They should build and maintain good reputation to become reputable peers.

3.4.3 Modeling the Reputation Management Infrastructure

In centralized P2P systems, there is a central entity where information regarding peers' actions can be gathered and accumulated. Peers are controlled and malicious peers can be identified somewhat easily. Peers can be protected against malicious actions performed by malicious peers. For these systems, reputation management is also centralized since the central entity can handle the trust information in addition to the lookup mechanism. The use of a centralized reputation management infrastructure simplifies significantly the management of trust information. In completely decentralized systems, malicious peers have more freedom to act maliciously and perform variety of attacks and it is more difficult to gather all the actions performed by the peers. In these systems, peers should protect themselves from malicious peers since they can not be identified easily and hence isolated from the system. Peers are required to gather information to assess the reliability of peers in providing the requested service. In these systems, a centralized reputation management entity is possible. [29] is an example where the Reputation Computation Agent RCA is used to collect reputation information and compute peers' reputation. However, the use of a centralized entity will exhibit a single point of failure and will be easy to attack. A natural possibility is to use a decentralized reputation management infrastructure to improve the robustness and the scalability of the system. In partially decentralized systems, a centralized reputation management entity is possible. To avoid the drawbacks of the centralized entity, a partially decentralized reputation management system is proposed as in IDA and MDA [39]. In these schemes, the authors take advantage from the use of supernodes in the partially decentralized architecture. Since supernodes are already used to handle search requests on behalf of the peers connected to them, supernodes are used to handle trust data and compute peers' reputation.

The followings are advantages and drawbacks of different types of trust and reputation management infrastructures:

- Centralized systems: use a trusted centralized server for managing reputation information. This server will handle and manage all the reputation data
  - Advantages:
    1. Easy to use and manage
    2. Simplifies the mechanism design
    3. Offers efficiency since all the reputation data received for peers will be handled to the requester peer.
    4. Helps keeping the data consistent and coherent
  - Drawbacks:
    1. Difficult to achieve in case not all the peers can trust one entity.
    2. Represents a single point of failure and a bottleneck since millions of peers can send queries to this entity.
    3. Represents a single point of attacks by misbehaving peers such as DoS attacks, sabotage and subversion
    4. Expensive to achieve high performance and robustness
    5. Not scalable
    6. Not resistant to lawsuit
- Completely decentralized systems: information regarding peers' reputation is stored at the peer's level and this information is scattered throughout the network.
  - Advantages:
    1. No single point of failure
    2. No need for a globally trusted entity
    3. Provides robustness and scalability
    4. Resistant to lawsuit
  - Drawbacks
    1. Message overhead as many messages are generated in order to maintain and manage reputation data

2. Flooding is required to get the required information from peers in order to aggregate different local trust values
3. The topology changes frequently due to the transient nature of peers that can join or leave the system at any time. Some reputation data may be lost or inaccessible.
4. Reputation data is more vulnerable to tampering with.

- Partially decentralized systems
  - Advantages:
    1. Efficient search for reputation data of the provider peers compared to completely decentralized systems
    2. Easier to manage
    3. Resistant to lawsuit
    4. Scalable
    5. Small number of supernodes to manage reputation data
    6. Easy access to reputation data by other supernodes
    7. Alleviating the peer from the burden of replying to unnecessary queries as it is the case in completely decentralized P2P systems. Peers are contacted by their supernodes only if they have the requested file.
    8. Efficient enforcement of service differentiation
  - Drawbacks:
    1. Supernodes should be trusted to handle reputation data
    2. Additional load for supernodes

4 The Anatomy of Reputation Systems

The goal from file sharing applications is to share files, and sharing experiences between peers will ultimately lead to providing peers with the adequate environment by identifying good peers and isolating malicious ones. Reputation-based P2P systems create the appropriate environment for peers to rely on each other for downloading authentic files. Since we can not predict future peers actions, peers' past actions are used to measure its reputation.

Open systems need robust reputation management. A good reputation system should [30]:

- Identify malicious peers in order not to be selected as transactions partners: this is a precaution measure before starting the transaction.
- Spread information regarding a malicious peer in case that a negative transaction occurred: this is a revenge measure after ending a transaction to help other peers in future interactions.

The survey of different reputation systems reveals the important mechanisms used to achieve good reputation management. The peer looking for a file is the requester peer, and peers that have the requested file are the provider peers. The components and issues involved in finding a file and downloading it in the case of a reputation-based P2P file sharing application are the followings:

1. The Local Trust: Here important issues include:
   a. What kind of trust information is gathered?
   b. Where to store trust information?
   c. Is the local trust information sufficient?
2. The Reputation Query: The requester peer sends a reputation query regarding the potential provider peers. Related issues include:
   a. To whom the requester peer should send a reputation query?
   b. How many peers, should the requester peer contact?
   c. What kind of information is sent to the requester peer?
3. The Reputation Computation
   a. How to deal with liar recommender peers?
   b. How to deal with fraudulent recommender peers?
   c. How to compute the reputation?
4. The Use of Reputation
   a. How to choose a peer based on the reputation value?
   b. How to evaluate a transaction after downloading a requested file?
5. Credibility Assessment
6. Incentives, Rewards and Punishment

In the following, we will explain and describe each one of these important components and present important solutions proposed in the literature.

4.1 The Local Trust

The requester peer sends a file search query and gets a search reply from the system containing a list of peers. The requester peer may or may not have previously interacted with some of the peers in the list. If the requester peer is not a new comer, and has already downloaded files from other peers, local trust information based on previous transactions can be used. The requester peer may:

- Eliminate from the list malicious peers that it has already interacted with.
- Accept to deal with good peers that have already provided him a good service.
- May not have any trust information for some of the peers if it did not interact with them in the past.

4.1.1 What Kind of Trust Information is Gathered?

Some reputation management schemes uses the number of negative and positive downloads (e.g., EigenTrust), other schemes use the negative downloads only (e.g., complaints in Binarytrust). For other schemes the size of the download is more important than the number of uploads (e.g., IDA, MDA). In debit-credit reputation computation DCRC and credit-only reputation computation CORC [29], peers contribution is tracked and is considered as its reputation.

For each transaction, the following information can be gathered for each peer:

1. The type of trust information:
   - The quality of the transaction: positive, negative or both (e.g., satisfied and unsatisfied transactions in EigenTrust, complaints for negative transactions in BinaryTrust).
   - The quantity of the transaction: the size of the files downloaded as positive for authentic files or negative otherwise (e.g., satisfactory downloads and uploads in IDA and MDA).

   Relying only on positive transactions will lead to dealing with malicious peers that conducted few successful transactions. While relying only on negative transactions may eliminate good peers. A combined approach is more efficient in identifying the real behavior of the peers.
2. A trustworthiness value which represents the outcome of the interaction. This value can be a binary value 0 or 1 (e.g., XREP, Travos, CredibilityRecords) which means that a requester peer is satisfied from the transaction or not, the provider peer is trusted or not trusted. The trustworthiness value can also be a discrete value (e.g., Excellent, Good, Fair and Poor) as in Amazon and DistributedTrust, a scaled integer (e.g., 1–5) (e.g., MDNT) or a real value on a continuous scale from [0,1] (e.g., Nice, PeerTrust), a probability value (for EigenTrust and PowerTrust). While a binary value does not allow partial trust, a continuous value expresses better how much trust is given. However, assigning $\{0,1\}$ is much easier. The trustworthiness value can be the result of a transaction (e.g., a cookie which is a signed receipt of successful transactions in Nice) or computed based on either the quality or the quantity of a transaction or a combined approach. In DCRC and CORC, peers' willingness to share the content they have and forward queries in addition of staying connected in case they are chosen to upload the requested file is considered for computing the peers' score. In some reputation systems, a value of 0 from [0,1] does not make a distinction between a malicious peer and a new comer to the system.
3. The time of the interaction: this can be used when computing reputation values (e.g., FuzzyTrust, Regret, MDNT, FineGrainedTrust).
4. The context of the interaction: the context of the interaction can differ from an application to another. An application may have different contexts. In the file sharing application, we assume to concentrate on one context which is providing authentic, not corrupted files.

Peers may keep track of all the records with every trusted peer (e.g., eBay) or only one record that summarizes all the transactions (e.g., TrustModel). Adopting this approach will reduce the storage cost. Keeping track of trust data will allow to benefit from the followings [15]:

- Recency: giving more weight to the recent values of the trust values. In this case, the use of *time of interaction* is important.
- Trend: it is preferable to know not only the trust value at a specific time slot but also the trend line over the last few time slots. Understanding the trend is important.
- Cyclical poor performance: keeping history of trust values according to their time slots may help in detecting such behavior.
- Variability in trustworthiness: it is important to know the variance value to detect the variability of the service provided.

It is important to note that the focus is more on the trustworthiness of peers rather than files since malicious peers can generate a large number of inauthentic files. Thus, identifying and isolating malicious peers will improve significantly the quality of the shared files. In [18], a reputation scheme is proposed for both peers and resources (e.g., files) to overcome the limitations of reputation schemes that tackle only the problem of peers' reputation. In the XRep protocol, each peer maintains two experiences repositories to keep track of authentic and inauthentic resources in addition to good and malicious peers, the peer had direct transactions with. *Credence* is another reputation scheme for identifying content that is not authentic (e.g., damaged, corrupt, missing contents, dangerous content, misleading metadata designed to confound user searches). However, computing reputations for files is usually time consuming when dealing with a large collection of files.

4.1.2 Where to Store Trust Information?

Distributing the trust data history is a challenging task [47]. Provider peers may change their pseudonyms easily to erase prior history. Some mechanisms can be deployed to avoid this by imposing an entry fee (e.g., Nice) or by using real names. However, the anonymity of peers may be affected. But, it is also important to protect the identities of the peers from malicious peers.

While in centralized P2P systems, the central entity will be used to store trust information (e.g., eBay, amazon), in completely decentralized P2P systems, the trust information regarding a trusted peer can be stored at:

- The trusting peer's level: in this case, it is necessary to contact all the trusting peers that have interacted with the trusted peer to compute the trusted peer's reputation. This task is very challenging in completely decentralized systems where trust information is scattered throughout the network (e.g., CredibilityRecords).
- The trusted peer's level: in this case, trust information is gathered at the trusted peer's level (e.g., EigenTrust, Nice). This makes the reputation computation process easier, however, malicious peers may manipulate this information. A

solution for this problem is to store the trust information at another peer's level. In DCRC and CORC, each peer is responsible of storing its own reputation for fast retrieval. To prevent malicious peers from thwarting the reputation system, reputation scores are signed by the RCA private key. To avoid that trusting peers drop negative reputation scores resulting from downloading files, the RCA keep the negative scores until these peers send credits for contributing to the system.

- Both the trusting peer's and the trusted peer's levels: since negative transactions may be deleted from the trusted peers records, the trusting peers keep this information (e.g., Nice).
- Third-Party peers: trust data is stored at another peer instead of the trusting or the trusted peers. If the trusted peer is responsible to store its reputation data, it can easily manipulate it. To prevent against such threat, another peer is selected to be responsible for storing this data. Another problem that may occur is that this intermediary peer may report false trust information. To solve this issue, a set of peers is responsible to store trust information of each peer. This way, minimizing the impact of false trust reports. As an example, in secure EigenTrust, a distributed hash table is used to assign to each peer, score managers that are responsible to store and compute peers' global trust values. Also in BinaryTrust, the storage of the output of negative transactions is realized by other peers.

Since the local trust information stored at each peer's level in completely decentralized P2P systems is proportional to the size of the network, the storage cost increases linearly as the number of peers increases. Trust management schemes should be able to minimize the storage cost.

In partially decentralized P2P systems, trust information can be stored at the supernode level and sent regularly to the trusted peer (e.g., IDA). To avoid tampering with this data, reputation information is encrypted and signed by a shared key known only to supernodes.

When storing reputation data at peer's level, reputation-based P2P systems need to provide mechanisms to ensure that reputation data is stored securely to prevent malicious peers from thwarting the reputation system and also to ensure the availability of trust information even with the transient nature of peers in P2P systems.

4.1.3 Is the Local Trust Information Sufficient?

At this stage, it is important to know if it is sufficient to choose the trusted peer according to local personal trust information or maybe it is necessary to get feedbacks form other peers in the system. In case where a few well-behaved peers know each other and always exchange files between them, local trust information is sufficient for making decisions. In systems where millions of peers are available to share files, local trust information is very limited. Therefore, relying on external advice from other peers is imperative. Reputation systems help in predicting peers' future behavior based on their past behavior. Since reputation is based on peers' opinions, the credibility of the peers must also be assessed.

4.2 The Reputation Query

To gather more information regarding the peers that have the requested file, the requester peer needs to ask other peers about their opinions. The requester peer queries about the requested peers' reputations in a specific recommendation context.

In centralized reputation systems, there is no need to gather reputation data from other peers since the information is already collected and stored at the central entity. In [29], the RCA facilitates the collection of trust data and the computation of reputation scores. In partially decentralized reputation systems, supernodes are used to collect reputation data (e.g., IDA, MDA). While in completely decentralized reputation systems, on demand processing of peers' reputation is needed by querying peers and collecting reputation data to compute peers' reputation.

In reputation-based systems, the reputation scheme collects, distributes and aggregates feedbacks about peers' past behavior. This scheme will help peers to identify trustworthy peers and decide whom to trust. However, for reputation systems to operate effectively is very challenging. These challenges will be detailed in the following sections.

4.2.1 To Whom the Requester Peer Should Send a Reputation Query?

In the physical world, when recommendations are needed for a person for example, it is not practical to send a query to any person. It is preferable to target the people who know better this person and ask for their feedbacks (e.g., co-workers, friends). The same process can be applied for the virtual world, however, in completely decentralized P2P systems, it is difficult to find the peers who interacted previously with the provider peer.

The followings can be the receivers of the reputation query:

1. The neighbors of the requesting peer: the requester peer can use its neighbors in the overlay network to get reputation information. For example, in the basic polling (P2PBasic) and the enhanced polling (P2PEnhanced) algorithms (P2PRep protocol), the requester peer broadcasts a reputation query, asking for the reputation of the provider peers. In LimitedReputation, the requester peer sends a reputation query to its neighbors (Neighbor-voting).
2. The peers that have already interacted with the requester peer and that are trustworthy: the requester peer will not send a reputation query to a malicious peer. Most probably, the malicious peer's recommendation is not credible. Another alternative in this approach is to ask the trustworthy peers to ask the trustworthy peers that they have interacted with (e.g., EigenTrust). Using this transitive trust, the requester peer gets more recommendations. In FuzzyTrust, recommender peers are chosen based on their number of performed transactions and local trust values. In LimitedReputation, the requester peer can also select recommender peers from its friends (Friend-voting), where friends are peers who have proved to be reputable.

3. The peers that have already provided the requester peer with accurate recommendations in previous queries (e.g., MDNT): the requester peer keeps track of past recommendations and uses metrics to identify good recommender peers. These recommender peers are knowledgeable peers and their opinions/ratings are accurate.
4. Specific selection of peers: as an example of social-based trust systems, Regret groups peers according to their relationships with the trusted agent. The most representative agent is selected as a recommender peer based on fuzzy rules.
5. Random selection of recommender peers: the requester peer can choose randomly some peers for their opinion.

4.2.2 How Many Peers Should the Requester Peer Contact?

The requester peer can send a reputation query asking for recommendations to the followings:

- A small number of peers: for example, a Time To Live (TTL) can be used to limit the number of queried peers and reduce the generated communication overhead. The forwarding of the query continues until the TTL is exhausted. However, with the TTL, the requester peer will not get a wide view of the peers' reputation, but the communication overhead will be reduced significantly. In case of Nice, a query is forwarded to only 5 users instead of following all the paths from the requester peer.
- A large number of peers: the more information gathered, the more precise is the reputation of the provider peers. However, this approach incurs additional communication overhead. Each forwarded message consumes bandwidth and processing at each peer it visits. This approach may impact badly the performance and the scalability of the system.

Sending a reputation query and asking for a feedback means that the requester peer is asking for peers' vote on the provider peers. Reputation-based systems can be divided into the following categories:

- Excessive voting systems: these systems require the voting process to collect and aggregate reputation values for each provider peer. This will incur high amount of message overhead and will introduce additional latency. Examples of these systems are: DistributedTrust, BinaryTrust, P2PRep, XRep and EigenTrust
- Moderate voting systems: these systems require the voting process in a moderate way. The voting process is restricted only to some of the peers in the network. This mechanism reduces significantly the traffic overhead and hence, using the network resources more efficiently. Examples of these systems are: the work done by Marti et al. LimitedReputation [36] and Selcuk et al. CredibilityRecords [51],
- No voting system: these system require no voting mechanism for computing reputation values. However, a high volume of traffic overhead is generated due the

use of a central entity for computing reputation values. As an example of these systems: DCRC and CORC in [29].

In general, there is a tradeoff between accuracy and communication overhead. The larger the number of feedbacks received from recommender peers, the more accurate peer's reputation is. The number of recommender peers is determined according to the requirements of the P2P application. In addition, if providing a feedback is manually performed and a rewarding mechanism is not provided, the system may not be able to collect sufficient number of feedbacks to compute reputation scores accurately.

The feedbacks sent to the requester peer include information regarding previous interactions with some of the provider peers. It is important to know what kind of information is conveyed in feedbacks. The content of feedbacks is needed for both credibility analysis and reputation computation.

4.2.3 What Kind of Information is Sent to the Requester Peer?

For each provider peer, local trust information stored at the recommender's peer level can be sent to the requester peer in addition to the following information:

1. A confidence value: this value confirms how confident the recommender peer is in the trust value.
2. the time of recommendation: this time is provided when the feedback is generated (i.e., eBay, Nice, MDNT, EigenTrust, Travos, Regret).

The confidence value is usually based on the number of interactions that occurred between a recommender peer and a provider peer. A small number of interactions indicates uncertainty and does not reflect the real peer's trust value, while a large number of interactions increases the accuracy of the provided trust information. The confidence value can also be a subjective value given by the recommender peer based on its own experience. The concept of the confidence value is similar to reviewing a submitted paper and getting a feedback from a reviewer who indicates explicitly its expertise in the field. This represents how confident is the reviewer in the review. In Travos, the confidence value is a metric that represents the accuracy of the trust value based on the number of transactions. If the requester peer has a low confidence value in its assessment of trust for a provider peer, then seeking recommendations from other peers is necessary while, having a high confidence value means that the requester peer does not need to ask for recommendations. In fact, it could be that the recommendations received will add more misleading information than giving more accuracy to the peer's reputation.

At the queried peers, the trust information is filtered and only transactions corresponding to the context of recommendation are chosen to be sent to the requester peer. In case that recommendations pass through intermediary peers (e.g., PeerTrust, Nice, BinaryTrust, MLE) [48], additional information regarding these peers may be added to ensure the transparency of this process. In case of TrustModel, each intermediary peer adds some comments on the credibility of the recommendation received.

The identity of recommender peers can be revealed or not. In P2PBasic, the identity of peers sending the feedback is not required, while in P2PEnhanced, the identity of these peers is included in the feedbacks. This way, the requester peer is able to identify the peer sending the feedback and gives a weight to its feedback according to the trust given to this peer.

In some of the proposed reputation systems, the history of all trust transactions for a specific context are sent to the requester peer. In other systems (e.g., EigenTrust), only an aggregated trust value is sent to reduce the communication overhead and provide better scalability, but it lacks transparency. Aggregating feedbacks is a difficult task. The reputation score must represent the real behavior of a peer. The aggregation method of the trust values was not clearly described in most reputation systems. In Nice and TrustModel for example, any aggregation method can be used by each peer according to its requirements or by all the peers in the system to ensure homogeneity and avoid conflict of interests. Different approaches have been proposed for the feedback aggregation method: in Regret, the weighted average of ratings is computed with the use of recency while, in EigenTrust, the difference between positive and negative transactions is normalized. In Travos, the feedback is an aggregation of the number of successful transactions and unsuccessful ones. The feedbacks received from recommender peers must provide sufficient information for credibility assessment. Incentive mechanisms are required to encourage peers to send recommendations.

4.3 Reputation Computation

Eliciting feedbacks from the peers that received the reputation query and had already interacted with some of the provider peers rises several issues. These peers may:

- Reply to the query honestly: sending the right feedback is the expected behavior from a well behaved recommender peer.
- Reply to the query dishonestly by lying: sending the wrong feedback in order to defame competitors or flatter conspirators. Reputation management schemes should be able to minimize the impact of this threat.
- Ignore the query: a free rider will ignore the need of the requester peer and will not take into account the reputation query. Ignoring the query by peers that have already interacted with the provider peers is an act of unreliability.

According to the reputation management scheme, the intermediary peers may transmit or not the feedback to other peers. Upon reception of feedbacks from other peers to get transmitted to the requester peer, the intermediary peers could omit the feedbacks or maliciously manipulate the content. Some security mechanisms are required to make sure that feedbacks are received intact by the requester peer. As an example, the recommender peer's digital signature can be added to ensure the recommendation integrity. Reputation management systems should provide mechanisms to enforce good contribution and cooperation from the peers that receive the reputation query.

The requester peer receives feedbacks from recommender peers that can be one-hop or multi-hop from the trusted peer or even all the peers in the system. Different types of opinions may result:

- First hand opinion: is a direct opinion from the queried peer to a provider peer. The queried peer has already interacted directly with the trusted peer.
- Second hand opinion: is an indirect opinion from a peer, that has been contacted by the queried peer, about a provider peer.
- Third hand opinion: is a public opinion

Different weights can be given to the received feedbacks according to their type. Typically, a first hand opinion is more trustworthy and then more important than the other ones.

This is one of the most important issues in reputation management systems and unfortunately, only few research works have focused on solving this problem. If the feedbacks received are not credible, the reputation can not be computed accurately. The majority of reputation schemes assume that peers are well-behaved, honest and do not provide any mechanism to check the accuracy of the gathered opinions which will reduce significantly the reliability of the reputation system. This problem is also referred to as mis-representation. In eBay, for example, the credibility mechanism is left to members who can go through all the ratings and decide if a specific member is credible or not. Recently, the credibility assessment becomes an important issue since in e-commerce transactions where money plays a vital factor, accurate reputations are necessary in making the right decisions.

It is imperative to distinguish between two important issues when dealing with liar and fraudulent recommender peers:

- Liar recommender peers: the recommender peer had actually a transaction with the provider peer. The recommender peer may reply negatively even if the output of the transaction was positive. The recommender peer may lie and provide a dishonest feedback for different reasons (e.g., to increase the reputation of a colluding peer).
- Fraudulent recommender peers: the recommender peer did not interact with the provider peer before and it provides the requester peer with a feedback. The feedback can be positive if the provider peer is a colluding peer or negative if the provider peer is a competitor (i.e., a good contributor peer).

Detecting malicious recommender peers is vital to the accuracy of the reputations and hence, the reliability of the reputation system. A credibility analysis is required to filter out recommendations, eliminate the suspicious feedbacks and select the accurate ones. The content of recommendations form the basis of reputation computation.

4.3.1 How to Deal with Liar Recommender Peers?

According to [30], there are two basic approaches that have been proposed: Endogenous and Exogenous methods. In Endogenous methods, the statistical properties of the reported feedbacks are used to detect unreliable feedbacks. In these methods, most of the proposed mechanisms assume that liar peers are the minority among peers and so the ratings that are different from the majority of peers are inaccurate. Exogenous methods rely on other information such as the reputation of the recommender or its feedback accuracy given its past recommendations or its relationship with the provider peer.

The following are some mechanisms that have been proposed to minimize the impact of liar recommender peers:

- Keeping track of past recommendations (e.g., number of accurate and inaccurate recommendations) and weight the feedbacks according to the credibility of the recommender peers (i.e., MDNT, Travos, MLE, CredibilityRecords).
- The multivariate outlier detection technique: this technique is used to detect liar peers in FineGrainedTrust [61]. Outlier detection is an important task in data analysis. The outliers describe the abnormal data behavior which means data that is deviating from the natural data variability.
- The use of suspicious transactions to measure the credibility of recommender peers [39]: a suspicious transaction is one in which the feedback sent by the requester peer is different from the one expected knowing the reputation of the provider peer. Peer's credibility is computed based on the ratio of the number of suspicious transactions over all the transactions performed by the peer.
- The use of trust and reputation values as credibility metrics for the recommender peers (i.e., EigenTrust, Nice, FuzzyTrust, P2PEnhanced). Trustworthy peers are considered as credible while untrustworthy peers are not. This is based on the fact that if peers are behaving correctly by sending authentic files, these peers will most probably be honest in their recommendations. In Regret, social relationships are also taken into account.
- Redundancy: the use of different score managers to compute the trust value and use a majority vote to eliminate the false reports by malicious score mangers (e.g., EigenTrust)
- The use of Beta distribution based on previous recommendations as in Travos
- To let only one rating count from any single IP address
- Enforcing policies: in eBay, when a seller receive multiple feedbacks from the same buyer within the same week, the net effect on that seller's feedback score is based on the number of negatives, neutrals and positive received [22].

Many reputation systems (e.g., EigenTrust, fuzzyTrust, P2PEnhanced) use a reputation value for a recommender peer as a credibility value which is not totally accurate since some peers could behave well while sending authentic files and provide at the same time inaccurate recommendations. The credibility of recommender peers is based on local trust values. These values are not accurate enough to measure the trustworthiness of recommender peers and it is not practical to collect feedbacks regarding the credibility of the recommender peers. This approach will incur additional overhead and waste of network resources. Moreover, if a peer is malicious according to its reliability in sending authentic files (i.e., its reputation) it will be difficult to trust its recommendation and there is no guaranty that a reputable peer will provide an honest opinion.

4.3.2 How to Deal with Fraudulent Recommender Peers?

The followings are some mechanisms that have been proposed to minimize the impact of fraudulent recommender peers:

- Only peers who were actually involved in a transaction are permitted to provide ratings (e.g., eBay).
- The use of proof of interactions: in TrustMe for example, both entities participating in a transaction sign a transaction certificate. This proof is needed to report the output of the interaction.
- The use of proof of processing: in DCRC and CORC, the RCA sends proof of processing for peers contribution to the system by processing, forwarding queries, staying online and serving files. To prevent malicious peers from getting credits without actually participating, the RCA maintains a transaction state where the credit_processed_list contains the list of peers who have already received the credit. The RCA ensures that a peer will collect credit only once for the same upload.

It is important to indicate that even with the presence of proof of transactions in some reputation management systems, it is difficult to prove that effectively there was a transaction between the two parties. In case of a file transfer for example, it is needed to prove that the file transfer has actually occurred. This proof must also indicate all information regarding the file transferred (e.g., size), the peer ID uploading the file, the peer ID downloading it and the time of interaction.

4.3.3 How to Compute the Reputation?

After filtering the feedbacks and getting rid of dishonest reports and fraudulent ones, the gathered data from different recommender peers is used for reputation computation in addition to the local trust data available at the requester peer.

Some of the proposed reputation schemes assign more importance to:

- Bad transactions versus good transactions: in some reputation schemes, it is proposed that the negative impact of bad behavior on reputation should outweigh the positive impact of good behavior.
- Local trust data versus the reputation information gathered from other peers: since the local trust data is more credible than the data collected from other peers.

However, in P2P systems with millions of users, local trust data may not be very helpful in taking decisions.

- Recent transactions versus old ones: to take into account the recency of interactions, the time of interaction is needed. This will help in identifying the traitors. In some reputation systems, old transactions are deleted and the focus is more on the recent transactions. In DCRC, a time stamp is used each time the RCA sends the reputation scores to peers, while, CORC time-stamps the reputation scores for expiration.

In some proposed reputation systems, a peer's reputation is equivalent to the group of peers it belongs to. A group's reputation can be, for example, the average of all the members reputations within this group. If the group has a high reputation, all its members are reputable. A peer can be rejected from the group in case it turned out to be acting maliciously.

Reputation systems should be exigent in terms of accuracy of the computed reputation according to the risk involved in the transactions. However, several factors affect the accuracy of reputation values such as: network overheads, congestion and loss of trust data during peers' communication. In some circumstances, this inaccuracy may be acceptable since the goal from reputation systems is to provide an approximation of the real peer's behavior.

Different approaches have been proposed used to aggregate trust values received from recommender peers and synthesize them to generate a reputation value for a provider peer [15, 30]:

1. Deterministic approach: In this approach, peers' reputation is based on a simple summation or average of collected ratings. Even in the most popular e-commerce applications that involve a huge amount of money, reputation values can be computed easily by simple operations. The reputation scheme used in eBay, for example, is based on the sum of the number of positive and negative ratings, while in amazon the reputation is computed based on the average of all the ratings. A weighted average of all the ratings is also possible based on different factors such as the credibility of recommender peers. This approach is easy to use and can be easily understood by users. Many proposed reputation systems used this approach for reputation computation. In MDNT, the reputation value is computed based on the weighting factor for the first, second and third opinions of third-party agents, the reputation opinion given by a recommender agent, the credibility of the recommender agent, and the time weighing factor that represents the importance of the opinion depending on the time of the last interaction between the recommender agent and the provider agent. In PeerTrust [59], different factors are taken into account such as transaction and community factors. In DCRC, reputation scores are computed based on Query-response Credit, the Upload Credit, the Download Debit and the Sharing Credit. In BinaryTrust, reputation is computed based on the number of negative complaints.

2. Probabilistic approach:
    a. Bayesian Networks: The Bayesian approach uses a probabilistic approach to the determination of the reputation. It is based on the Bayes formula. Bayesian systems take binary ratings as input, and are based on computing reputation scores by statistical updating of beta probability density functions (PDF). Several works have used this approach including [41], [58], and [62]. In Travos, reputation is computed based on Beta Probability Distribution. However, the Bayesian approach suffers from strong assumptions of independence that are made.
    b. Maximum Likelihood Estimation: MLE [20] uses a probabilistic approach to compute the reputation value based on the probability of recommender peers to provide inaccurate information. In MLE, the reputation value is the probability of a peer to cooperate and it is chosen to maximize the probability of the available ratings.
3. Fuzzy logic: Trust and reputation can be represented as linguistically fuzzy concepts. The information received from recommender peers is characterized by being imprecise and not accurately quantified. Fuzzy systems are used to deal with such situations by providing rules for reasoning with fuzzy measures. Different factors (e.g., the credibility of recommender peers) can be represented by fuzzy sets and membership functions are used. Several works have been proposed to compute reputation scores based on fuzzy logic systems including [50, 53]. Regret considers the following dimensions in reputation computation: the *individual dimension* which is the direct trust obtained by previous transactions, the *social dimension* which refers to a trust of an agent in relation with a group and finally the *ontological dimension* that includes the particularity of each agent. In FuzzyTrust, fuzzy inference is used to produce local trust values and aggregate them to global reputation values.
4. Flow models: Systems that compute trust or reputation based on transitive iteration through looped or arbitrarily long chains. As an example, EigenTrust, and FineGrainedTrust.

4.4 The Use of Reputation

4.4.1 How to Choose a Peer Based on the Reputation Value?

After computing the reputation value, the following approaches can be adopted:

- The ranked-based approach: peers are ranked according to how reliable they are likely to be. The more reliable they are, the more trusted they are, the more the requester peer is expecting to get the required file. In this approach, the peer with the highest reputation value will be selected to upload the file. Highly reputable peers will handle almost all the uploads, yielding to an increase in their reputation. However, these peers will be overwhelmed by download requests. This approach suffers from unbalanced load share among reputable peers. This approach has been adopted by EigenTrust and Travos.

- The threshold-based approach: The computed reputation scores may be compared against a trust threshold value. A random peer from the provider peers whose reputation values are greater than this trust threshold is selected. This approach spreads the load between these peers. This approach has been adopted by Nice, MDNT, MLE, and BinaryTrust.
- The Probabilistic-based approach: choose the peer that will upload the file probabilistically based on its trust value. This probability is proportional to its normalized trust value and with a probability of $p\%$ select a new comer (e.g., 10% for EigenTrust). This approach distributes better the load share among reputable peers, giving them a chance to increase their reputation in addition to increasing the reputation of new comers.

Once the requester peer finds a provider peer and it is willing to download from it the requested file, the requester peer becomes the trusting peer and the provider peer becomes the trusted peer. Both the trusting and the trusted peers will participate in a transaction.

4.4.2 How to Evaluate a Transaction After Downloading a Requested File?

It is important at this stage that a peer is able to ascertain when a transaction is successful. In case of failure, it could be due to network congestion or network failure, and sometimes it is due to the maliciousness of the transaction partner. Assigning a trust value based on the quality of the transaction is subjective. The subjectivity of trust values is inherent in most trust systems.

According to a detailed study on eBay, recommenders peers may not send negative feedbacks by fear of reprisals or may also tend to reciprocate in both a positive and a negative way. To deal with the problem of reprisals, some reputation systems provide mechanisms such as:

- A peer can have an identity as a regular peer and another identity as a recommender peer. Recommender peers will not be easily recognized.
- Recommender peers can remain anonymous as in BinaryTrust. Recommender peers fill complaints and their identity is not provided. In this case, keeping records for assessing the credibility of recommender peers may not be possible. However, the identity of the mediators storing the recommendations are known.
- The privacy of recommender peers is protected by using symmetric-key cryptographic functions as in FineGrainedTrust.

4.5 Credibility Assessment

After assessing the transaction output, it is imperative to update the credibility of recommender peers. In MDNT, the credibility of the recommender peer is computed based on the distance between the transaction result and the trust value provided by this recommender peer. Updating the credibility of recommender peers will be beneficial for the assessment of feedbacks in future transactions. In Travos, the credibility of recommender peers in terms of accuracy of the provided recommendation is updated according to the transaction output. In MDA, the feedback sent to the supernode of the trusting peer is used to check if the transaction is considered suspicious or not. The credibility of the trusting peer is updated accordingly. This credibility will be used in computing the reputation of the trusted peer.

In completely decentralized P2P systems, credibility values could be stored at the trusting peer's level for easy access in case this information is needed. In partially decentralized systems, peers' credibility could be also stored at the supernode level. MDA counts the number of suspicious transactions to measure the credibility of peers.

4.6 Incentives, Rewards and Punishment

In P2P systems, if all peers receive the same service regardless of their behavior, peers will not be motivated to strive for high reputation values since they will be always asked to upload files without receiving any special benefit or reward. This is why service differentiation is needed.

The system may reward reputable peers with increased connectivity to other peers, greater bandwidth, and/or higher priority/probability of performed requests. Rewarding these peers will give users an incentive to get a high reputation value by sharing authentic files. Peers can also be rewarded for providing feedbacks and sharing trust data with other peers. Several incentive mechanisms have been proposed to motivate peers to contribute to the system [23, 24, 27, 43–45, 60].

While good peers are rewarded for exhibiting a good behavior, malicious peers can be punished by sending them lists of peers with low reputation values to download from them. In [43], the authors introduce a reputation-based mechanism that assigns better service to higher performing peers. The reputation-based policies are classified into two dimensions: *Provider Selection* and *Contention Resolution*. In *Provider Selection* policies, a peer among peers providing a service is chosen to provide the service. Three schemes have been proposed for *Provider Selection*: *Highest Reputation* where peer with the highest reputation is selected, *Comparable Reputation* where peers can request services only from peers with reputation values comparable to their reputation and *Black List* where peers with low reputation are not providing any service. However, in *Comparable Reputation* policy uses the concept of "Layered Communities" and provides the requesting peer with a list of peers having similar reputation values. This approach will incur an important increase of malicious uploads. Indeed, if a peer receives a service from a lower reputable peer, it will most probably receive a bad service (e.g., malicious file) and hence does not help the peer in providing good service to others. *Contention Resolution* policies help in selecting a peer among all peers requesting a service from the same provider peer. Two policies are presented: *Highest Reputation* policy where the peer with the highest reputation is selected to get the service and Probabilistically Fair w.r.t Reputation policy where a peer is selected with a probability according to its reputation.

In [45], the authors analyze the effectiveness of different incentives mechanisms to motivate peers to share files. The authors present a *reputation-based peer-approved* scheme. The scheme uses a reputation mechanism based on rating peers according to the number of files they are advertising. Peers are allowed to download files only from peers with lower or equal rating. The results show that the scheme can be used to counter the selfish behavior. However, this scheme will allow malicious peers to advertise a high number of corrupted files. According to this scheme, these peers will still receive good service. Even non malicious peers may advertise a large number of non popular or useless files and still benefit from the system.

In [60], the authors propose a reputation scheme that combines trust and incentive mechanisms. The proposed scheme uses explicit and implicit evaluations such as files' vote and retention time, download volume and users' rank to construct direct trust relationships. Based on the reputations, service differentiation is used to motivate users to share, vote on files, rank users and remove fake files. However, performance evaluation is needed to assess the performance of the proposed scheme.

Since the peers differ from each other in the type of services and resources they contribute to the system, [27] proposes a Service Differentiation Protocol (SDP) for service differentiation in completely decentralized unstructured P2P networks. This protocol works as follows:

- During the *search* phase, a peer sends its reputation score along with the Query message. Each peer that receives this query extracts the reputation score and maps this value to a Level of Service (LoS). This peer will provide this LoS to the requester peer.
- During the *content download* phase, the peer requesting the file sends its reputation score to the peer uploading the requested file. This peer will send the file with a rate of transfer according to the reputation score of the requester peer.

In [38], the contribution of peers rather than the reputation of peers is used as a guideline for service differentiation. Peers' contribution is based on their availability to share files and their involvement in the system (i.e., upload of authentic files).

In [34], the authors propose a service differentiation based on the amount of services each node has provided to a P2P community. A resource distribution mechanism is proposed to increase the utility of the whole network and provide incentive for nodes to share information. A generalized mechanism that provides incentives for nodes having heterogeneous utility functions is also described.

5 Design Requirements

Reputation systems should have minimal overhead in terms of infrastructure, computation, storage and message complexity. The design of reputation management schemes must consider the followings issues [30]:

- Enforcing Local Control: as trust data can be stored by peers in the system, it is important not to allow these peers to arbitrarily manipulate trust data. Local control mechanisms are needed to protect trust data.
- Minimizing Storage Cost: this can be achieved by minimizing the trust data that need to be stored at peers level.
- Minimizing Bandwidth Cost: this can be realized by minimizing the messages exchanged between peers.
- Fault Tolerance: the topology of P2P networks is changing frequently due to the transient nature of peers that join and leave at any time. Leaving the system without any notice, may result in unavailability of trust information stored at peers level. Trust information need to be replicated to assure having the required information.
- Scalability: scalability means the ability of the reputation scheme to scale with an increase in the number of peers. This increase will yield to an increase of transactions among the peers. Peers will need to handle more trust data by collecting, storing, generating more computations and replying to queries. To investigate the scalability of a reputation system, it is needed to address the load placed on peers, especially on highly reputable, due to high demand for upload transactions and on the network as a result of exchanging messages between peers.
- Reliability: reputation systems should provide mechanisms to protect peers from malicious threats. Based on peers' reputation, peers can be identified as good or malicious. Peers need to rely on reputation systems to efficiently identify malicious peers and isolate them from the system. The more reliable reputation systems are, the more trust is given by peers to these systems.

6 Centralized Reputation Systems

6.1 e-Commerce Applications

The Internet opened up new opportunities for millions of people to interact with each other through applications such as electronic markets. However, in e-Commerce sites, customers do not have enough information about the sellers and the products/services offered. Trust systems create a platform between different parties to learn from each other and build trust [19, 47].

eBay [1] uses the feedback profile for rating sellers and computing the sellers' reputation. Buyers rate their transaction partners with a positive or negative feedback, and explain why. The reputation is computed by assigning 1 point for each positive comment and –1 point for each negative comment. The reputation score for a seller is the sum of the received ratings.

The reputation system used in eBay has contributed significantly to increase eBay's revenues. eBay recognizes that the collected ratings to compute their members' reputations are one of the company's main assets. eBay relies on a central trusted server to maintain the reputation system that has been improved recently. For members identity, eBay members use pseudonyms and personal information is used to identify members and kept confidential by the system. No special benefits are given to new users to increase their reputation.

The eBay feedback system suffers from the following issues:

- Easy to attack
- A seller may have a good reputation value by satisfying many small transactions even if he did not satisfy a transaction with a higher amount
- No mechanism is used to detect liar members that send wrong feedbacks. This task is left to users to detect such behavior.

eBay's reputation concept has been widely used to increase the trust level of online users. Similar reputation systems have been proposed in e-Commerce applications [15]:

- BizRate.com: is a shopping search engine which lists stores and products. BizRate.com has an index of over 30 million products provided by more than 40,000 stores. BizRate.com claims that they use feedbacks collected from more than on million online users each month. BizRate.com uses ShopRank which is a proprietary shopping search and rating algorithm. This algorithm weights up price, popularity and availability of products against the reputations of sellers. BizRate.com provides:
  - Rating of products: a rating scale of 1–5 stars (awful, poor, average, very good, excellent). In addition a breakdown of the overall product rating is also available (e.g., ease of use, portability, sound quality) and the reviews provided by users. Based on this information, a user can decide to buy or not the product.
  - Rating of merchants: for each product, a list of stores that sell the product, along with the price, the availability of the product and the rating of the store. This rating is based on a smiley scale (poor, satisfactory, good, outstanding) in addition to a "Customer Certified" logo for stores who satisfy specific criteria. This logo increases the users' trust in these stores. Only collected data during the last 90 days is considered for computing the merchants' reputation.
- Amazon.com: started as online bookstore and expended to sell different kind of products. Amazon.com uses ratings for products and for merchants. In merchant rating, sellers are rated based on the quality of the service provided on a scale from 1 to 5 stars in addition to comments left by buyers. In addition, the "Safe Buying Guarantee" increases the buyers confidence and encourage them to trust

the sellers. Products in Amazon.com are also rated by users. Electronic products are only rated while other products are rated and reviews are provided. Users check reviews and vote whether the reviews are helpful or not.

- Elance.com: is a web-based project marketplace that provides small businesses easy access to projects such as: web design, development, software, administrative services, writing, etc. Service providers are rated by service buyers while service providers leave only comments to service buyers.
  - Rating of service providers: each project is rated by the service buyer according to different criteria (e.g., quality of work, responsiveness, professionalism, subject matter expertise, adherence to schedule, adherence to work). The rating is based on a five-point scale from *Extremely Unsatisfied* to *Extremely Satisfied*. In addition to this rating, the service buyers can leave comments to describe the service provider's performance. For each criterion, a numerical score is computed for all projects. The reputation of a service provider is an overall numerical score based on the projects scores
  - Service buyers: Service providers can leave only comments to service buyers and no rating is provided
- MoneyControl.com: is used to track the stock market. People can read different posted opinions and rate them. These opinions inform the readers about the stock market. These opinions help the readers make informed decisions regarding the stock they want to invest in. Opinions given by users are rated on a five stars scale. The average of all the ratings provided by users for a specific opinion is its overall rating and is expressed by a number between 0 and 5. Rating of reviewers is also available. A user can track the users who always provide good opinions. These users form the trusted agents network. A reviewer's reputation is based on the number of the trusted agents who track this reviewer.
- Yahoo.com: provides a variety of products to customers. Both rating of products and merchants are available. In product rating, users can rate the products in addition to ratings provided by experts. Customers prefer more this latter rating since the expert rating is more trusted. Merchants are also rated based on the feedbacks collected from customers. Customers who made the purchase and rated the merchant are given more weight than other users.
- Alibris.com: provides information regarding sellers of books, music and movies. It offers access to more than 50 million books. Both merchants rating and product rating are available. Merchants are rated based on the number of Alibris orders delivered by the seller. The rating scale is based on 1–5 stars and new sellers belong to the 4 stars category during the first 90 days. Products are not rated based on the content but on the quality of the products since Alibris.com is used to sell used products.
- CNET.com: provides an up-to-date information regarding technology products. Ratings of products is realized by editors. These editors are experts and their opinion is more trusted by customers. Editors use the products and rate them according to different criteria (e.g., set up, design, features, performance, service and support) according to the type of the product (e.g., camera, computer). To

compute the quality of a product, a specific weight is given to each criterion. The rating generated is a numerical score between 1 (Abysmal) and 10 (Perfect). Users can also provide their opinion regarding a specific product to exchange information between each other. Ratings for merchants is also provided based on four criteria: site functionality, store standards, order fulfillment, and customer feedback.

The feedbacks received from online users represents a rich set of collected information (e.g., ratings, comments, opinions, feedbacks) that helps the business managers in discovering and identifying the needs of customers and improving the services provided to them and hence, increasing the customers' trust in the business. As a consequence, the profit generated is increased in addition to the business reputation and value. The feedbacks received strengthens the relationship between customers and the business in addition to providing to online users the information needed to reduce the risk involved in virtual environments. While in physical environments, customers can really feel the products before buying them, exchanging information by sending feedbacks allow customers make informed decisions.

6.2 P2P Systems

Although these P2P systems use a distributed overlay for search and control messages, a centralized reputation system is used.

6.2.1 Reputation Management Using DCRC and CORC

In [28, 29], the reputation system uses objective criteria to track peers' contribution in the system. Each peer stores the reputation value locally for a fast retrieval. Two mechanisms are proposed to compute the reputation:

- The Debit Credit Reputation Computation (DCRC)
- The Credit Only Reputation Computation (CORC)

By using a Reputation Computation Agent (RCA), peers' reputations are updated periodically in a secure and distributed manner. The DCRC mechanism credits peers for serving content and debits for downloading. The second mechanism (CORC) credits peers for serving content with no debits for downloading. The two mechanisms credit peers for query processing, query forwarding and for staying online. RCA is responsible for computing peers' reputation and certain amount of accuracy is lost due to the periodic update of reputation scores and the loss of data during the communication with RCA. Peers can be differentiated according to their *behaviors* and their *capabilities*. The behavior of a peer depends on its contribution to forwarding, processing requests and uploading files. The capability of a peer depends on the processing power, memory, bandwidth and storage capacity.

In the DCRC scheme, a peer's reputation score is computed using the following components:

- Query-Response Credit: peers are credited for being online and processing query-response messages.
- Upload Credit: each peer uploading a file gets a credit for serving the file and contributing to the good functioning of the system
- Download Credit: each peer downloading a file, will get a debit
- Sharing Credit: a peer gets credit for sharing hard-to-find files.

The CORC mechanism is similar to the previous scheme except that the download will not be debited. This score will only increase. In order to prevent peers from acting maliciously once they get a high CORC score, CORC time stamps the reputation scores for expiration.

Drawbacks:

- RCA is contacted periodically by peers for updating peers' reputation scores. This is potentially a central point of failure.
- RCA needs to be replicated to ensure system robustness. No schemes have been provided to explain replication mechanisms.
- Malicious downloads are not taken into consideration in peers' reputation.

In [28], two methods for tracking reputation are proposed: *Strong Reputations* and *Weak Reputations*. In *Strong Reputations*, the RCA is expected to have a copy of all the content served by peers to ensure content reliability. In addition, the RCA crawls periodically the P2P topology to maintain snapshots of the topology. This assumption is not practical in real P2P systems even with the fact that RCA is not required to serve the content. This proposed scheme for reputation tracking incur higher overheads compared to *Weak Reputations*.

6.2.2 A Fine-Grained Reputation System for Reliable Service Selection: FineGrainedTrust

In [61], the authors propose a reputation system built upon the multivariate Bayesian inference theory. Reference [61] defines the reputation of a server (i.e., file provider, peer) as the probability that he is expected to demonstrate a certain behavior, as assessed by a client based on self experiences (i.e., the output of direct transactions) and users' feedbacks.

For reputation management, a centralized server is used as:

1. Account Manager: crediting/rewarding users and maintaining social groups.
2. Query Processor: for reputation queries
3. Feedback Collector: collects feedbacks
4. Reputation Engine: computing reputation scores

FineGrainedTrust has the following characteristics:

- Reputation data: After each transaction with a server, the user will increment the corresponding Quality of Service QoS level by one. This way, each user keeps only one vector for each server he interacted with.
- Reputation data storage: QoS information is stored across system users either in a random fashion or through a distributed hash table. In this latter case, the central server let λ users store the reputation feedback to improve system tolerance in case users are not available. The larger this value, the higher fault tolerance, the larger the communication overhead and the average storage cost.
- Reputation computation: Users keep QoS experiences with servers after each transaction. They return QoS experiences after receiving a query from the centralized server. Users may inquire the centralized server about servers' reputations. Collecting all QoS information by the centralized server simplifies significantly the computation of peers' reputations. Old experiences are not as important as recent ones.

Drawbacks:
The centralized server is a single point of failure and is easy to attack (c.f. Section 3.4.3).

7 Decentralized Reputation Systems

Since decentralized P2P applications are characterized by the absence of a central authority that coordinates the reputation management. Peers store information about past experiences with other peers and may be required to get other peers' opinions for reputation computation. Several decentralized reputation management schemes have been proposed for completely decentralized systems [13, 17, 20, 32, 36, 42].

7.1 The Distributed Trust Model: DistributedTrust

Abdul-Rahman et al. [11] proposed a model for trust based on distributed recommendations. This work is one of the first works on distributed trust models and that can be used in P2P systems. The proposed approach is based on four goals:

- Decentralization: each agent is responsible for managing trust information.
- Generalization of the notion of trust by using trust categories and trust values for different levels of trust in each category.
- The use of explicit trust statements in order to reduce ambiguity.
- Recommendations are used to get trust information regarding other agents in the system.

In this trust model, a trust relationship is between two entities, is non symmetrical and is conditionally transitive. The model has two types of trust relationships:

- Direct trust relationship: when an agent has trust in another agent.
- Recommender trust relationship: when an agent trust another agent to give recommendations about another agent's trustworthiness.

In this approach, each agent stores its trust data regarding other agents. Key-based encryption is used to protect recommendation messages from malicious agents. Network traffic is generated only for recommendations in case that the agent has not a direct trust value for another agent in the system.

Drawbacks:

- Each agent needs to store all history of past experiences and received recommendations. Storing this information will provide for the user a kind of global view of the whole network, however, a large storage capacity is needed.
- Updating this information can be time consuming and difficult.
- Increase of network traffic due to message exchange between agents in order to get reputation information.

7.2 The Binary Distributed Trust Model: BinaryTrust

In [13], the trust model is based on binary trust. An agent can be trustworthy or not. Each transaction between two agents can be either performed correctly or not. When an agent cheats, this agent becomes untrustworthy and a complaint is sent to other peers. In [13], only dishonest transactions are considered, based on the fact that malicious behavior is the exception. Reputation of a peer is based on the global knowledge of the complaints. PGrid that is a data storage, is used to store complaints. This trust model works as follows:

- When a peer file a complaint about another peer, the peer will send complaints to other peers using *insert* messages
- When a peer wants to get the trustworthiness of a peer, the peer will search for the complaints on that peer and will identify the peers that store the complaints. In order to avoid additional traffic overhead, once the peer requesting the trustworthiness of another peer receives similar trust information from a certain number of peers, there is no need for further search.

Drawbacks:

- Some peers may receive complaints about themselves. Conflict of interest may occur and peers can delete this information to drop the complaints.
- No mechanism is provided to prevent from inserting arbitrary complaints about peers
- Maintenance of PGrid is required.

7.3 Reputation Management by Choosing Reputable Servents: P2PBasic and P2PEnhanced

In [17], the distributed polling algorithm *P2PRep* is used to allow a servant *p* (i.e., the resource requester) looking for a resource to ask about the reputation of offerers (i.e., resource providers) by polling peers. After receiving a response from all the resource providers available to provide peer *p* with the requested resource, peer *p* selects a set of peers from the offerers and broadcasts a message asking other peers to give their opinion about the reputation of the offerers. Two variants of the algorithm are provided: in *the basic polling*, peers send their opinion and peer *p* uses the vote to determine the best offerer. In *the enhanced polling*, the peers provide their own opinion about the reputation of the offerers in addition to their identities. This latter will be used by peer *p* to weight the vote received.

Credibility Management is considered in *the enhanced polling* algorithm. Trust data of the peers sending their opinion to the resource requester will be considered in computing the reputation of the resource providers. Trustworthy peers are given more weights than untrustworthy peers in the reputation computation. In *the basic polling* algorithm, credibility of peers is not taken into consideration.

Drawbacks:

- The proposed schemes incur considerable overhead by polling peers for their votes. The *basic polling* algorithm checks whether the voters have provided the vote by sending *TrueVote* and *TrueVoteReply*. In the *enhanced polling* algorithm, *AreYou* and *AreYouReply* messages are used to check the identity of the voters. This will further increase the network overhead.
- Each peer has to keep track of past experiences with all other peers. The reputation of the peers is used to weight their opinions.

In [16], the authors propose SupP2PRep which is a protocol for reputation management via polling in P2P networks with superpeers. In [16], when a servent is looking for a resource, it broadcasts a *Query* message and receives a list of provider peers. The requester peer polls its peers by broadcasting a message *PollRequest* requesting their opinion about the selected provider peers. These peers reply with a *PollReply*. The superpeers collects the *PollReply* messages into one message *CumulativePollReply* and then sent to the requester peer. This proposed protocol incurs additional messages overhead for direct communication between peers. No performance analysis nor performance evaluation are presented.

7.4 Reputation Management by the XRep Protocol: XRep

In this paper [18], the authors propose an approach that uses combined reputations of servents and resources. The authors describe the XRep protocol used for maintaining and exchanging reputations and its advantages against security attacks in

P2P systems. In XRep, each servent maintains information based on its own experiences on resources and servents and can share this information with other peers. Each peer maintains two experience repositories:

- A resource repository: for storing resources identifiers and a binary value to describe each resource if it is good or bad
- A servent repository: for storing the servent identifier for each peer it interacted with, in addition to the number of successful and unsuccessful downloads.

The XRep protocol is based on the following phases:

- *Resource searching* phase: search the peers that have the requested resource.
- *Resource selection and Vote polling* phase: select the resource from the received list. The requester peer broadcasts *Poll* messages to enquire about the reputation of the resource and the peers providing this resource.
- *Vote evaluation* phase: evaluate the reputation of the requested resource based on the received votes.
- *Best servent check* phase: check that the best servent provides really this resource and that the resource digest is in fact reliable.
- *Resource downloading* phase: decide from which servent to download the resource and contact this servent directly for the download operation.

Drawbacks:

- Inquiring about the reputation of resources and the resource providers incurs considerable traffic overhead.
- No performance results are provided to convince the reader that the reputation management based on both the resources and the servents outperform the schemes that are based only on the reputation of the servents.

7.5 Reputation Management Using EigenTrust

In [32], the EigenTrust algorithm assigns to each peer in the system a global trust value based on peer's history of uploads. This trust value reflects the experiences of all peers with the peer. The authors propose a distributed and secure method to compute global trust value based on power iteration.

EigenTrust has the following characteristics:

- Reputation data: local trust values that represents the number of both satisfied and unsatisfied transactions. Local trust values are normalized and are between 0 and 1.
- Reputation computation: EigenTrust is based on transitive trust. The global reputation of a peer *i* is given by the local trust values assigned to peer *i* by other peers, weighted by the global reputations of the assigning peers. The use of transitive trust leads to a system where global trust values correspond to the left principal eigenvector of a matrix of normalized local trust values. Since each peer *i*

computes and reports its own trust value, malicious peers can easily report false trust values. In secure EigenTrust, a distributed hash table is used to assign score mangers that are responsible to compute global trust values. Each score manager is responsible for a set of peers. For each peer, the score manger learns about the peers that downloaded files from this peer and gets trust assessments from them.

- Credibility mechanism: different score managers are used to compute the global trust value for a peer. A majority vote is used on the trust values to reduce the impact of malicious score managers that report false trust values.
- Malicious peers policy: EigenTrust is robust to malicious collectives of peers who know each other and try collectively to subvert the system. In secure EigenTrust, using the one-way hash function, it is not possible for a score manager to know from whom the global trust value is computed. Malicious peers can not increase the reputation of each other. In addition, peers can not know their location in the hash space, thus, peers are unable to manipulate their own trust value. The system break up malicious collective through the presence of pre-trusted peers.

Drawbacks:

- Normalizing local trust values will not make the distinction between peers who the requester peer did not interact with and peers that performed more unsatisfied transactions than satisfied ones.
- The scheme requires reputations for each provider peer to be computed *on-demand* which requires cooperation and collaboration from a large number of peers in performing computations.
- The scheme introduces additional latency and requires long periods of time to collect data and compute a global trust value for each provider peer.
- The use of a distributed hash table and score managers for each peer in the system in order to collect trust local information and compute global trust values increases significantly the communication overhead.

7.6 Limited Reputation Sharing in P2P Systems: LimitedReputation

In [36], the proposed algorithms use only limited reputation sharing between peers. Each peer records statistics and ratings regarding other peers. As the peer receives and verifies files from peers, it updates the stored data. In the proposed voting reputation system, the requester peer receives ratings from other peers and weights them accordingly to the ratings that the requester peer has for these peers to compute a reputation value. The peers can be selected from the neighbor list (Neighbor-voting) or from the friend list (Friend-voting). In the latter case, friends are chosen from peers who have proved to be reputable. Note that a peer can be reputable, but not credible. No mechanism is given to detect liar peers.

7.7 Reputation Management Using Trust and Credibility Records: CredibilityRecords

In [51], a reputation-based distributed trust system is proposed. In this system, the outcomes of past transactions are stored in trust vectors. These vectors are maintained by peers that perform the downloads. Trust vectors are constant length, binary vectors of m bits. Each bit represents the result of a transaction: 1 if successful, 0 if not. A number is associated with each vector to indicate the number of significant bits. When the requester peer receives a list of provider peers, the peer will compute the reputation values for these peers based on local information. If this information is not available, a trust query is issued in order to inquire about the reputation of the provider peers. The responses are weighted by the credibility ratings of recommender peers. The credibility vectors are similar to the trust vectors. Each bit represents the result of a previous judgment: 1 if the judgment was right, 0 if not.

7.8 Reputation Management Using CCCI Methodology: MDNT

In [15], the authors presented a conceptual framework for measurement of quality and trust. They presented quality assessment through CCCI metrics (Correlation of delivered quality against defined quality, quality Commitment to each of the defined Quality Assessment Criteria, Clarity of each criterion from both parties' views and Influence of each criterion on the overall quality assessment).

The CCCI methodology for a trustworthiness measure provides four metrics, and defines the maximum possible correlation value for a business service interaction, $Corr_{qualities}$ and the maximum possible values of $Commit_{criterionc}$, $Clear_{criterionc}$, $Inf_{criterionc}$. The ratio of the correlation value and the maximum possible correlation value determines the relative correlation value. By using this value, the trustworthiness value can be easily computed.

7.9 Cooperative Peer Groups in Nice

NICE system is a platform for implementing cooperative applications over the Internet. A cooperative application allocates a subset of its resources (e.g., processing, bandwidth and storage) to be used by peers. In Nice, the resource provider is trusted. To access a remote resource, the trustworthiness of the resource requester is inferred. A trust value represents how likely a user consider another user to be cooperative.

In a successful transaction between two users Alice and Bob where Alice consumes a set of resources from Bob, Alice signs a cookie. This cookie states that she has successfully completed a transaction with Bob. A cookie can be stored by Bob to prove his trustworthiness to others. However a negative cookie may be destroyed by Bob. In this case, Alice may store it.

In Nice, each user stores a set of signed cookies. In case Alice wants to get resources from Bob, there are two possibilities:

- Alice has cookies from Bob: Alice gives these cookies to Bob and Bob computes a trust value for Alice
- Alice has no cookies from Bob: Alice uses the cookies that she has. The users who signed these cookies are contacted to check if they have cookies from Bob. For example, Alice has a cookie from Carol, and Carol has a cookie from Bob. Alice gets a copy of the cookie that Carol has and presents the two cookies to Bob. This means that Bob trusts Carol and she trusts Alice. Based on these cookies, Bob infer a trust value for Alice. Based on this trust value, the requested resource will be granted or not to Alice.

In Nice, a query from the requester peer is forwarded to only 5 users instead of following all the paths. Each time, a user is giving a cookie to another user, a copy of its cookies are also given. The users will be selected based on the digests that show a cookie from the resource provider. A digest of negative cookies can also be sent to identify uncooperative users. A preference list is also used to keep track of trustworthy resource providers for future use.

8 Partially Decentralized Reputation Systems

While the centralized systems suffer from the single point of failure, the major challenge in completely decentralized systems is how to collect feedbacks and perform reputation computation efficiently. The high amount of traffic generated from sending a query and getting back results causes inefficiency in using network resources. Moreover, peers in completely decentralized systems have the same role although these peers have different capabilities in terms of processing power, bandwidth, memory and storage capacity, in addition to peers' uptime. Peers that have less capabilities and short uptime should not assume the same role, at least not at the same level. In contrast, the communication protocol used in partially decentralized systems reduces significantly the number of messages exchanged during the search phase. The supernode architecture offers an intermediate design, minimizes the weaknesses of both centralized and completely decentralized systems and combines the advantages of these systems. Several reputation-based systems have been proposed for completely decentralized systems. However, all proposed research works in this field have completely focused on these systems. Almost no attention was directed toward partially decentralized systems that have received a tremendous interest from the online community. KaZaA Media Desktop (KMD) a proprietary partially-decentralized P2P system has introduced a *Participation Level* for rating files and peers. This *participation level* is considered as peers' reputation in the literature. Priority is given to peers with high *participation level*, however the exact process of how this priority is given is not known. In KaZaA, malicious peers will still have a high value of *participation level* even if their participation is affecting badly other peers if they are uploading corrupted content. KaZaA has no mechanism to detect malicious peers.

8.1 BitTorrent

BitTorrent, is a widely used second generation P2P protocol that adopts the *tit-for-tat* strategy. Using this strategy, peers are able to optimize their download and upload rates. Recent studies [10, 31, 57] have shown that the *tit-for-tat* strategy does not effectively reward good peers and punish free riders. Selfish peers can get more bandwidth while honest peers can receive low download rates. In [52], trust has been incorporated to the BitTorrent protocol, however, in this work trust is defined in terms of uploads compared to the downloads not in terms of maliciousness of the provider peer.

8.2 The Inauthentic Detector Algorithm (IDA) and the Malicious Detector Algorithm (MDA)

In [38, 39], trust is addressed according to the following dimensions: (1) *Authentic Behavior*, (2) *Credibility Behavior*, and (3) *Contribution Behavior*.

- *Authentic Behavior*: represents peer's reliability in sending authentic file in terms of accuracy and technical quality. To measure the *Authentic Behavior* of peers, the *Inauthentic Detector Algorithm* is proposed to identify malicious peers and isolate them. This scheme takes into account the size of the file uploaded. Malicious uploads are significantly reduced in addition to distributing the load uniformly among reputable peers.
- *Credibility Behavior*: represents peers' sincerity in providing an honest feedback. The concept of *Suspicious Transaction* is introduced and used to compute the credibility of a peer. A *Suspicious Transaction* is a transaction in which the appreciation is different from the one expected knowing the reputation of the peer uploading the requested file. The *Malicious Detector Algorithm* is proposed to identify peers that send the wrong feedback and minimize their impact. Peers' *Authentic Behavior* is computed according to the credibility of peers downloading the requested files.
- *Contribution Behavior*: represents the positive contribution of a peer to the system. Peers' contribution is based on:
  - Peers' *Availability*: being available for uploading requested files.

– Peers' *Involvement*: non-malicious uploads performed versus downloads received by a peer.

A *contribution-based service differentiation* scheme is proposed to differentiate between peers that contribute positively to the system (i.e., altruistic), and free riders and malicious peers (i.e., egoistic). This service differentiation is based on peers' contribution rather than peers' reputation.

According to [40], service differentiation is divided into two categories: *implicit* and *explicit*. The *Implicit* service differentiation results from the normal evolution of the system. *Explicit* service differentiation, is the one that results from the explicit decision of supernodes or peers. Using peer's contribution as a guideline for service differentiation will provide better service to peers that contribute positively and will reduce the level of service provided to free riders and malicious peers. Peers are also forced to continuously contribute to benefit from the services provided hence, minimizing the impact of the milking phenomenon.

9 Conclusion

Reputation-based trust management in peer-to-peer systems is an interesting research area and very challenging. Reputation is used to build trust among peers, minimize the risk involved in the transactions and increase users' confidence and satisfaction. Reputation is based on evaluating the transactions performed by peers. In this chapter, we surveyed reputation-based trust management P2P systems. We investigated the existing research work proposed to address peers' reputation. To understand better the functioning of reputation systems, we divided a typical reputation system into components. A description is given to each component to help in analyzing and summarizing the efforts of researchers in addressing peers' reputation. We also presented some of the existing centralized, completely decentralized and partially decentralized reputation management systems. The survey of different reputation systems reveals the important mechanisms used to achieve efficient reputation management.

The success of reputation-based e-Commerce applications has fostered the advancement of reputation management and its use in a wide area of applications. The study of P2P reputation management systems in this chapter exposes the different mechanisms used to solve the various challenges faced by these systems. Since reputation management is a risk assessment device, different constraints and parameters should be taken into consideration while designing a reputation system according to the risk involved in the transactions. In particular, file sharing systems are different from e-Commerce applications. The stability of these reputation systems and their robustness against malicious attacks, and security threats are of paramount importance and represent the main future research directions.

References

1. eBay Feedback. Http://www.ebay.com
2. Gnutella Protocol Specification v0.4. Http://www9.limewire.com/
3. Gnutella2 Specification. Http://www.gnutella2.com/
4. KaZaA. Http://www.kazaa.com/
5. Morpheus. Http://www.morphus.com/
6. Oxford Dictionary. Http://www.askoxford.com/
7. PPLive. Http://www.pplive.com/en/about.html
8. Skype. Http://www.skype.com/intl/fr/welcomeback/
9. UUsee. Http://newteevee.com/2007/03/07/chinas-uusee-scores-235-million/
10. Bharambe, A., Herley, C., Padmanabhan, V.: Analyzing and Improving a BitTorrent Network's Performance Mechanisms. In: IEEE 25th INFOCOM, pp. 1–12 (2006)
11. Abdul-Rahman, A., Hailes, S.: A distributed trust model. In: The 1997 workshop on New security paradigms, pp. 48–60. ACM Press (1997)
12. Abdul-Rahman, A., Hailes, S.: Supporting Trust in Virtual Communities. In: Proceedings of the 33rd Hawaii International Conference on System Sciences, p. 6007. IEEE Computer Society, Washington, DC, USA (2000)
13. Aberer, K., Despotovic, Z.: Managing Trust in a Peer-2-Peer Information System. In: 9th International Conference on Information and Knowledge Management, pp. 310–317 (2001)
14. Oram, A.: Peer-to-Peer: Harnessing the Power of Disruptive Technologies, pp. 21–37. O'Reilly Books (2001)
15. Chang, E., Dillon, T., Hussain, F.K.: Trust and Reputation for Service-Oriented Environments. Wiley (2006)
16. Chhabra, S., Damiani, E., di Vimercati, S.D.C., Paraboschi, S., Samarati, P.: A Protocol for Reputation Management in Super-Peer Networks. In: DEXA Workshops, pp. 979–983 (2004)
17. Cornelli, F., Damiani, E., di Vimercati, S.D.C., Paraboschi, S., Samarati, P.: Choosing Reputable Servents in a P2P Network. In: 11th International World Wide Web Conference, pp. 376–386 (2002)
18. Damiani, E., di Vimercati, S.D.C., Paraboschi, S., Samarati, P., Violante, F.: A reputation-based approach for choosing reliable resources in peer-to-peer networks. In: 9th ACM conference on Computer and communications security, pp. 207–216. ACM Press (2002)
19. Dellarocas, C.: The Digitization of Word of Mouth: Promise and Challenges of Online Feedback Mechanisms. Management Science **49**(10), 1407–1424 (2003)
20. Despotovic, Z., Aberer, K.: P2P reputation management: probabilistic estimation vs. social networks. Computer Networks **50**(4), 485–500 (2006)
21. Deutsh, M.: The resolution of Conflict:Constructive and Destructive. Tech. rep., New Haven, Yale University Press (1973)
22. eBay: http://www.ebay.com
23. Feldman, M., Lai, K., Stoica, I., Chuang, J.: Robust Incentive Techniques for Peer-to-Peer Networks. In: Proceedings of the 5th ACM conference on Electronic commerce, pp. 102–111. ACM, New York, NY, USA (2004)
24. Feldman, M., Papadimitriou, C., Chuang, J., Stoica, I.: Free-riding and whitewashing in Peer-to-Peer systems. In: Proceedings of the ACM SIGCOMM workshop on Practice and theory of incentives in networked systems, pp. 228–236. ACM, New York, NY, USA (2004)
25. Gambetta, D.: Can We Trust Trust? In: Trust: Making and Breaking Cooperative Relations, chap. 13, pp. 213–237. Published Online (2000)
26. Grandisan, T., Sloman, M.: A survey of Trust in Internet Applications. In: IEEE Communications Surveys, vol. 3 (2000)
27. Gupta, M., Ammar, M.: Service Differentiation in Peer-to-Peer Networks Utilizing Reputations. In: ACM Fifth International Workshop on Networked Group Communications, pp. 70–82 (2003)

28. Gupta, M., Ammar, M.H., Ahamad, M.: Trade-offs between Reliability and Overheads in Peer-to-Peer Reputation Tracking. Computer Networks **50**(4), 501–522 (2006)
29. Gupta, M., Judge, P., Ammar, M.: A Reputation System for Peer-to-Peer Networks. In: ACM 13th International Workshop on Network and Operating Systems Support for Digital Audio and Video, pp. 144–152 (2003)
30. Josang, A., Ismail, R., Boyd, C.: A Survey of Trust and Reputation Systems for Online Service Provision. Decis. Support Syst. **43**(2), 618–644 (2007)
31. Jun, S., Ahamad, M.: Incentives in BitTorrent Induce Free Riding. In: Workshop on Economics of Peer-to-Peer Systems, pp. 116–121 (2005)
32. Kamvar, S.D., Schlosser, M.T., Garcia-Molina, H.: The EigenTrust Algorithm for Reputation Management in P2P Networks. In: 12th International World Wide Web Conference, pp. 640–651 (2003)
33. Lee, S., Sherwood, R., Bhattacharjee, B.: Cooperative Peer Groups in NICE. In: IEEE Infocom, pp. 1272–1282. San Francisco, USA (2003)
34. Ma, R.T.B., Lee, S.C.M., Lui, J.C.S., Yau, D.K.Y.: Incentive and Service Differentiation in P2P Networks: a Game Theoretic Approach. IEEE/ACM Trans. Netw. **14**(5), 978–991 (2006)
35. Marsh, S.: Formalising Trust as a Computational Concept. Ph.D. thesis, University of Stirling (1994)
36. Marti, S., Garcia-Molina, H.: Limited Reputation Sharing in P2P Systems. In: ACM Conference on Electronic Commerce, pp. 91–101. New York, USA (2004)
37. Marti, S., Garcia-Molina, H.: Taxonomy of trust: categorizing P2P reputation systems. The Computer Networks Journal, Special Issue on Management in Peer-to-Peer Systems: Trust, Reputation and Security **50**(4), 472–484 (2006)
38. Mekouar, L., Iraqi, Y., Boutaba, R.: Free Riders under Control through Service Differentiation in Peer-to-Peer Systems. In: IEEE International Conference on Collaborative Computing: Networking, Applications and Worksharing (CollaborateCom 2005), pp. 10–20 (2005)
39. Mekouar, L., Iraqi, Y., Boutaba, R.: Peer-to-Peer Most Wanted: Malicious Peers. The Computer Networks Journal, Special Issue on Management in Peer-to-Peer Systems: Trust, Reputation and Security **50**(4), 545–562 (2006)
40. Mekouar, L., Iraqi, Y., Boutaba, R.: A Contribution-Based Service Differentiation Scheme for Peer-to-Peer Systems. Submitted (2008)
41. Mui, L., Mohtashemi, M., Halberstadt, A.: A Computational Model of Trust and Reputation for E-businesses. In: Proceedings of the 35th Annual Hawaii International Conference on System Sciences, pp. 2431–2439. IEEE Computer Society, Washington, DC, USA (2002)
42. Papaioannou, T.G., Stamoulis, G.D.: Effective Use of Reputation in Peer-to-Peer Environments. In: IEEE/ACM CCGrid: International Symposium on Cluster Computing and the Grid, pp. 259–268 (2004)
43. Papaioannou, T.G., Stamoulis, G.D.: Reputation-based Policies that Provide the Right Incentives in Peer-to-Peer Environments. the Computer Networks Journal: Special Issue on Management in Peer-to-Peer Systems: Trust, Reputation and Security pp. 563–578 (2006)
44. Ranganathan, K., Ripeanu, M., Sarin, A., Foster, I.: To Share or not to Share' An Analysis of Incentives to Contribute in File Sharing Environments (2003)
45. Ranganathan, K., Ripeanu, M., Sarin, A., Foster, I.: Incentive Mechanisms for Large Collaborative Resource Sharing. In: International Symposium on Cluster Computing and the Grid, pp. 1–8 (2004)
46. Ratnasamy, S.: A Scalable Content-Addressable Network. Ph.D. thesis, University ofCalifornia, Berkeley (2002)
47. Resnick, P., Kuwabara, K., Zeckhauser, R., Friedman, E.: Reputation Systems. Commun. ACM **43**(12), 45–48 (2000)
48. Ruohomaa, S., Kutvonen, L., Koutrouli, E.: Reputation Management Survey. In: Proceedings of the The Second International Conference on Availability, Reliability and Security, pp. 103–111 (2007)

49. Sabater, J., Sierra, C.: REGRET: Reputation in Gregarious Societies. In: Proceedings of the fifth international conference on Autonomous agents, pp. 194–195. ACM, New York, NY, USA (2001)
50. Sabater, J., Sierra, C.: Reputation and Social Network Analysis in Multi-agent Systems. In: Proceedings of the first international joint conference on Autonomous agents and multi-agent systems, pp. 475–482. ACM, New York, NY, USA (2002)
51. Selcuk, A., Uzun, E., Pariente, M.: A reputation-based trust management system for P2P networks. In: Fourth IEEE International Symposium on Cluster Computing and the Grid, pp. 251–258 (2004)
52. Shah, P., Paris, J.: Incorporating Trust in the BitTorrent Protocol. In: International Symposium on Performance Evaluation of Computer and Telecommunication Systems, pp. 586–593 (2007)
53. Song, S., Hwang, K., Zhou, R., Kwok, Y.: Trusted P2P Transactions with Fuzzy Reputation Aggregation. IEEE Internet Computing **9**(6), 24–34 (2005)
54. Stoica, I., Morris, R., Karger, D., Kaashoek, M.F., Balakrishnan, H.: Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications. In: Proceedings of the ACM SIGCOMM '01 Conference, pp. 149–160. San Diego, California, USA (2001)
55. Suryanarayana, G., Taylor, R.N.: A Survey of Trust Management and Resource Discovery Technologies in Peer-to-Peer Applications. Tech. rep., ISR (2004)
56. Teacy, W.T., Patel, J., Jennings, N.R., Luck, M.: TRAVOS: Trust and Reputation in the Context of Inaccurate Information Sources. Autonomous Agents and Multi-Agent Systems **12**(2), 183–198 (2006)
57. Thommes, R., Coates, M.J.: BitTorrent Faireness: Analysis and Improvements. In: Workshop of the Internet Telecommunications and signal Processing (2005)
58. Wang, Y., Vassileva, J.: Trust and Reputation Model in Peer-to-Peer Networks. In: Proceedings of the 3rd International Conference on Peer-to-Peer Computing, p. 150. IEEE Computer Society, Washington, DC, USA (2003)
59. Xiong, L., Liu, L.: PeerTrust: Supporting Reputation-Based Trust for Peer-to-Peer Electronic Communities. IEEE Transactions on Knowledge and Data Engineering **16**(7), 843–857 (2004)
60. Yang, M., Feng, Q., Dai, Y., Zhang, Z.: A Multi-dimensional Reputation System Combined with Trust and Incentive Mechanisms in P2P File Sharing Systems. In: Proceedings of the 27th International Conference on Distributed Computing Systems Workshops, p. 29 (2007)
61. Zhang, Y., Fang, Y.: A Fine-Grained Reputation System for Reliable service Selection in Peer-to-Peer Networks. Trans. IEEE **18**(8), 1134–1145 (2007)
62. Zhou, R., Hwang, K.: PowerTrust: A Robust and Scalable Reputation System for Trusted Peer-to-Peer Computing. IEEE Transactions on Parallel Distributed Systems **18**(4), 460–473 (2007)