P2P Reputation Management Using Distributed Identities and Decentralized Recommendation Chains

Prashant Dewan and Partha Dasgupta

Abstract—Peer-to-peer (P2P) networks are vulnerable to peers who cheat, propagate malicious code, leech on the network, or simply do not cooperate. The traditional security techniques developed for the centralized distributed systems like client-server networks are inappropriate for P2P networks by the virtue of their centralized nature. The absence of central authority in a P2P network poses unique challenges for reputation management in the network. These challenges include identity management of the peers, secure reputation data management, Sybil attack, and above all, availability of reputation data. In this paper, we present a cryptographic protocol for ensuring secure and timely availability of the reputation data of a peer to other peers at extremely low costs. The past behavior of the peer is encapsulated in its digital reputation, and is subsequently used to predict its future actions. As a result, a peer's reputation motivates it to cooperate and desist from malicious activities. The cryptographic protocol is coupled with self-certification and cryptographic mechanisms for identity management and countering Sybil attack. We illustrate the security and the efficiency of the system analytically and by means of simulations in a completely decentralized Gnutella-like P2P network.

Index Terms—Peer-to-peer networks, distributed systems, security, reputations, identity management.

1 Introduction

Peer-to-peer (P2P) networks are self-configuring networks with minimal or no central control. P2P networks are more vulnerable to dissemination of malicious or spurious content, malicious code, viruses, worms, and trojans than the traditional client-server networks, due to their unregulated and unmanaged nature. For example, the infamous VBS.Gnutella worm that infected the Gnutella network, stored trojans in the host machine.

The peers in the P2P network have to be discouraged from leeching on the network. It has been shown in Tragedy of Commons [1] that a system where peers work only for selfish interests while breaking the rules decays to death. Policing these networks is extremely difficult due to the decentralized and ad hoc nature of these networks. Besides, P2P networks, like the Internet, are physically spread across geographic boundaries and hence are subject to variable laws.

The traditional mechanisms for generating trust and protecting client-server networks cannot be used for pure[1] P2P networks. This is because the trusted central authority used in the traditional client-server networks is absent in P2P networks. Introduction of a central trusted authority like a Certificate Authority (CA) can reduce the difficulty of securing P2P networks. The major disadvantage of the centralized approach is, if the central authority turns malicious, the network will become vulnerable. In the absence of any central authority, repository, or global information, there is no silver bullet for securing P2P networks.

In this paper, we investigate Reputation Systems for P2P networks—a more ambitious approach to protect the P2P network without using any central component, and thereby harnessing the full benefits of the P2P network. The reputations of the peers are used to determine whether a peer is a malicious peer or a good peer. Once detected, the malicious peers are ostracized from the network as the good peers do not perform any transactions with the malicious peers. Expulsion of malicious peers from the network significantly reduces the volume of malicious activities.

All peers in the P2P network are identified by identity certificates (aka identity). The reputation of a given peer is attached to its identity. The identity certificates are generated using self-certification, and all peers maintain their own (and hence trusted) certificate authority which issues the identity certificate(s) to the peer. Each peer owns the reputation information pertaining to all its past transactions[2] with other peers in the network, and stores it locally. A two-party cryptographic protocol not only protects the reputation information from its owner, but also facilitates secure exchange of reputation information between the two peers participating in a transaction.

The experiments show that the proposed reputation infrastructure not only reduces the percentage of malicious

1. The network does not have any central server, global repository or, global information.

2. A transaction can be as simple as a transfer of a file from the source (peer) to the destination (peer) or as complex as a securities transaction.

- *P. Dewan is with Intel Corporation, 119 NE Atlantic Pl., Hillsboro, OR 97124. E-mail: prashant.dewan@gmail.com.*
- *P. Dasgupta is with the Department of Computer Science, Arizona State University, AZ.*

Manuscript received 28 Nov. 2007; revised 31 Aug. 2008; accepted 23 Dec. 2008; published online 6 Feb. 2009.
Recommended for acceptance by S. Wang.
For information on obtaining reprints of this article, please send e-mail to: tkde@computer.org, and reference IEEECS Log Number TKDE-2007-04-0162.
Digital Object Identifier no. 10.1109/TKDE.2009.45.

transactions in the network, but also generates significantly less network traffic as compared to other reputation-based security solutions for P2P networks.

The main contributions of this paper are:

1. A self-certification-based identity system protected by cryptographically blind identity mechanisms.
2. A light weight and simple reputation model.
3. An attack resistant cryptographic protocol for generation of authentic global reputation information w.r.t. a peer.

2 Related Work

2.1 Structured and Unstructured P2P Networks

P2P networks can be categorized into structured and unstructured P2P networks. The proposed system can be used on top of both structured and unstructured P2P networks. In structured networks, the location of the data is a function of data itself or its metadata. As a result, the search space is constrained by the metadata. The overlay networks like Chord [2], CAN [3], and PASTRY [4] are structured networks and as a result the search in these networks (Chord is O (log N) where N is the number of nodes in the network) is much more efficient than in purely unstructured (without any super nodes) P2P networks. Moreover, in structured networks, all the nodes know the fundamental structure of the network and hence can prune their search to the relevant nodes. The unstructured P2P networks do not have a well-known architecture. In unstructured networks, there is no relationship between the data or metadata and its location. As a result search is of the order of O(N) in these networks, where n is the number of nodes (each node will receive a query message at least once).

The reputation schemes proposed in this paper are independent of the structure of the network. It assumes that a search function is available and does not put any constraint on the implementation of the search function. As a result, the proposed scheme is equally useful in both the unstructured and the structured networks. The knowledge of the structure of the network can be used for optimizing the algorithm. We do not propose any optimizations in this paper.

2.2 Distributed Systems Security

In this section, we review some of the solutions developed for protecting the users of distributed systems using distributed CAs. This section is focused on distributed systems with one or more central components.

2.2.1 Publius

Publius [4][3] is a monolithic system comprised of a set of independently managed servers. It is censorship resistant and allows the publisher to publish anonymously. It uses cryptographic secret splitting techniques and divides the secret among a set of servers. It provides anonymity to the author of the content, as the servers hosting the content are not aware of the content or the author.

3. Publius was named after the pen name used by a group of writers for writing federalist papers in 1787-1788, and had an impact on the US constitution.

2.2.2 Groove

A commercial application, Groove [5], builds self-administering, context-sensitive, and synchronized share spaces for exchanging files in fine granularity (small chunks). It ensures the secrecy of shared spaces and authentication of the members of a group. A good description of Groove is provided in [5]. Groove uses shared spaces and assumes that the members holding rights to a given share space would (more often than not) trust each other.

2.2.3 SDSI

SDSI [6] is a Simple Distributed Security Infrastructure, simplifies the X.509 certificates design and provides the means for self-certification, local name spaces, secure formation of groups, and simple access control mechanisms. It also provides methods to incorporate global name spaces and globally trusted identities within the SDSI infrastructure. In SDSI, the "authority" is distributed among all members of the networks, which concurs with the underlying principles of P2P networks.

2.2.4 Dynamic Trust Management

Dynamic Trust Management encapsulates trust management in dynamic distributed environments, where the members of the system assume frequently changing multiple roles. In addition, the members themselves are transitory. Agile Management of Dynamic Collaborations [7], a project of the Stanford University, has developed methods for identification of components, their authentication, secure group communication protocols, and dynamic trust management.

2.2.5 RBAC

Role-Based Access Control was introduced in 1992 by Ferraiolo and Kuhn [8]. RBAC associates permissions with roles and not with users. As a result, a given user needs to have the credentials of the role to be able to obtain the authorization associated with the role. The users are assigned roles in a many-to-many relationship, i.e., one user can have many roles and vice versa. Currently, there are four models of RBAC: core, hierarchical, constrained, and unified.

2.2.6 Cryptographic Blinding

Blinding was introduced by Chaum in 1983 [9]. Cryptographic blinding enables an authority to digitally sign a document without seeing the content of the document.

2.2.7 COCA

COCA [10] uses a set of distributed CA servers and provides fault tolerance and mitigation against denial of service attacks. COCA puts lose constraints on the communication channels among the servers and between the client and the servers in order to improve deployability. It uses byzantine quorum system to achieve availability.

2.3 Reputation Systems

Reputation systems have been used both in client-server and P2P networks. The current state of art in reputation systems for P2P networks can be classified into three main categories. The first two categories consist of the reputation

models and systems developed for the P2P networks. These reputation systems exemplify the usefulness of a reputation system and other related reputation metrics [11], for mitigation of the impact of malicious nodes on P2P networks. The third category consists of the reputation systems developed for client-server systems.

2.3.1 Reputation Models

Resnick et al. [12] defines the reputation system as "a system that collects, distributes, and aggregates feedback about consumer's past behavior." The authors outline the problems in eliciting, distributing, and aggregating feedback. Resnick et al. explain the problem of *pseudospoofing* in [13]. Pseudospoofing is the use of multiple pseudonyms in a system by the same real-life entity. The disadvantage is that any entity can discard a handle or a pseudonym with which a bad reputation is associated and join the system as a new user, under a new pseudonym. This can possibly nullify the usefulness of a reputation system, which assigns reputations to handles. The authors also advocate that the newcomers should pay their dues in order to mitigate the effect of pseudospoofing. In other words, the newcomers should not only use the services of the system but should also contribute to the system as per the system guidelines.

PeerTrust [14] allocates the reputation information to a certain node on the network for storage, by using hash functions. Any peer looking for the reputation of another peer uses the search mechanism of the underlying network to search for the information. The authors of PeerTrust argue that trust models based solely on feedback from other peers in the community are ineffective and inaccurate. The authors recommend the "degree of satisfaction" of the peer from previous transactions and the number of transactions a peer performs in the system should be accounted for before calculating the reputation of the recommended peer.

Abdul-Rahman and Hailes [15] have proposed another trust model and the corresponding metrics. They argue that Bayesian probability may not be the best metric for representing degree of trust, because probability is inherently transitive while trust is not. In addition, the authors provide methods for combining recommendations and use the context of recommendations and recommender weights to evaluate the reputations from recommendations.

Aberer and Despotovic [16] have proposed completely distributed solution for trust management over the P-Grid peer-to-peer network. They store reputation data in the form of a binary search tree, over the network. Any agent looking for the recommendation data of another agent searches the P2P network and computes the reputation from the recommendations received. Chen and Singh [11] and Schein et al. [17] also provide trust models, similar to those mentioned above.

Dellarocas [18] has enumerated the design challenges in the online reporting systems. Dellarocas surveys online reputation, reporting mechanisms, and the corresponding issues. In addition, the author provides a good overview of recommendation repositories, professional rating sites, collaborative filtering systems [19], and regression approaches. Dellarocas also enumerates the attacks on reputation systems and techniques for foiling those attacks. The attacks that can be inflicted on the reputation systems are ballot stuffing and bad mouthing. In ballot stuffing, a peer receives a large number of (false) positive recommendations from its friends, to raise its own reputation. Bad mouthing implies issuing a large number of negative recommendations to a specific peer. The author advocates that the problems of negative and positive discrimination can be solved by maintaining anonymity of requesters.

2.3.2 Reputation Infrastructure

P2PRep [20] developed by the security group of Università di Milano, is a reputation-based system developed on top of Gnutella. In P2PRep, a requester finds a list of possible providers, polls the current neighbors of the tentative providers and takes votes from them on the goodness of the provider. P2PRep is a highly communication intensive system due to its statelessness, i.e., peers do not store state (reputation info) of other peers. P2PRep makes an implicit assumption that the neighbors of a node will have the reputation information of the node. This may or may not be true.

In RCert [21] reported by Ooi et al. and RCertPX reported by Liau et al., the reputation data are stored with the provider. The provider chains the recommendations together and signs it. A requester contacts the last requester of the provider in order to obtain the time stamp of the chain. The time stamp is used to validate the last recommendation in the chain. The time stamp of the last recommendation is stored by the last rater (requester). Having verified the last time stamp, the new requester revokes the time stamp of the last requester. RCertPX prevents the node from using the older copies of RCert. This system is very near to the proposed system but it is hinged on trusting the provider and assumes unique identities for peers. Liu et al. [22] have further developed the solution proposed in this paper. They also use recommendation chains as suggested in this paper but use third party witnesses who are trustworthy to both the requester and the provider.

In [23], Zhou et al. propose a bloom filter approach for fast reputation aggregation. They use gossip algorithms to disseminate reputation information in a network. The Gossip-based protocol is designed to tolerate dynamic peer joining and departure, as well as to avoid possible peer collusions.

2.3.3 Client-Server (Centralized) Reputation Systems

In the reputation systems based on the client-server model, the server provides pseudonyms (identities) to users and inducts them into the system. Once logged into the system, a requester (client) selects a service provider (server) (from other users) for a given service, based on the reputation of the service provider. The requester then receives a service from the provider. Once the transaction is complete, the requester gives recommendation to the server based on its satisfaction level from the transaction. Amazon, eBay, and Monster follow the client-server-based reputation system. Although the server forms a single point of failure, the users (clients) trust the server to ensure the security and integrity of the reputation data of users. Some of the other websites, which use various kinds of reputation mechanisms, are moviefinder.com, reel.com, and CDNOW.com.

Xu et al. have proposed a multilevel reputation scheme [24] which is also dependent on a central reputation computation engine. The authors attack the hard problem of load balancing in a p2p system since peers with good reputation get overloaded by requests. Their scheme assigns trust values to contents and reputation values to peers and allows peers at a certain reputation value to only access content at same or less trust value thereby diminishing the load balancing problem in P2P networks. Gupta et al. [25] talk about credit-only and credit-debit scheme. Their system contains a Reputation Computation Agent for computing and storing reputations. The reputation computation agent does form a single point of failure or attack. Piatek et al. [26] showed that even one hop bidirectional reputation propagation can improve the download times by 75 percent in a set of 100 peers where each peer follows a tit-for-tat strategy.

2.4 Identity Management

We included this section in order to summarize the identity management methods used by the current P2P networks, reputation systems, and client-server networks. In P2P networks, there is no way to ascertain the distinctness of a peer in the absence of a central agency or without using external means. This thesis has been named as the Sybil attack and proved in [27] and has been reiterated in [6].

2.4.1 P2P Networks

In Gnutella, a peer is identified by its servent id that is a function of its IP address. In the DHT-based systems, like Chord, CAN, and Pastry [28], the peers are assigned identifiers on the basis of their location in the network. All the above systems predominantly use the peer identifiers for locating content and not for evaluating reputation or for enforcing security. The identifier allocation strategies in these networks are able to meet their objectives. However, these allocation strategies are not apt for a reputation-based system (on a P2P network) because they do not restrict the number of identities a peer can possibly generate.

2.4.2 Trusted-Third-Party Systems

Identity management for systems that have at least one trusted third party has been widely researched [29], [30]. Most identity mechanisms for P2P networks depend on the trusted third party for identity management. Not much literature is available for identity management of systems where the trusted third party is absent. Aberer and coworkers have proposed an identity management system [31] developed on the top of PGrid. In [31], the peers select their own PKI-based identifiers and the network is used as a repository for mapping the identifier to the current location (IP address) of the peer.

2.4.3 PGP

PGP [32] is based on PKI and enables users to construct their own *web of trust* without using any central trusted server.[4] It is based on a P2P model, where each certificate represents a peer. It cannot be used for identity management in a reputation system as it does not restrict its users from generating large number of certificates (multiple identities).

2.4.4 IDEMIX

IBMs project Idemix [29] satisfies some of the needs of anonymity and discretionary disclosure of credentials. It is based on the principle that only the information that is absolutely necessary for the execution of a transaction should be disclosed to the other party. For example, if Alice needs a rental car and has a driver's license, then the rental company only needs to verify that Alice possesses a driver's license. The rental company does not need to know the name and address of Alice. This is true except in case that Alice gets involved in an accident with a rental car or if she tries to run away with the car. In that situation, the rental company will need additional information about Alice. Hence, a safety procedure is implemented by which the rental company acquires the name and address of Alice in certain untoward scenarios. Here, it is interesting to note that in order to impose constraints on subjects enjoying anonymous multiple identities, a backup procedure is needed in certain situations to trace those identities to the real-life subjects.

3 Reputation System

3.1 Threat Model

A Gnutella-like network, has a power-law topology and supports Insert and Search methods. The peers follow predefined Join & Leave protocols. The peers are connected with insecure communication channels. As the peers are likely to have conflicting interests, a source of motivation is needed to reduce the number of *leechers*. Leechers are peers who derive benefit from the system without contributing to the system. The rogue peers can also spread malware in the network (when other peers download content from them). Finally, peers need a mechanism to judge the quality of the content before making Go/No-Go decision in transactions and thereby develop trust relationships with other peers.

A perfect reputation system can provide the means to achieve the above goals. Any reputation system is vulnerable to ballot stuffing and bad mouthing as described in [18]. An imperfect reputation system by itself generates vulnerabilities that can be exploited by attackers. Peers need to have a unique handle to which their reputations can be attached. In the absence of any trusted central agency, an attacker can gather infinite identities and start issuing recommendations to itself. A peer might modify the reputation data stored in the network to maliciously raise its own reputation. Finally, there are other vulnerabilities that come in the picture depending on how a given reputation system is designed. We explain those vulnerabilities and their corresponding mitigation in the sections where we enumerate the design decisions.

3.2 Self-Certification

In order to participate in the reputation system, a peer needs to have a handle. The reputation of a peer is associated with its handle. This handle is commonly termed as the "identity" of the peer even though it may not

4. Although PGP servers are available now but they are more for convenience than necessity.

"identify" a peer, i.e., it may not lead to the real-life identity of the peer. A peer receives a recommendation for each transaction performed by it, and all of its recommendations are accumulated together for calculation of the reputation of a given peer.

In a centralized system, a trusted authority would have issued these identity certificates. In a decentralized reputation system, self-certification [33] splits the trusted entity among the peers and enables them to generate their own identities. Each peer runs its own CA that issues the identity certificate(s) to the peer. All the certificates used in self-certification are similar to SDSI certificates [6]. The reputation of a peer is associated with its identity and the reputation of a CA is the accumulated reputation of the identities.

Self-certification obviates the centralized trusted entity needed for issuing identities in a centralized system. Peers using self-certified identities remain pseudononymous in the system as there is no way to map the identity of a peer in the system to its real-life identity. Although anonymity or at least pseudonymity is extremely desirable in P2P networks, in a reputation system it is a double edge sword. In the absence of any mapping between multiple identities and their owner (peer), the system is vulnerable to Sybil attack or Liar farms.

A malicious peer can use self-certification to generate a large number of identities and thereby raise the reputation of one of its identities by performing false transactions with other identities. The malicious peer does not even need to collude with other distinct peers to raise its reputation, but only needs to generate a set of identities for itself. Such a large set of identities managed by one peer is called an *identity farm*. The set of identities that issue false recommendations is called a *liar farm*. This attack belongs to the class of attacks termed Sybil attacks. In simple words, a peer having an identity farm is equally capable of subverting a reputation system as a peer that has colluded with a large number of other peers.

An identity farm can be countered if, either a peer is restricted to one identity or all the identities of a peer can be mapped back to the peer. A peer can be restricted to one identity by mapping its identity to its real-life identity and thereby sacrificing anonymity, or by making the identity generation extremely resource intensive such that the peer cannot afford to generate multiple identities. Identity generation can be made resource intensive by using traditional micropayment methods, although the resource restrictions are likely to have a varied impact depending on each peer's resourcefulness.

In self-certification, we use a combination of both approaches. Each peer's CA can generate multiple identities. The recommendations received by a peer's identity from different identities of other peers, signed by the other peer's CA, are identified as signed by the same CA, and are averaged to counter the liar farms. In a transaction, the requester averages all the recommendations of the provider by CAs of the provider's past recommenders. Hence, all the past recommendations owned by the provider carry equal weight but they get averaged. Finally, it adds the averages of each CA to calculate the reputation of the provider identity. Hence, a peer cannot use its own identities (all generated by the same CA) to recommend its other identities.

A more determined malicious peer might start multiple CAs and generate multiple groups of identities. In order to counter a rogue peer having multiple CAs, the peers are divided into groups based on different criteria such that a peer cannot become a part of multiple groups. For example, a P2P network installed in a school classifies the peers by different departments, a P2P network installed in a city classifies the peers by their zip codes. Each peer obtains its group certificate from the appropriate authority and attaches it to its CA. The peer sends its blinded credentials to the group authority and the authority verifies the credentials and signs the group certificate. The authority remains stateless, i.e., it does not maintain any information to correlate a certificate with the peer.

Unlike the traditional CA or distributed CA-based approaches, grouping of peers preserves the anonymity of the peers; when combined with self-certification it curtails the possibility of a Sybil attack. In contrast to the traditional CA-based approach, neither the group authority nor the transacting peers can establish the identity of the peer. In addition, certificate revocations are not needed in the group-based approach as the group authority only vouches for the real-life existence of the peer, unlike the traditional certificate-based approaches where various certificate attributes are attested by the authority and necessitate revocation if any of those attributes mutate in time. If a highly reputed identity is compromised, its misuse would be self-destructive as its reputation will go down if misused.

The peer is denoted by P while the authority is denoted by A. Here $P \longrightarrow A : X$ denotes that the peer (P) sends a message X to the authority (A). The symbol P_{K2} represents the private key of the peer P and P_{K1} represents the public key of the peer P. $E_K(\Gamma)$ represents encryption of the phrase (Γ) with key K, while $EB_K(X)$ represents blinding phrase X with key K.

1. P$\longrightarrow$A: B1 $= \{EB_{Ka}(I_{Alice_r})\}, I_{Alice}$
   The peer Alice generates a BLINDING KEY, Ka and another identity for herself (I_{Alice_r}). Alice cannot be identified from her identity (I_{Alice_r}). Subsequently, she blinds her identity (I_{Alice_r}) with the blinding key Ka. B1 represents the blinded identity. Alice sends B1 to the authority with her real identity that proves her membership to a group.
2. A$\longrightarrow$ P: $B2 = E_{P_{Authority_{K2}}}\{B1 = EB_{Ka}(I_{Alice_r})\}$
   The authority signs the blinded identity, B1 and sends it (B2) back to the peer. The peer unblinds the signed identity and extracts the identity authorized by the authority $E_{P_{Authority_{K2}}}\{I_{Alice_r}\}$.
3. P: $E_{P_{Authority_{K2}}}\{I_{Alice_r}\} = \{EB_{Ka}\{B2\}\}$.

The fundamental assumption in the group-based approach is that in a P2P network, peers would be more interested in the ranks of the prospective providers than in the absolute value of the reputations. The simulations show that this approach changes the absolute reputations of peers considerably but it has only a minimal impact on the relative ranks of the peers. This approach is inspired from the Google page rank concept in which the pages in proximity of each other do not contribute as much to the

page rank of the target page as compared to pages at a distance [34]. The relative ranks do not stop the peers from setting thresholds. The thresholds can be based on ranks. Setting thresholds based on absolute values has very limited utility. Google uses ranks rather than the absolute numbers of links pointing to/from pages. It is well evident from the Google example that rank-based mechanisms are scalable. It can be argued that there might be some systems where absolute values are needed. This paper does not consider that case as use of absolute values needs more system context specific information that is outside the focus of this paper.

It can be argued that this approach is unfair to peers whose genuine recommendations come from peers that are a part of a large group. We agree with the argument and our experiments show that the relative ranks of the providers change only minimally. Hence, the providers are only slightly effected (Δ Mean Rank Difference ≈ 14 for varied sizes groups) by the grouping of recommendations. The requesters that give the recommendation to the providers are not affected by the grouping of recommendations.

3.3 Reputation Model

Once a peer has obtained its identity, it joins the P2P network using the standard Join method of the particular P2P network. The peer (requester) searches for one or more files using the Search method provided by the network. On the basis of the responses received, as a result of its search request, the requester generates a list of peers who have the requested file(s). The number of peers who offer a particular file is denoted by RANGE. The requester selects the peer (provider) with the highest reputation from the list and initiates the cryptographic protocol. The cryptographic protocol is presented in detail in the next section. In the protocol, the requester uses the Download method of the network, to download the file from the provider. Subsequently, it verifies the integrity, authenticity, and the quality of the file. Depending on its verification results, it sends a recommendation between MIN_RECOMMENDATION and MAX_RECOMMENDATION to the provider. The recommendations are constrained to boundaries in order to make sure that one recommendation does not completely nullify or drastically improve the reputation of a provider. Once the provider receives the recommendation, it averages the previous recommendations received by it and the recent recommendation to calculate its reputation. The above-mentioned steps are repeated for every transaction.

There is a big body of work in Decision Theory, Game Theory, and Probability [35], [36], [37] which can be used for selecting appropriate values of the above-mentioned parameters and the definition of function F() depending on the levels of the threat faced by the peers in the P2P network. In this paper, we define the function F() as the arithmetic average of the recommendations received by the provider. The proposed reputation model is independent of the topology of the P2P network, addressing schemes for its nodes, bootstrap mechanisms, joining and leaving protocols of peers, and the name service. In other words, the choice of any of these components has no impact on the reputation model and vice versa.

If the system allows the peers to issue both positive and negative recommendations to other peers, some peers might get victimized by bad mouthing, i.e., a requester can potentially issue a negative recommendation to the provider even if the provider deserved a positive recommendation for a given transaction. On the other hand, if only positive recommendations are allowed then it would be hard to differentiate a relatively new peer from a chronic bad peer. Therefore, here we make an assumption that both positive and negative recommendations are allowed and a given peer will stop interacting with peers who regularly issue negative recommendations.

3.4 Reputation Exchange Protocol

Once the requester has selected the provider with the highest reputation, it initiates the reputation exchange protocol with the provider. In the reputation exchange protocol, the requester is denoted by R while the provider is denoted by P. Here $R \longrightarrow P : X$ denotes that the requester (R) sends a message X to the provider (P). The symbol P_{K2} represents the private key of the peer P and P_{K1} represents the public key of the peer P. $E_K(\Gamma)$ represents encryption of the phrase (Γ) with key K, while $EB_K(X)$ represents blinding phrase X with key K. $H(\lambda)$ denotes a one way hash of the value λ. This protocol only assumes that *insert* & *search* functions are available and are not resilient to peers that may not follow the recommended *join* & *leave* protocol of the network. The steps in the reputation exchange protocol are as follows:

Step 1: $R \longrightarrow P$: *RTS & IDR*

The requester sends a REQUEST FOR TRANSACTION (RTS) and its own IDENTITY CERTIFICATE (IDR) to the provider. The provider needs the identity certificate of the requester as the provider has to show it to the future requesters in Step 7.

Step 2: $P \longrightarrow R$ *: IDP & TID &* $E_{P_{K2}}(H(TID \parallel RTS))$

The provider sends its own IDENTITY CERTIFICATE (IDP), the CURRENT TRANSACTION ID (TID) and the signed TID, $E_{P_{K2}}$(H(TID$\|$RTS). The signed TID is needed to ensure that the provider does not use the same transaction id again. In the end of the protocol, this signed TID is signed by the requester also and stored into the network where it will be accessible to other peers.

Step 3: $R : LTID = Max\,(Search\,(PK1 \parallel TID))$

The requester obtains the value of the LAST TRANSACTION ID (LTID) that was used by the provider, from the network. The requester concatenates the public key of the provider with the string TID and performs the search. Any peer having the TID for the provider replies back with the TID and the requester selects the highest TID out of all the TIDs received. The highest TID becomes the LTID. It is possible that the provider might collude with the peer who stores its last LTID and change the LTID. As the LTID and related information would be signed by the requester, the provider cannot play foul.

Step 4: $R : IF(LTID \geq TID)$ GO TO Step 12

If the value of the LTID found by the requester from the network is greater than or same as the TID offered by the provider, it implies that the provider has used the TID in some other transaction. Hence, it is trying to get another

recommendation for the same transaction number (TID). The requester suspects foul play and jumps to Step 12.

Step 5: $R \longrightarrow P$: *Past Recommendation Request* & r

If the check in Step 4 succeeds, i.e., the requester is sure that the provider is not using the same transaction number, it requests the provider for its previous recommendations. In other words, if the current transaction is the Nth transaction for the provider, the requester makes a request for $N-1\text{th}, \text{N}-2\text{th}$ and so on recommendations till $N-r$th recommendation where r is less than N. The value of r is decided by the requester and it is directly proportional to the requester's stake in the transaction.

Step 6: $P \longrightarrow R$: CHAIN, $E_{P_{K2}}$ (CHAIN)

$$\text{CHAIN} = (\{REC_{N-1} \| E_{Z_{N-1K2}}(H(REC_{N-1})\} \| \{REC_{N-2} \| E_{Z_{N-2K2}}(H(REC_{N-2}, REC_{N-1}))\} \| \{REC_{N-3} \| E_{Z_{N-3K2}}(H(REC_{N-3}, REC_{N-2}))\} \| \ldots . \{REC_{N-4} \| E_{Z_{N-4K2}}(H(REC_{N-r}, REC_{N-r-1}))\})$$

The provider sends its past recommendations $(REC_{N-1}, REC_{N-2} \ldots REC_{N-3})$ which were provided by peers $(Z_{N-1}, Z_{N-2}, \ldots Z_{N-3})$. The provider signs the CHAIN so that the requester can hold the provider accountable for the chain. As the recommendations have been signed by the previous requesters, the provider could not have maliciously changed them. If the requester (say Z_l) has signed both the (lth) and the previous ($l-1$th) recommendation using its private key Z_{K2}, as $E_{Zn_{K2}}(H(REC_{N-3} \| REC_{N-(l-1)}))$, there is no way a provider can modify the CHAIN. In other words, the provider cannot simply take away a bad recommendation and put in a good recommendation in order to increase its reputation.

Step 7:

$$R : Result = Verify(REC_{N-1}, REC_{N-2} \ldots REC_{N-r})$$

$If\ Result$!= $Verified$ GO TO STEP 12

The requester verifies the CHAIN by simple public key cryptography. If it has the certificates of all the peers with whom the provider has interacted in the past, the verification is simple. In the case it does not have the required certificates; it obtains the certificates from the provider itself. The provider had obtained its requester's certificate in Step 1. In addition, the requester checks for liar farms as mentioned in paragraph 2 of Section 3.2. If the verification fails the requester jumps to Step 12.

Step 8: $P \longrightarrow R : File\ or\ Service$

The provider provides the service or the file as per the requirement mentioned during the search performed for the providers.

Step 9:

$$\text{R} \longrightarrow \text{P} : \text{B1} = EB_{Ka}(\text{REC} \| \text{ TID } \| E_{R_{K2}}\{\text{H}(\text{REC}, \| \text{TID})\})$$

Once the requester has received a service, it generates a BLINDING KEY, Ka. The requester concatenates the RECOMMENDATION (REC) and the TRANSACTION ID (TID) it had received in Step 2 and signs it. Subsequently, it blinds the signed recommendation with the blinding key, Ka. The recommendation is blinded in order to make the provider commit to the recommendation received before it sees the value of the recommendation such that it does not disown the recommendation if it is low. The provider receives the blinded recommendation from the requester. The blinded recommendation is also signed by the requester. The blinded recommendation contains the Chain that the provider can subsequently use to validate its reputation to another requester.

Step 10:

a. $\text{P} \longrightarrow \text{R} : \text{B1} \| E_{P_{K2}}(\text{H}(\text{B1}), nonce), nonce$

b. $\text{R} \longrightarrow \text{P} : \text{Ka}$

The provider cannot see the recommendation but it signs the recommendation and sends the NONCE and the signed recommendation back to the requester. The requester verifies the signature and then sends the blinding key Ka to the provider which can unblind the string received in Step 10a and checks its recommendation.

Step 11: Insert

$$(\text{IDR}, \{\text{REC} \parallel \text{TID} \parallel E_{R_{K2}}\{\text{H(REC)} \parallel \text{H(TID)}\}\})$$

The requester signs: the recommendation that was given to the provider (REC), the transaction id (TID), and its own identity certificate and stores it in the network using the Insert method of the P2P network. *This completes the transaction.*

Step 12: Step 12 explains the steps a requester executes when it expects foul play:

ABORT PROTOCOL

$$R\text{: } Insert(IDR, \{\text{CHAIN} \parallel TID \parallel E_{R_{K2}}\{H(\text{CHAIN}) \| H(TID)\}\})$$

If the verification in Step 7 fails, the requester takes the CHAIN that was signed by the provider and the Transaction Id (TID), signs it and uses the INSERT method of the network to insert the chain and its own identity certificate into the network. As a result, any subsequent requester will be able to see failed verification attempt and will assume a MIN_RECOMMENDATION recommendation for that TID for the provider. The requester cannot insert fake recommendations into the network because it has to include the TID signed by the provider. If the requester reaches Step 12 from Step 4. It will request for the Chain from the Provider and subsequently will perform $R : Insert(IDR, \{C\text{HAIN} \| TID \parallel E_{N-R_{K2}}\{H(TID \parallel RTS)\}\})$.

3.5 Analysis of the Protocol

The requester needs to initiate only one search request in the network in order to collect the recommendations received by the provider in the past. This protocol handles the problem of irregular availability of the peers in the network, which is one of the major problems in P2P networks.

1. *The provider will not (intentionally) send the wrong TID in Step 2*. Let the id that the provider sends be TID′ and the last Transaction Id for the provider be LTID. The TID′ should always be equal to $\text{LTID} + 1$. If $\text{TID}' > \text{LTID} + 1$, then there will be unexplained missing recommendations. If $\text{TID}' < \text{LTID} + 1$, then the provider will be caught in Step 4 of the protocol, as the last id used by the provider was made a public information (by the previous requester) and is available to all the peers now. If a peer is taking

the role of a provider for the first time, then the TID will be 0.

2. *The provider will not abort the transaction in Step 8.* It is possible for the provider to abort the transaction after giving the requester the requested information or file in Step 8. It is also possible for requester to abort the transaction after Step 9. In both scenarios, the provider will not have a recommendation for the transaction id, TID. If the provider does not sign the blinded recommendation that the requester sent her, the requester can release the recommendation in Step 11 without obtaining provider's signature.

   In its next transaction $\mathrm{TID}+1$, the provider will not be able to show the recommendation for transaction, TID to the requester of transaction, $\mathrm{TID}+1$. Therefore, the new requester will search the network using the Search method for TID. If it finds TID, it will also find the recommendation provided to the provider in the transaction. The requester will be accountable because the TID was signed by the provider. The provider will have to accept the recommendation because it will also include the signature of the provider, TID & $E_{P_{K2}}$ (H (TID)). If the new requester does not find the recommendation, then it can believe the provider and provide him minimal recommendation for the transaction, TID.

   If the provider returns the signed blinded recommendation in Step 10, B1 & $E_{P_{K2}}$ (H (B1)), but the requester does not send the key, Ka and jumps to Step 10 without performing the intermediate steps, then the provider can search the network and get the signed recommendation of the requester. If the requester never performs Step 10, then in the next transaction, $\mathrm{TID}+1$, the new requester will search for LTID and he will not find it. Hence, the transaction TID can be considered as aborted and the next transaction can be done with transaction id, TID.
3. *Collusion by rogues or liar farms.* All reputation systems are susceptible to collusion due to the nature of a reputation system. Two or more rogues might collude in order to increase each others reputation. It has been shown in [38], [39] that when a large number of autonomous entities have to collude to do anything malicious then such an event is highly unlikely. The impact of collusion can be mitigated by categorizing recommendations by identities, by authorizing agencies, by time, etc., and identifying the outliers. The list of colluders can be published, thereby, protecting other peers from the attack. Peers have a motivation against collusion because once identified they will not be able to participate in the network. The chain of recommendations of the colluders will provide the evidence that certain peers are colluding, thereby, preventing rogue peers from incriminating good peers.
4. *Multiple requesters and concurrency*. In the current protocol, a provider will not be able to use the same identity in concurrent transactions. The first option for protocol extension is that the providers *introduces* all its requesters to each other. Subsequently, the verification in Step 4 is done among the group of requesters and the results are adjusted in order to incorporate TID difference due to multiple requesters. After incorporating the extensions, it would still be a biparty protocol where the provider is the first party and the group of requesters is the second party.

3.6 Salient Features of the Protocol

The main features of the protocol are as follows:

1. Legitimate global reputation information w.r.t. a given provider is available to all peers at one place (with the provider itself). The requester does not have to initiate multiple search requests in the network in order to collect the recommendations received by the provider in the past. It only has to issue one search request to retrieve the last transaction information of the provider and it can verify all the recommendations of the provider. This not only reduces the turnaround time of the transaction but also saves considerable volume of resources.
2. The provider is accountable for all its past transactions. It cannot maliciously meddle with its transaction history by adding or deleting any recommendation because the recommendations are chained in a sequence and signed by the past requesters. The provider cannot change any recommendation because they are digitally signed by the requesters.
3. As the global information of the provider is stored by the provider itself, this protocol is not affected by erratic availability of past recommenders or any other peer in the network. As long as the requester and the provider are connected to the network, the transaction can be completed fruitfully. Even if one of them leaves the network it can come back and complete the transaction. *In short, it handles the problem of irregular availability of the peers in the network, which is one of the major problems in P2P networks.*
4. The requester cannot (gainfully) maliciously abort the transaction in the middle. In other words, the requester cannot take the service from the provider and then logoff without giving a recommendation to the provider. If it does so it would be equivalent of giving MAX_RECOMMENDATION to the provider. The provider can take advantage of this by generating multiple premature disconnections. This attack does not work because then the future requester will find the signed TID which will counter the provider's claim that it was not issued a recommendation. The requester cannot harm the provider's reputation by logging off the network abruptly and cannot get any benefit for itself by this action.
5. This protocol cannot stop a requester from giving a "bad" recommendation to the provider even if the latter provides a legitimate file. This protocol does not stop bad mouthing nor does it prevent ballot stuffing [19]. The proposed scheme necessitates

collusion among a large number of peers for ballot stuffing or bad mouthing to work, and hence it is unlikely to be successful in the proposed system although it is not impossible.

4 Experiments and Results

4.1 Evaluation of Self-Certification

We simulated a peer-to-peer network to understand the effect of the Network Size (N), the Number of Transactions (T), and the Group Size (d) on the Mean Rank Difference (M) over all the peers in the network. The Mean Rank Difference (M) is calculated by averaging the rank difference of all the peers in the network for one instance of the simulation. The rank difference for a peer is the difference in the rank of the peer when the proposed identity mechanisms are used, and when it is not used. Specifically, we tried to answer the following two questions:

1. Is the mean rank difference a good predictor of the rank difference of individual nodes? What is the variation in the rank difference of individual peers in the simulation for a given value of d? Mathematically, what percentage of nodes has their rank difference equal to the mean rank difference?
2. Does the network size (N), group size (d), or the number of transactions (T) in the network impact the mean rank difference in the network? In other words, what is the expected mean rank difference for other network configurations which are different (in terms of size, group size d, or number of transactions) than the networks simulated by us?

The simulated network consisted of 1,000 peers in which the peers performed 20,000 transactions per instance simulation, at a group size $d = 3$. The peers were assigned unique IP addresses. Additionally, each peer was assigned a goodness factor to account for the fact that a good peer is likely to participate in higher number of transactions (as a provider) than a bad peer. Hence, the reputation of a good peer is likely to escalate faster than the reputation of a bad peer. $\mathrm{MAX\_RECOMMENDATION}$ was set to 1 while the value of $\mathrm{MIN\_RECOMMENDATION}$ was set to -2. This was done to ensure that a peer lost more reputation on performing a malicious transaction as compared to the reputation gained by doing a good transaction. In addition, the results did not vary much when the values of $\mathrm{MAX\_RECOMMENDATION}$ and $\mathrm{MIN\_RECOMMENDATION}$ were selected to be outside the set $[-2, 1]$. The percentage of malicious peers was varied from 10 percent to 90 percent. The probability that a peer would cheat was set to 1/2 in order to account for the fact that in the real world honesty is not constant and varies with time and stakes.

For each iteration of the simulation, a randomly selected peer became the provider and another randomly selected peer assumed the role of a requester. After the transaction, the requester gave a recommendation to the provider. For each recommendation received by the provider, its reputation was incremented by its goodness factor. After 20,000 transactions, the ranks of the peers were calculated without using the proposed identity management mechanism ($d = 3$). The differences of the ranks were averaged for all peers in the network and the results were statistically analyzed.

The above steps were repeated 20 times and results averaged, for each of the following three scenarios:

1. The group size was varied from 5 to 50 (Step[5] 5) and the other parameters were constant. This case enabled us to investigate the impact of averaging on mean ranks of peers,
2. The group size and the network size were kept constant and varied the number of transactions from 2,000 to 20,000 (step 2,000), and
3. The number of transactions and the group size were kept constant to 20,000 and 10, respectively, and the network size was varied from 200 to 1,000 peers (step 200).

Finally, we calculated the mean rank difference for each set of 20 simulations and statistically analyzed using regression analysis, T-Test, and Analysis of Variance test (ANOVA) [40] to deduce the answers for the above two questions.

4.2 Self-Certification Results and Analysis

In the first experiment ($\mathrm{N} = 1{,}000$, $\mathrm{T} = 20$ K and $\mathrm{d} = 3$), the rank difference averaged across the peers was 13.246 ± 0.81 with a 95 percent confidence level. Although the result of the first experiment (Fig. 1) does not resemble a normal distribution, we decided to treat it as normally distributed data as per the recommendations of the central limit theorem [41] for a sample size larger than 30. We inferred that 68 percent of the nodes in a P2P network will have a rank difference in the range of 13.246 ± 13.07. In other words, 680 nodes for every 1,000 nodes will have a rank difference of less than 27. Besides the standard error for this analysis was very low (≈ 0.4), hence the Mean Rank Difference gives a reasonably good picture of the rank differences of the individual nodes.

In order to answer the second question, we did an incremental regression analysis and an ANOVA test to ascertain how do the variation in network size, number of transactions, and group size change the mean rank difference. Heuristically, we expected that the mean rank difference should change with a change in the group size but it should remain invariant when either of the other two parameters is varied. Our null hypothesis (H_0) was that none of the three factors had any impact on the mean rank difference. The following regression equation quantifies the impact of the variability of the three factors on mean rank difference:

$$\text{Mean Rank Difference} = \Delta + \beta_1 * d + \beta_2 * N + \beta_3 * T.$$

The Significance F value for the ANOVA test was less than zero. Hence, the null hypothesis (H_0) was proved to be untrue and had to be rejected. In other words, the group size, network size, and transactions had an effect on the mean rank difference and the coefficients, β_1, β_2, and β_3 could not be zero. This inference was substantiated by the fact that the p values for the coefficients, β_1, β_2, and β_3 were extremely low ($\ll 10^{-80}$). Hence, all the three parameters had an impact on the value of the mean rank difference. This result was contrary to our heuristic.

5. Step X: The parameter is incremented by X.

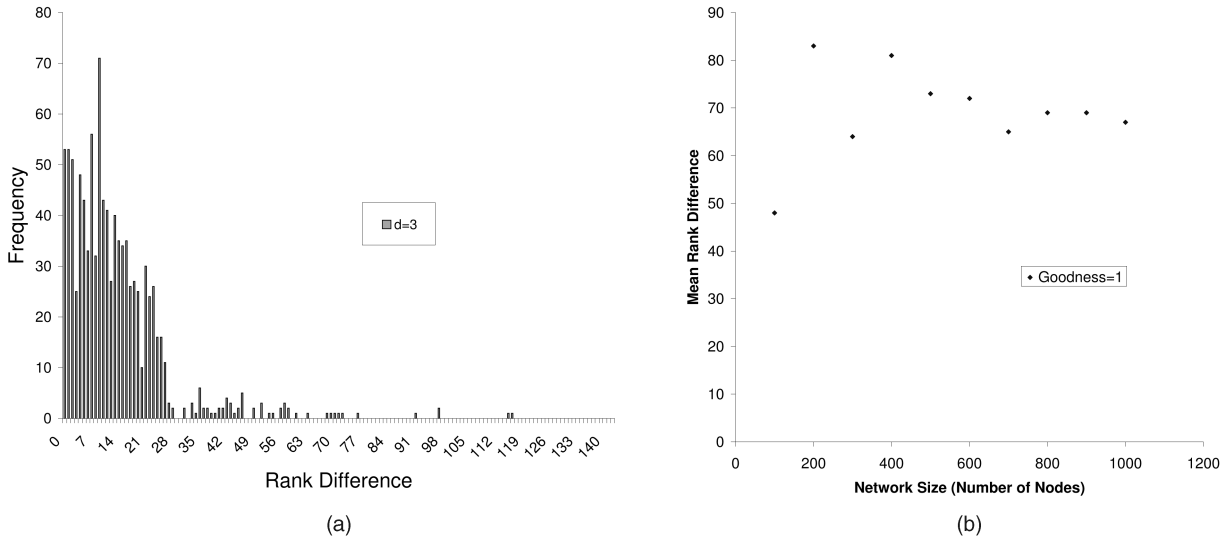


Fig. 1. Variation in mean rank difference. (a) $d = 3$. (b) $\mathrm{Goodness} = 1$.

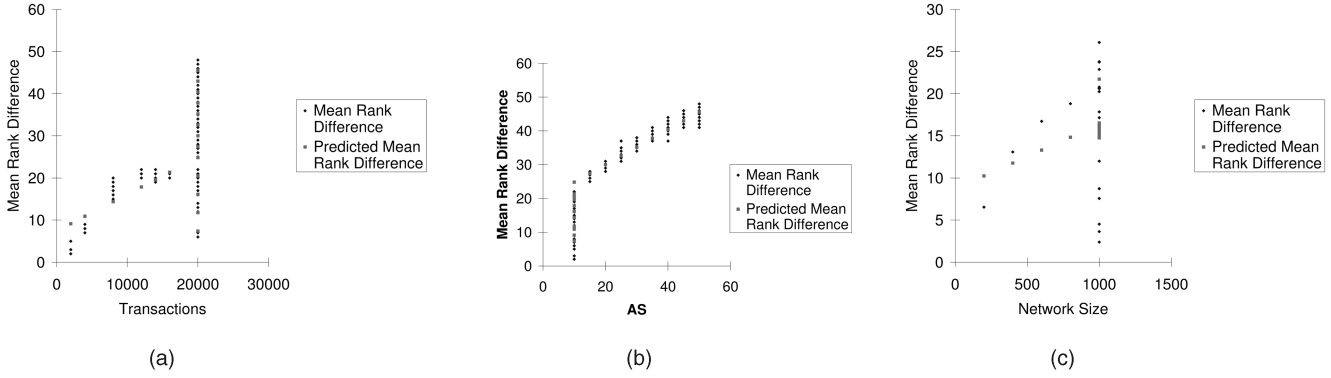


Fig. 2. Variation in mean rank difference (line fit plots). (a) Transactions $d = 3$. (b) Group size. (c) Network size.

Interestingly, the values of the coefficients, $\beta_1 = 0.52 \pm 0.02, \beta_2 = 0.02, \beta_3 = 0.0008$, and $\Delta = -19.5 \pm 0.98$ at the 95 percent confidence level, were in partial conformance to our heuristic. The value of β_1 (Fig. 2) was the highest among the three coefficients; hence, as expected, a small variation in the group size had the highest impact among the three factors on the mean rank difference. The increase in the number of transactions in the network had a minimal impact ($\beta_2 = 0.02$) on the mean rank difference (Fig. 2). This is ascribed to the fact that with every additional transaction, with a peer in a new[6] security zone, the probability of the fact that the next transaction of the provider will again be with a peer in a newer security zone becomes lesser and lesser. In addition, with every transaction of the provider with a peer in the old security zone, the mean rank difference increased. Hence, the reputation calculated with the proposed identity mechanism did not increase while the information calculated without it increased by a very small factor. Therefore, the mean rank difference increased.

The value of $\beta_2 = 0.02$ was against our initial hypothesis that the network size should not change the mean rank difference (Fig. 2). In order to find out why the network size was changing the mean rank difference, we had to perform an additional experiment. In this experiment, we made an assumption that irrespective of the reputation, the probability of a node being involved in a transaction is the same. In other words, the goodness factor of all the nodes was set to 1 and a bad node and a good node had an equal likelihood of participating in a given transaction. This was different from the assumption made in the earlier experiments where a good node had a higher goodness factor, thereby, accounting for involvement in a larger number of transactions. The result of this experiment showed that there was little or no variation in the mean rank difference w.r.t. the network size, when the likelihood of participation of all the nodes was equal. The correlation coefficient (r) was 0.09247 as is visible in Fig. 1. Hence, we inferred that in the first set of experiments, the increase in the network size raised the number of highly reputed nodes or the nodes with a high value of goodness factor. The reputation difference between the instances, when the proposed identity mechanism was used and when it was not used, was higher for the higher ranked (highly reputed) nodes as compared to the lower ranked nodes; hence, the change in the rank for higher rank nodes was larger. As a result, the mean rank difference for the network was higher.

4.3 Evaluation of the Cryptographic Protocol and the Reputation Model

The protocol was evaluated with a simulating 5,000 peers participating in 20,000-140,000 transactions. The RANGE

6. None of the peers belonging to that security zone has been a requester for this provider.

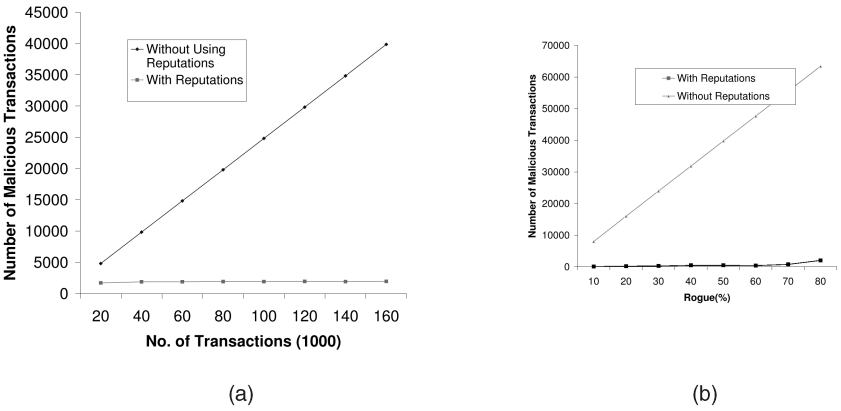


Fig. 3. Variation in total number of malicious transactions. (a) When reputations are used $d = 3$. (b) Variations in number of rogues.

variable was set to 10. In other words, it was assumed that each file was available with 10 possible providers. The requesters did not have any a priori knowledge of rogue nodes. The rogue nodes performed maliciously with a probability of 1/2, i.e., rogues cheated in one out of every two transactions.[7] Each simulation was executed five times and the results were averaged. The number of repetitions was set to 5 because after the fifth iteration of the simulation, the average values were more or less constant.

4.3.1 Cumulative and Individual Benefits of Using Reputation

We wanted to quantify the holistic and individualistic benefits of using the proposed reputation model for a P2P network. We measured the change in the number of malicious transactions with an increase in the total number of transactions in the network. The number of rogues was set to a constant at 50 percent and the number of transactions was incrementally raised from 20,000 to 140,000. As is visible in Fig. 3, the total number of malicious transactions increased considerably with an increase in the number of transactions when the proposed model was not used but are more or less constant when the proposed model was used. In the presence of an increasing number of rogues (10-90 percent), when the total number of transactions is constant ($= 140{,}000$), the rate of increase in the number of malicious transactions was much less when reputations were used (Fig. 3).

Subsequently, we analyzed the experience of each peer when the reputations are not used as compared to when they are used. As visible in Fig. 4, when the reputations were not used, the mean of the number of malicious transactions experienced by each good node was 7.966 ± 5.52 with a 95 percent confidence. This mean drastically reduced, when the reputation model is used, to 0.4 ± 1.2 with a 95 percent confidence. From the first three simulations, we concluded that the proposed model reduces the number of malicious transaction from the perspective of the network and from the perspective of each peer.

Lastly, the performance of the proposed system was compared with Eigen Trust reported by Kamvar et al. [42]. In order to replicate the simulations performed by Kamvar et al., we set the number of peers to 100 with 40 malicious peers and 60 good peers. The number of transactions was set to 1,500. We did not consider the other parameters mentioned by Kamvar et al. because their strategy encapsulates the search function also but the proposed system is used on top of any search function. The results have been illustrated in Fig. 5. The proposed system is more effective in reducing the number of "inauthentic downloads" or "malicious transactions." In a network where 60 percent nodes are malicious, the proposed system reduces the number of malicious transactions from 15 percent (in Eigen Trust) to 2 percent. Fig. 5 also shows that with an increasing number of malicious nodes in the network, the proposed system increasingly becomes more effective.

4.4 Overall Evaluation of the System

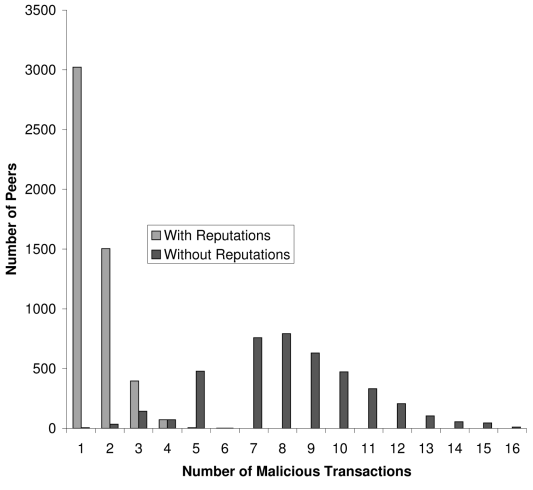
In order to evaluate the combined benefit of self-certification, the cryptographic protocol, we modified the experiments done for the evaluation of the cryptographic protocol, and added an availability factor, AF, to each node. The availability factor accounts for the erratic availability of the past recommenders of a given peer. AF values from 50 percent to 90 percent were randomly allocated to peers. The number of malicious transactions were counted and compared with the results obtained in Section 4.3. As is visible in Fig. 6, the number of malicious transactions in the system are the same when the protocol is used separately and when used along with self-certification.

5 Discussion

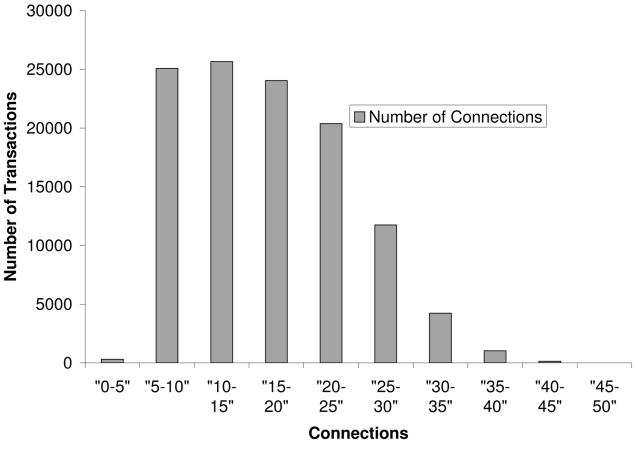
5.1 Network Traffic

We compared the cryptographic protocol with P2PRep [43]. Bandwidth consumption is the Achilles heel of any peer-to-peer network. The proposed protocol generates less network traffic than the P2PRep voting protocol. In both, P2Prep and the proposed protocol, the requester maintains a list of possible information providers when the discovery protocol finishes. P2PRep is a stateless system while the proposed system is stateful. P2PRep is highly communication intensive because of its statelessness. In phase 2 of P2PRep, the initiator polls the peers in the network for a vote on the provider. Subsequently, each peer sends a message containing a vote to the initiator. As a result, the amount of traffic

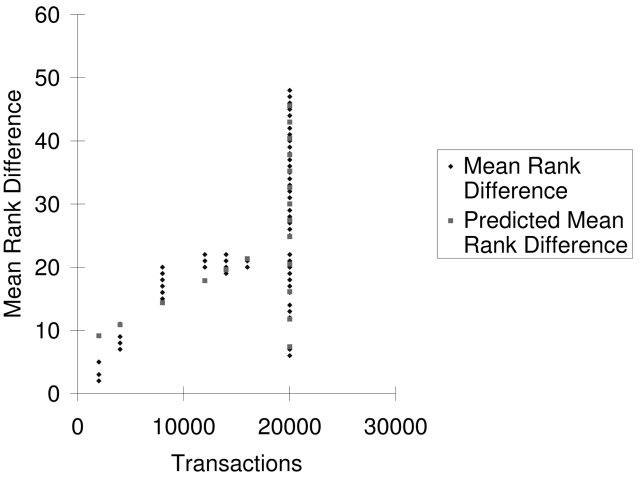
7. The probability was set to 1/2 because if a rogue cheats in all its transactions then it would be identified quickly so a good strategy for the rogue is to cheat in alternate transactions.



(a)



(b)



(c)

Fig. 4. Variation in total number of malicious transactions. (a) Number of peers versus malicious transactions $d = 3$. (b) Mean rank difference versus number of connections. (c) Mean rank difference versus number of transactions.

increases linearly as the number of peers who reply to the initiator increase. On the other hand, the proposed system uses minimal communication as the security of system lies in the reputation values stored by the provider itself.

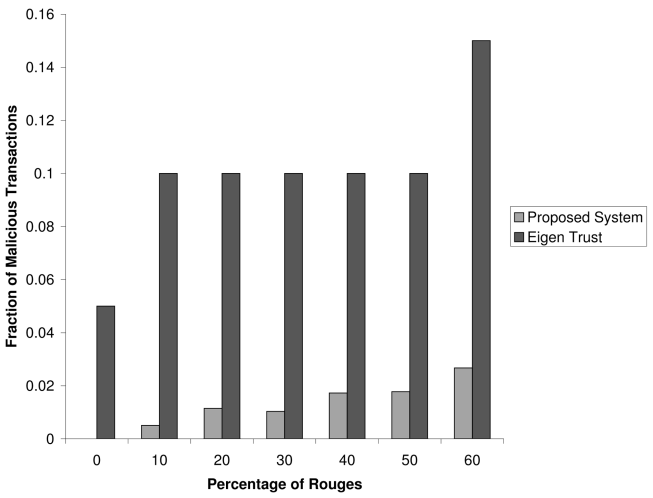


Fig. 5. Number of malicious transactions (proposed system versus Eigen Trust).

5.2 More Relevant Information

In P2PRep, a requester polls peers, which are topologically closer to the provider. This strategy may not return a high percentage of relevant recommendations. In a large network, it is highly likely that only a fraction of the peers have had any previous transaction with the possible provider. Topological proximity does not increase or decrease the probability of a transaction between two peers. In other words, the fact that two peers are neighbors does not necessitate the fact that either of the two peers can vouch for the trustworthiness of the other peer. Unlike in P2PRep, a requester in the proposed system can make sure that it retrieves the complete past history of the provider. Hence, a requester using the proposed protocol obtains a higher volume of relevant information than a requester in P2PRep.

5.3 Low Stake Fast Transaction

In the proposed protocol, it is recommended for the requester to search the network for LTID but it is not mandatory. The requester does not have to ever disclose the

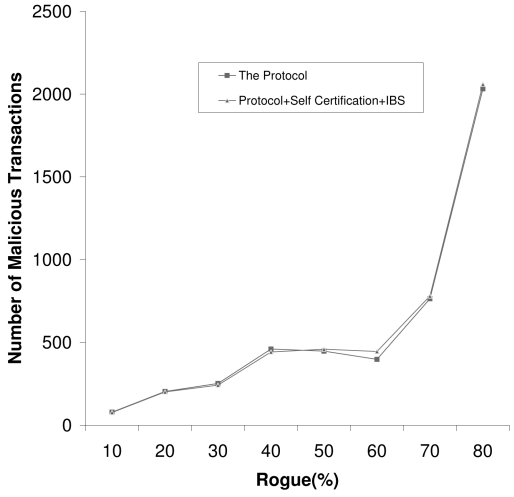


Fig. 6. Number of malicious transactions when self-certification, identity management, and the cryptographic protocol are combined.

LTID in the elicitation-storage protocol. In other words, even if the peer who wants to perform a transaction does not know the LTID, it can still perform the transaction. For example, if a peer only wants a "standards file," which is commonly available, it does not have to search for LTID and still the probability of the provider cheating is low. In P2PRep, the vote polling cannot be bypassed because the provider also receives the poll request. Hence, if the provider does not receive any poll request, it will know that the requester has not performed any security checks. Hence, the probability of the provider cheating is high.

5.4 Stale LTID

It is possible (though unlikely) that the requester might never be able to retrieve the correct LTID from the network or it retrieves a stale LTID. If the LTID retrieved by the requester is less than the one offered by the provider then the requester can simply go ahead with the transaction because then it will be the responsibility of the provider to explain the transactions of the missing LTID. If the LTID retrieved by the requester is greater than the one offered by the provider, then the provider is surely trying to cheat. The requester can abort the transaction.

6 Conclusions and Future Work

This paper presents Self-certification, an identity management mechanism, reputation model, and a cryptographic protocol that facilitates generation of global reputation data in a P2P network, in order to expedite detection of rogues. A reputation system for peer-to-peer networks can be thwarted by a consortium of malicious nodes. Such a group can maliciously raise the reputation of one or more members of the group. There is no known method to protect a reputation system against liar farms and the absence of a third trusted party makes the problem of liar farms even more difficult.

The self-certification-based identity generation mechanism reduces the threat of liar farms by binding the network identity of a peer to his real-life identity while still providing him anonymity. The Identity mechanism is based on the fundamental that the ranks of the peers are more relevant than the absolute value of their reputation. The cost of this security is the difference in the ranks of the providers because of the use of the proposed mechanism.

The global reputation data are protected against any malicious modification by the third party peer and are immune to any malicious modifications by their owner. The proposed protocol reduces the number of malicious transactions and consumes less bandwidth per transaction than the other reputation systems proposed in its category. It also handles the problem of highly erratic availability pattern of the peers in P2P networks.

Currently, the reputation of the provider is considered and the reputation of the requester is ignored. This system can be extended to encapsulate the reputations of both the provider and the requester. In addition, instead of generic number values, the reputation values can be modified in accordance with the context of the reputation.

References

[1] H. Garett, "Tragedy of Commons," *Science,* vol. 162, pp. 1243-1248, 1968.

[2] I. Stoica, R. Morris, D. Liben-Nowell, D. Karger, M.F. Kaashoek, F. Dabek, and H. Balakrishnan, "Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications," *Proc. ACM SIGCOMM,* pp. 149-160, Aug. 2002.

[3] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Schenker, "A Scalable Content-Addressable Network," *SIGCOMM Computer Comm.. Rev.,* vol. 31, no. 4, pp. 161-172, 2001.

[4] A. Rowstron and P. Druschel, "Pastry: Scalable, Decentralized Object Location, and Routing for Large-Scale Peer-to-Peer Systems," *Proc. IFIP/ACM Int'l Conf. Distributed Systems Platforms (Middleware),* pp. 329-350, Nov. 2001.

[5] G. Networks, "Groove Networks," http://www.groove.net/products/workspace/securitypdf.gtml, 2009.

[6] R.L. Rivest and B. Lampson, "SDSI: A Simple Distributed Security Infrastructure," *Proc. Crypto '96,* pp. 104-109, Aug. 1996.

[7] N. Li and J.C. Mitchell, "RT: A Role-Based Trust-Management Framework," *Proc. Third DARPA Information Survivability Conf. and Exposition (DISCEX III),* Apr. 2003.

[8] D. Ferraiolo and R. Kuhn, "Role-Based Access Controls," *Proc. 15th Nat'l Computer Security Conf.,* May 1992.

[9] D. Chaum, "Blind Signatures for Untraceable Payments," *Proc. Advances in Cryptology (Crypto '82),* 1983.

[10] L. Zhou, F. Schneider, and R. Renesse, "COCA: A Secure Distributed Online Certification Authority," *ACM Trans. Computer Systems,* vol. 20, no. 4, pp. 329-368, Nov. 2002.

[11] M. Chen and J.P. Singh, "Computing and Using Reputations for Internet ratings," *Proc. Third ACM Conf. Electronic Commerce,* pp. 154-162, 2001.

[12] P. Resnick, R. Zeckhauser, and E. Friedman, "Reputation Systems," *Comm. ACM,* vol. 43, pp. 45-48, Dec. 2000.

[13] E. Friedman and P. Resnick, "The Social Cost of Cheap Pseudonyms," *J. Economics and Management Strategy,* vol. 10, no. 2, pp. 173-199, 2001.

[14] L. Xiong and L. Liu, "PeerTrust: Supporting Reputation-Based Trust in Peer-to-Peer Communities," *IEEE Trans. Knowledge and Data Eng.,* vol. 16, no. 7, pp. 843-857, July 2004.

[15] A. Abdul-Rahman and S. Hailes, "Supporting Trust in Virtual Communities," *Proc. Hawaii Int'l Conf. System Sciences,* Jan. 2000.

[16] K. Aberer and Z. Despotovic, "Managing Trust in a Peer-2-Peer Information System," *Proc. 10th Int'l Conf. Information and Knowledge Management (CIKM '01),* pp. 310-317, Nov. 2001.

[17] A.I. Schein, A. Popescul, L.H. Ungar, and D.M. Pennock, "Methods and Metrics for Cold-Start Recommendations," *Proc. 25th Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval,* pp. 253-260, 2002.

[18] C. Dellarocas, "Immunizing Online Reputation Reporting Systems against Unfair Ratings and Discriminatory Behavior," *Proc. ACM Conf. Electronic Commerce,* pp. 150-157, Oct. 2000.

[19] C. Dellarocas, *Building Trust On-Line: The Design of Reliable Reputation Mechanism for Online Trading Communities.* MIT Sloan School of Management, 2001. Q2

[20] E. Damiani, S.D.C. di Vimercati, S. Paraboschi, and P. Samarati, "Managing and Sharing Servents' Reputations in p2p Systems," *IEEE Trans. Knowledge and Data Eng.,* vol. 15, no. 4, pp. 840-854, July 2003.

[21] B.C. Ooi, C.Y. Kiau, and K. Tan, "Managing Trust in Peer-to-Peer Systems Using Reputation-Based Techniques," *Proc. Fourth Int'l Conf. Web Age Information Management,* Aug. 2003.

[22] L. Liu, S. Zhang, K.D. Ryu, and P. Dasgupta, "R-Chain: A Self-Maintained Reputation Management System in p2p Networks," *Proc. 17th Int'l Conf. Parallel and Distributed Computing Systems (PDCS),* Nov. 2004.

[23] R. Zhou, K. Hwang, and M. Cai, "Gossiptrust for Fast Reputation Aggregation in Peer-to-Peer Networks," *IEEE Trans. Knowledge and Data Eng.,* vol. 20, no. 9, pp. 1282-1295, Aug. 2008.

[24] Z. Xu, Y. He, and L. Deng, "A Multilevel Reputation System for Peer-to-Peer Networks," *Proc. Sixth Int'l Conf. Grid and Cooperative Computing (GCC '07),* pp. 67-74, 2007.

[25] M. Gupta, P. Judge, and M. Ammar, "A Reputation System for Peer-to-Peer Networks," *Proc. 13th Int'l Workshop Network and Operating Systems Support for Digital Audio and Video (NOSSDAV),* 2003.

[26] M. Piatek, T. Isdal, A. Krishnamurthy, and T. Anderson, "One Hop Reputations for Peer to Peer File Sharing Workloads," *Proc. Fifth USENIX Symp. Networked Systems Design and Implementation (NSDI '08),* pp. 1-14, 2008.

[27] J. Douceur, "The Sybil Attack," *Proc. IPTPS '02 Workshop,* 2002.

[28] M. Castro, P. Druschel, A. Ganesh, A. Rowstron, and D.S. Wallach, "Secure Routing for Structured Peer-to-Peer Overlay Networks," *Proc. Fifth Symp. Operating Systems Design and Implementation,* pp. 299-314, Winter 2002.
[29] J. Camenisch and E.V. Herreweghen, "Design and Implementation of the Idemix Anonymous Credential System," technical report, IBM Research Division, 2002.
[30] L. Alliance, "Identity Systems and Liberty Specification Version 1.1 Interoperability," Project Report, Liberty Alliance Project, technical report, 2003.
[31] M. Hauswirth, A. Datta, and K. Aberer, "Handling Identity in Peer-to-Peer Systems," *Proc. Sixth Int'l Workshop Mobility in Databases and Distributed Systems, in Conjunction with 14th Int'l Conf. Database and Expert Systems Applications,* Sept. 2003.
[32] P. Zimmermann, *The Official PGP User's Guide.* MIT Press, 1995.
[33] P. Dewan, "Injecting Trust in Peer-to-Peer Systems," technical report, Arizona State Univ., 2002.
[34] A. Clausen, "How Much Does it Cost to Buy a Good Google Pagerank?" unpublished, Oct. 2003.
[35] G. Shafer and J. Pearl, *Readings in Uncertain Reasoning.* Morgan Kaufmann, 1990.
[36] F.K. Robert and A. Wilson, *The MIT Encyclopedia of the Cognitive Sciences (MITECS).* Bradford Books, 1999.
[37] D.P. Foster and H.P. Young, "On the Impossibility of Predicting the Behavior of Rational Agents," technical report, John Hopkins Univ., 1999.
[38] T. Rabin and M. Ben-Or, "Verifiable Secret Sharing and Multiparty Protocols with Honest Majority," *Proc. 21st Ann. ACM Symp. Theory of Computing,* pp. 73-85, 1989.
[39] C. Cachin, K. Kursawe, A. Lysyanskaya, and R. Strobl, "Asynchronous Verifiable Secret Sharing and Proactive Cryptosystems," citeseer.nj.nec.com/cachin02asynchronous.html, 2002.
[40] D.C. Montgomery, *Design and Analysis of Experiments.* J. Wiley and Sons, 2000.
[41] D. Rumsey, *Statistics for Dummies.* J. Wiley and Sons, 2003.
[42] S.D. Kamvar, M.T. Schlosser, and H. Garcia-Molina, "The Eigentrust Algorithm for Reputation Management in P2P Networks," *Proc. 12th Int'l World Wide Web Conf.,* pp. 640-651, 2003.
[43] E. Damiani, D. di Vimercati, S. Paraboschi, P. Samarati, and F. Violante, "A Reputation-Based Approach for Choosing Reliable Resources in Peer-to-Peer Networks," *Proc. Conf. Computer and Comm. Security (CCS '02).* pp. 207-216, 2002.

Q3

Prashant Dewan received the PhD degree in computer science in 2004 and the MS degree in 2002 from Arizona State University. He is a research scientist in Intel Corporation. His research interests are distributed computing, peer-to-peer networks, virtualization-based security, and reputation systems.

Partha Dasgupta is an associate professor in the Department of Computer Science at Arizona State University. He leads the Distributed Systems Lab in Arizona State University. His research interests are multicore architectures, networking, and sensor networking.

Queries to the Author

Q1. The affiliation of the author ''P. Dasgupta'' has been set as per the information provided in the biography in the manuscript. Please check it is correct.
Q2. Reference [19] has been set as a book-type reference. Please check and confirm it is correct.
Q3. Acronyms ''DEA,'' ''DEI,'' and ''DTI,'' appearing along with the authors' names have been deleted in Reference [43]. Please check.