Research Article

TrustyFeer: A Subjective Logic Trust Model for Smart City Peer-to-Peer Federated Clouds

Heba Kurdi ^{1,2}, **Bushra Alshayban**,^{1,3} **Lina Altoaimy** ⁴, and **Shada Alsalamah** ^{5,6}

¹Computer Science Department, King Saud University, Riyadh, Saudi Arabia

²Mechanical Engineering Department, Massachusetts Institute of Technology (MIT), Cambridge, MA, USA

³Technical and Vocational Training Corporation, Riyadh, Saudi Arabia

⁴Information Technology Department, King Saud University, Riyadh, Saudi Arabia

⁵Information Systems Department, King Saud University, Riyadh, Saudi Arabia

⁶Media Lab, Massachusetts Institute of Technology (MIT), Cambridge, MA, USA

Correspondence should be addressed to Heba Kurdi; hkurdi@ksu.edu.sa

Received 16 November 2017; Revised 15 January 2018; Accepted 24 January 2018; Published 25 February 2018

Academic Editor: Syed H. Ahmed

Copyright © 2018 Heba Kurdi et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Cloud computing plays a major role in smart cities development by facilitating the delivery of various services in an efficient and effective manner. In a Peer-to-Peer (P2P) federated clouds ecosystem, multiple Cloud Service Providers (CSPs) collaborate and share services among them when experiencing a shortage in certain resources. Hence, incoming service requests to this specific resource can be delegated to other members. Nevertheless, the lack of preexisting trust relationship among CSPs in this distributed environment can affect the quality of service (QoS). Therefore, a trust management system is required to assist trustworthy peers in seeking reliable communication partners. We address this challenge by proposing TrustyFeer, a trust management system that allows peers to evaluate the trustworthiness of other peers based on subjective logic opinions, formulated using peers' reputations and Service Level Agreements (SLAs). To demonstrate the utility of TrustyFeer, we evaluate the performance of our method against two long-standing trust management systems. The simulation results show that TrustyFeer is more robust in decreasing the percentage of services that do not conform to SLAs and increasing the success rate of exchanged services by good CSPs conforming to SLAs. This should provide a trustworthy federated clouds ecosystem for a better, more sustainable future.

1. Introduction

Smart cities hold the promise for economic development, organizational performance, social equity, and quality of living. This attracts the majority of the world's population to seek a sustainable future by moving to urban environments [1]. Consequently, contemporary cities are left with no option but to utilize their infrastructure and services to handle such increasing demand. City authorities need innovative solutions than can efficiently address the problems of urban development. The vision of a smart city is to deploy Information and Communications Technology (ICT) in a smart and efficient manner to enhance the quality of life for citizens and ensure access to all the services offered by the city's government [1, 2].

Cloud federation is an emerging approach in the modern cloud computing landscape that can be adopted to achieve the vision of the smart city [1]. Advances in cloud computing have made it the first choice for storing data and providing functionality, including, but not limited to, pay-per-use, on-demand services, elasticity, remote access, and the pooling of resources [3]. To ensure a high Quality of Services (QoS), services offered over the cloud are based on various Service Level Agreements (SLAs) between a Cloud Service Provider (CSP) and cloud service consumer [4]. However, it is very important for CSPs to guarantee and maintain the quality of their services at satisfactory levels, and thus, when a CSP is asked for any unavailable service, it may interact with other providers to offer this service to conform to the SLA. This is precisely the notion behind federated

clouds. The term federated cloud refers to a type of cloud computing paradigm where several independent CSPs agree to share their infrastructures/resources to accommodate a wider range of requested services with a higher QoS [5]. Each participant in the federated cloud model has access to a much larger pool of resources, and the peak-load handling capacity for each cloud increases without having to administer or maintain additional computing resources. There are two categories of federated clouds based on the communication/interaction methods: Centralized and Decentralized Peer-to-Peer (P2P) [6, 7]. On the one hand, in the centralized category, the system relies entirely on a central party that acts as a control/connection node between different clouds [8]. In the decentralized P2P category, on the other hand, each cloud is linked directly to the other clouds within the cloud ecosystem without central management [9]. Nevertheless, due to the open nature of federated cloud systems, distributed structure, and independent provisioning of resources, CSPs in these systems share resources with each other without a preexisting trust relationship. Therefore, they are vulnerable to harmful attacks by selfish and malicious CSPs.

Establishing a trust relationship between CSPs is one of the key challenges in a P2P federated cloud environment, thus leading to it becoming an emerging area of research [10]. Malicious CSPs can harm P2P cloud federation in several ways that can hinder the overall effectiveness of such federations. First and foremost, malicious CSPs can provide inauthentic services or services that do not conform to SLA. Second, they may also lie about their feedback regarding the QoS they receive from other service providers [11]. Finally, such malicious CSPs with low reputation values can create new accounts to clear their already damaged reputations and continue harming the system; this is known as whitewashing [12]. Hence, establishing a trust relationship between CSPs is one of the key challenges in a P2P federated cloud environment. According to Filali and Yagoubi [13], trust measures and metrics are two main parts that must be considered when establishing a trust management system. On the one hand, trust measures have values that are either discrete or continuous. Metrics (mathematical model), on the other hand, can be based on various approaches: simple summation, average of ratings [14], flow-based models [15, 16], fuzzy logic [17–21], probabilistic models such as Bayesian systems [22–24] or beta probability density [25, 26], or a subjective logic approach [27, 28].

Given the need to establish trust between CSPs in P2P federated clouds, we propose TrustyFeer, a trust management system that implements a trust overlay network (TON) [29] on top of the federated cloud and focuses only on the interactions between P2P CSPs. We also compare the results against two state-of-the-art reputation systems, EigenTrust [30] and TNA-SL [28]. The reason for choosing these two systems is because they are the most known trust management systems and many recent proposed systems were based on them [31–35]; in addition, they are already implemented in the simulation tool [36] that we used in our evaluation. Based on the results, we found that our proposed system achieves a better performance in varying the number of CSPs and

malicious CSPs in the system. The main contributions of this paper can be summarized as follows:

- (1) We applied the basic idea of subjective logic [27, 28] and propose new trust formulas (as described in Section 3.3) and weight trust values based on the global reputation of each CSP and the degree of its conformance to an SLA.
- (2) We built a strictly controlled evaluation framework for simulating trust relationships in P2P federated clouds.

The remainder of this paper is organized as follows. Section 2 reviews the existing work on trust management systems for both P2P networks and cloud computing. Section 3 explains the design of the proposed system, while Sections 4 and 5 describe the evaluation plan and the experimental results. A summary and concluding remarks are provided in Section 6.

2. Literature Review

This section discusses several studies in the field of trust management systems in open environments, such as e-commerce, P2P networks, web services, grid computing, and cloud computing [50, 51]. For the purpose of this review, trust management systems are classified into two categories: trust systems in P2P networks and trust systems in P2P clouds.

2.1. Trust in P2P Networks. Trust management systems in P2P environments can be divided into three main categories. The first category is peer-based trust systems, where the decision for which source to download is reached based on the honesty of peers; secondly, we have file-based trust systems, where the decision for which source to download is reached based on the quality of the provided file; the final category is hybrid-based systems, which are combinations of peer and file systems [49]. Among the different categories, peer-based systems are the most commonly employed.

EigenTrust [30] is one of the most popular peer-based algorithms in P2P networks. It lessens the impact and influence of malicious nodes within the P2P system. In EigenTrust, the global reputation of a peer in the system is calculated using the left principal eigenvector of a matrix of normalized local reputation values. In addition, an overall history of the system is available, and each peer is known and considered in the calculation of the reputation values, which is performed in a distributed and node-symmetric manner with minimal overhead on the network [31].

An enhanced EigenTrust algorithm, called HonestPeer, was proposed in [35]. In this technique, the set of honest peers with high reputation values are given greater roles in calculating the global reputations of other peers. HonestPeer dynamically selects the honest peers based on the quality of the provided files, rather than depending only on the static group of pretrusted peers, as in EigenTrust. Compared with the EigenTrust algorithm, HonestPeer has a better success rate and a minimal percentage of inauthentic downloads.

In PeerTrust [37], certain factors are considered in the trust assessment of a specific node, such as the feedback

and its scope, the credibility of the source, the context of transactions, and the community context. The metric of general trust combines the above factors and then significantly decreases ordinary threats, such as man-in-the-middle attacks, nodes in compromised bases, and tainted information being spread within the decentralized environment of the P2P. However, in this scheme, the underlying presumption is that the trust value of a peer is a measure of its reliability. Therefore, peers with higher trust values always provide more reliable feedback, but this is not always correct.

The Grid Peer [52] trust model is another upgraded version of the PeerTrust model. This model has corrected certain flaws in the PeerTrust model and has also created a place for PeerTrust within the environment of the grid. This is achieved by modifying the definition of satisfaction criteria and introducing a decay function in the algorithm. The trust factor in the satisfaction criteria is responsible for handling all the requirements that can be satisfied by the resource source. The provider of resources is chosen from a performing grid by taking into account the basic requirements that the resource consumer wants to fulfill. The addition of a decay function, which is upgraded via a feedback trust calculation algorithm, forms the basic concept of the algorithm.

In CuboidTrust [39], three trust parameters, namely, contribution, trustworthiness, and quality of resource, are used to build four relations, and the global trust value of every single peer is calculated using the power iteration. CuboidTrust provided good results and brought about a substantial reduction in the number of inauthentic resource downloads in different threat models. The parameter of trustworthiness was considered in CuboidTrust and PeerTrust, and not in EigenTrust. Therefore, both CuboidTrust and PeerTrust perform efficiently, even in presence of various malicious peers in the system.

PowerTrust [38] utilizes a TON to simulate the inter-relationships of trust present between nodes. This scheme utilizes the power-law feedback properties of P2P networks and uses random selection power nodes, which are those with the best reputations. Compared with the previously mentioned systems, PowerTrust achieved much better results in terms of the precision of global reputation and the speed of aggregation. However, PowerTrust is prone to threats by malicious pretrusted peers. This is because, in such a model, the status of completely reliable peers (pretrusted peers are considered to be in EigenTrust) is given to power nodes [53]. Therefore, there is a high probability of severe system damage if power peers turn malicious.

GossipTrust [41] is especially designed for unarranged P2P networks. The system leverages a gossip-based protocol to aggregate the score of a global reputation. Every peer randomly contacts the others and exchanges information regarding the reputation of the data in a periodic manner. The proposed gossip-based protocol is simple, does not require error recoveries, and provides controlled overheads compared with optimum deterministic protocols, which include information building of a data dissemination tree. In addition, GossipTrust provides a fast aggregation module of local trust scores, a new efficient scheme for storing

reputation information, and secure communication using identity-based cryptography.

In [40], the proposed BP/P2P system computes reputation and trustworthiness values by using a belief propagation-based distributed message passing algorithm between peers on a factor graph representation of a P2P network. Using BP/P2P, the reputations of peers are determined based on the quality of service a peer receives, and trustworthiness is determined based on the ratings provided by each peer after successful transactions. A comprehensive evaluation showed that the BP/P2P is efficient in calculating trust values, filtering malicious ratings, and reducing errors in the reputation values of peers. Moreover, compared with EigenTrust and PowerTrust, BP/P2P is more efficient in detecting and eliminating malicious nodes.

In VectorTrust [43], a Bellman–Ford-based algorithm is utilized for the quick compilation of trust scores. To analyze and compile trust values, the trust vector aggregation (TVAA) is proposed. Every single trust path in TVAA is collected, and the highest trust rating is given to a target peer. VectorTrust can be employed in decentralized and distributed networks, where no global trust data is available. As the complexity of both topology and P2P networks is increasing, VectorTrust scales effectively owing to its high speed of convergence and manageable costs.

Hierarchical fuzzy trust management (HF Trust) [42] makes use of fuzzy logic to model trusts. Every peer keeps records of all local exchanges, to determine if the peer has fulfilled the requirements or not. The factors related to trust are subjected to fuzzy inference by peers to produce a local trust index. All the data from every peer regarding local transactions is compiled by the HF Trust system, and every peer's global reputation is prepared. The seven significant parameters for evaluating trust are explained using an application that allows file sharing between peers. A considerable improvement in the performance of the P2P system has been demonstrated using this trust model, as it brings about a substantial reduction in the number of nongenuine files in the network.

The most important algorithm, on which the TrustyFeer system is based, is the trust network analysis with subjective logic (TNA-SL) algorithm [27, 28]. This algorithm is used to discover trust networks between two parties and derive trust measures from such networks. Trust in TNA-SL is stored as an opinion, and each opinion consists of four tuples (b, d, u, α) . These four tuples represent belief, disbelief, uncertainty, and a base rate, respectively, where $(b + d + u) = 1$, and α (a real number) in the range $[0 \cdots 1]$. Subjective logic offers different types of operators from binary logic and probability calculus as well as specific operators for combining and merging different opinions. This variety of operators makes it possible to support a wide range of different applications and systems. Details regarding the opinion formulation and operators are addressed in the following sections.

The previously mentioned systems are categorized as peer-based approaches. There is less research available regarding file-based approaches. The system proposed in [48] is one such system, which depends on the quality of files when calculating trust values. This system prevents inauthentic

message distributions and downloads in a structured P2P network. It depends on a distributed hash table in which there is an ID key for each file-based on its name and contents. The reputation values are stored in the file repository and the peer repository of each peer. Only trustworthy peers can evaluate the files and give reputation values. The trustworthiness of a file is based on some system parameters and equations, which are calculated after each transaction.

AuthenticPeer [49] is a hybrid-based trust management system in P2P wireless sensor networks. It uses techniques from the peer and file approaches to maximize features and overcome limitations. An evaluation of this system showed that the performance of the reputation system is enhanced compared with EigenTrust and Incremental EigenTrust.

2.2. Trust in P2P Cloud Computing. The volume of research on the P2P cloud computing environment is quite limited. The most similar system to the TrustyFeer approach is proposed in [46]. This system uses the feedback aggregation from distributed peer clouds in the system to calculate trust values. Peers are rated based on their previous compliance to SLAs with other peers. Penalties are employed to identify previous negative interactions, while rewards are used to identify previous positive interactions. However, this system only considers limited models of malicious peers and is evaluated under PeerSim, which is a P2P networks simulator.

In this model [44], there are two trust tables for each peer: a direct trust table and a recommended list. When a peer needs to calculate the trust value of another peer, it first checks its direct trust table. If a trust value exists for the peer, then it will adopt this. Otherwise, the peer checks the recommended list to find a peer that has a direct trust relationship with that specific peer. The trust value is determined based on the queries exchanged among nodes in the cloud. It also considers the following metrics: processing capacity, operating system, storage space, and links. However, this system is designed for private clouds and does not consider SLAs.

A reputation management model implemented for P2P in a cloud service provisioning environment was introduced in [45]. This model is similar to the one in [44] but makes use of Cloud Brokers (CBs). In this scheme, the CB is responsible for validating the trust based on certain parameters: storage capacity, processing capacity, links, and data cost. When one peer needs to communicate with another, it requests information from the CB or other peers. The system was validated using the CloudAnalyst tool [54]. However, in this system, the SLA measurements are also not considered.

Another trust management system in cloud computing has been designed especially for federated clouds [47]. This is a distributed framework that allows providers to determine the trustworthiness of different federated cloud computing providers. Trust is specified using personal experiences, reputation, and honesty ratings. The storage of trust values is distributed in each cloud, to allow clouds to make independent decisions regarding selection based on the trustworthiness of other clouds. Malicious peers in the system will have less ability to interfere with network operations. Therefore, providers can defend themselves against malicious ratings and satisfy the clients' QoS requirements. However, SLA

FIGURE 1: Trust overlay network over a P2P federated cloud.

measurements are not considered and information about the performance and simulation is not provided.

In contrast to the above proposed schemes, our proposed system combines the advantages of the various trust management systems. First, it uses the global reputation weight when calculating trusts, as applied in PowerTrust [38]. Second, it builds a TON on top of P2P federated clouds, as in PowerTrust [38] and VectorTrust [43]. Third, it calculates trust and reputation values based on subjective logic, similar to [28], which is more accurate owing to the different factors it considers for each score. Fourth, it introduces new formulas for calculating the subjective logic components (i.e., belief, disbelief, uncertainty, and the base rate) based on the SLA parameters. Fifth, it uses the SLA parameters to calculate the trust, which are the most important parameters for improving the performance in cloud computing environments. Table 1 summarizes the different characteristics of the previously discussed systems.

3. System Design

TrustyFeer reputation system builds a TON [29] on the top of a P2P federated cloud, as shown in Figure 1. The vertices in the network ($CSP_1, CSP_2, \dots, CSP_6$) represent the CSPs in the system, while the directed edges represent the direct recent transactions or interactions between two providers. An edge label represents the opinion of the source CSP regarding the destination CSP.

In this section, we describe the architecture and operations of the TrustyFeer system. We also discuss the computation of the local and global trust values using subjective logic [28].

3.1. System Architecture. The architecture of the TrustyFeer trust management system, illustrated in Figure 2, consists of three basic components, and each component has a different task. The basic components of the TrustyFeer trust management system are the system registry, the reputation database management system, and the provider peer. Next, we describe each component in detail.

- (1) System registry: this is an in-memory database that contains the list of CSPs in the system and the services

TABLE 1: Comparison of different trust management systems.

Type	Systems	Environment	Tool used	Trust value evaluation	Performance measured
Peer-based trust systems	EigenTrust [30]	P2P network	P2P simulator	Calculating the sum of positive and negative ratings	Number of inauthentic files
	PeerTrust [37]	Decentralized P2P network	Mathematica 4.0	Normalizing the transaction rating	Trust computation error against malicious behaviors
	PowerTrust [38]	P2P Grid computing	Discrete-event driven simulator	Bayesian theory	Reputation convergence overhead, ranking discrepancy, and aggregation errors
	HonestPeer [35]	P2P network	P2P trust simulator	Calculating the global reputation of other peers	Success rate and the percentage of inauthentic downloads
	CuboidTrust [39]	P2P network	-	Using power iteration	Number of inauthentic resource downloads under various threat models
	BP-P2P [40]	P2P network	MATLAB	Using belief propagation distributed message	Error in the reputation values of peers, the computational complexity, and the communication overhead
	GossipTrust [41]	P2P network	P2P trust simulator	Using gossip protocol	Query success rate
	HFTrust [42]	P2P network	Simulation using Java	Using fuzzy logic	Number of inauthentic files in the system
	VectorTrust [43]	P2P network	VTSim simulator using Netlogo	Using Bellman-Ford-based algorithm	Convergence speed, communication overhead, malicious peer detection rate and detection speed
	TNA-SL [28]	P2P network	-	Using subjective logic	-
	Trust model for reliable file exchange [44]	P2P private cloud	-	Calculating the sum of all metric values multiplied by a given weight	-
	Reputation-based trust model [45]	P2P private cloud	CloudAnalyst environment tool.	Using a mathematical model	Trust rate
File-based trust systems	Trust modeling [46]	P2P clouds	PeerSim simulator	Using game theory	Level of trust in terms of trusted and untrusted peers
	Service trust worthiness [47]	Intercloud computing	-	Based on quorum	-
Hybrid-based trust systems	Reputation management system [48]	Structured P2P network	-	Using file reputation and peer reputation	Degree of preventing untrustworthy files from spreading compared to existing systems
	AuthenticPeer [49]	P2P sensor network	P2P trust simulator	Using [30] for peer quality and [48] for file quality	Success rate and the failure rate of authentic files

FIGURE 2: System architecture.

FIGURE 3: Transitive trust example.

they offer. It acts as a resource management system, which maintains traces of all providers in the system and dynamically updates the current services offered by each CSP.

- (2) Reputation database management system: this stores the reputation matrix and updates the matrix after each round.
- (3) Provider peer: each provider peer consists of the following four components:
 - (a) Trust manager: this is responsible for trust calculations. It receives the services score from the service manager to compute the trust matrix and sends this to the opinions database. It contains two components: the *transitive calculator* and the *parallel calculator*.
 - (i) The *transitive trust calculator* is used to calculate the indirect trust, as illustrated in Figure 3.
 - (ii) The *parallel trust calculator* is used to calculate the parallel trust for one node, as illustrated in Figure 4.
 - (b) Communication manager: this requests the system registry to check the services of other providers and update their available services. It also requests for other providers in the system to receive services. In addition, it updates the trust matrix for the client provider. It also checks the

FIGURE 4: Parallel trust example.

reputation database to obtain the trust values of other providers. In addition, it exports the trust matrix to the reputation database.

- (c) Opinions database: this contains the matrices of opinions for all providers in the system. It also receives requests from the trust manager to store or update the matrix after each cycle.
- (d) Services manager: this contains the list of services and accepts requests for services. In addition, it rates and normalizes the received services. It also sends the scores of rated services to the trust manager.

3.2. System Operations. The TrustyFeer system consists of a number of CSPs in P2P formation. Each provider peer in the system has an associated $n \times n$ matrix of opinions, where n is the number of providers in the system. In a similar approach to [28], the trust of provider A in provider B is represented by opinions on four factors, b , d , u , and α , which represent belief, disbelief, uncertainty, and the base rate, respectively, with $(b + d + u) = 1$. Here, α is a real number in the range $[0 \cdots 1]$.

Table 2 details the calculations of opinions, as proposed in [28], based on previous interactions. The new proposed

TABLE 2: Opinion calculation [28].

Belief	(Positive/(positive + negative + 2.0))
Disbelief	(Negative/(positive + negative + 2.0))
Uncertainty	(2.0/(positive + negative + 2.0))
Base rate	1.0 if the CSP pretrusted, 0.5 otherwise.

```

(1) procedure Initialization
(2)   for each provider in the system do
(3)     believe = 0.0
(4)     disbelieve = 0.0
(5)     uncertainty = 0.0
(6)     alpha = 0.0
(7)   end for
(8) end procedure

```

ALGORITHM 1: Initialize global trust matrix.

formulas for calculating belief, disbelief, uncertainty, and base rate values are discussed in the next section.

The discount operator is used to evaluate transitive chains. For example, when CSP A wants to calculate an opinion regarding CSP C using information from the intermediate CSP B, the discount operator is used. In this case, the notation is written as

$$\omega_C^{(A:B)} = \omega_B^A \otimes \omega_C^B. \quad (1)$$

The consensus operator is used to average two opinions together. For example, when CSP A and CSP B both have opinions about CSP C, the consensus operator is used to consolidate them. The notation for this is

$$\omega_C^{A \oplus B} = \omega_C^A \oplus \omega_C^B. \quad (2)$$

Table 3 presents the calculation of the above two operators. In addition, the single trust value extracted from an opinion factor is called the expected value (E) and is calculated as in (3):

$$E = b + \alpha * u. \quad (3)$$

3.3. Trust Evaluation. At the initial stage, all CSPs in the system are initialized at the following values of opinion factors: $b = 0$, $d = 0$, $u = 1$, and $\alpha = 0.5$, as presented in Algorithm 1.

Then, after each transaction, the values b , d , u , and α of the opinion factors for the sender provider are updated using new proposed formulas (4), which consider the global reputation of the sender provider when calculating the opinion factors:

$$b_B^A = S_B^A * E$$

$$d_B^A = (1 - S_B^A) * E$$

TABLE 3: Calculation of discount and consensus operators [28].

Discount: $\otimes$	Consensus: $\oplus$
$b_c^{A:B} = b_b^A b_c^B$	$b_c^{A \oplus B} = b_c^A u_c^B + b_c^B u_c^A / \text{denominator}$
$d_c^{A:B} = b_b^A d_c^B$	$d_c^{A \oplus B} = d_c^A u_c^B + d_c^B u_c^A / \text{denominator}$
$u_c^{A:B} = d_b^A + u_b^A + b_b^A u_c^B$	$u_c^{A \oplus B} = u_c^A u_c^B / \text{denominator}$
$\alpha_c^{A:B} = \alpha_c^B$	$\alpha_c^{A \oplus B} = \alpha_c^A$
Denominator = $u_c^A + u_c^B - u_c^A u_c^B$.	

$$u_B^A = 1 - (b_B^A + d_B^A)$$

$$\alpha_B^A = E^{t-1},$$

where t represents the current transaction.

(4)

S_B^A is the score of the service given by provider A (i.e., the sender provider) to provider B (i.e., the receiver provider). Algorithm 2 shows the pseudocode for updating opinion values.

3.4. Trust Aggregation. Considering the scenario in Figure 5, where the five CSPs, A, B, C, D, and E, have opinions between them shown on edges, the procedure for determining the trust A has in E, described in Algorithm 3, is as follows:

- (1) Create a matrix, OF, of opinion factors with dimension (5×5), where each element of the matrix defines a CSP–CSP relationship. All opinions are computed based on the formulas described in Section 3.3.
- (2) Compute the square of the matrix OF using the discount and consensus operators, as presented in Table 3. The discount operator is used to calculate the trust on each path, and then the consensus operator is used to determine the consensus of these all together.
- (3) Create a different matrix OF' to store the opinions with maximum confidence resulting at each position throughout the multiplication process.
- (4) Define a global trust matrix EV(OF') that contains the expected trust values, where EV(OF', A, E) represents the trust the CSP A has in the CSP E.

4. Evaluation Methodology

The main factors that affect P2P federated clouds are the number of CSPs and the models and percentages of malicious CSPs. In order to evaluate the efficiency of our proposed system, we used the following materials:

- (i) Hardware: processor: Intel core i5; speed: 1.1 GHz; RAM: 2 GB; hard disk: 200 GB; keyboard: standard keyboard; monitor: SVGA.
- (ii) Software: to develop and test trust and reputation algorithms in dynamic environments, we used P2P trust simulator [36], which is a Java-based, well-established open source simulation framework. It consists of two main elements: the trace generator and

```

(1) procedure Update
(2)   if the client provider in a new direct transaction then
(3)     Update opinion values of server provider in the client provider's matrix
(4)      $believe = ServiceScore * ExpectedValueOfClientProvider$ 
(5)      $disbelieve = (1.0 - ServiceScore) * ExpectedValueOfClientProvider$ 
(6)      $uncertainty = 1.0 - (believe + disbelieve)$ 
(7)   end if
(8) end procedure

```

ALGORITHM 2: Update opinion values for server provider.

```

(1) procedure ComputeTrust
(2)   create matrix OF of opinion elements
(3)   for each CSPs in the system do
(4)     discount and then consensus opinions in OF
(5)   end for
(6)   for each position in matrix OF do
(7)     if the believe of the opinion in OF is max then
(8)       store the opinion with the max believe in OF'
(9)     end if
(10)  end for
(11)  export the stored value in OF' to the global trust matrix
(12)  compute opinion expected values as trust values
(13) end procedure

```

ALGORITHM 3: Compute trust value.

FIGURE 5: Example of a trust network graph of five CSPs.

the simulator. The trace generator outputs a trace file containing the initialization of the network and transactions. The simulator takes the output trace file and the reputation algorithm as input and then simulates the network run and outputs statistics on how the trust manager performed. The two benchmark systems, EigenTrust [30] and TNA-SL [28], are already implemented in the simulator and used in the evaluation of the proposed system. In addition, P2P trust simulator is used to evaluate trust and reputation systems in P2P network and clouds environments [35, 41, 46].

We considered the following performance measures:

- (1) The percentage of inauthentic services exchanged by good CSPs, which should be minimized. These are the services that do not conform to SLA measures.

- (2) The success rate for good CSPs, which describes the number of valid services received by good CSPs satisfying SLA measures divided by the number of transactions attempted by good CSPs.

Success rate

$$= \frac{\text{\# of valid services received by good providers}}{\text{\# of transactions attempted by good providers}} \quad (5)$$

The hypothesis is that using TrustyFeer in a P2P federated cloud will increase the success rate of services exchanged by good CSPs that conform to SLAs and decrease the percentage of invalid services that do not conform to SLAs under various conditions, such as different of models and percentages of malicious providers and different numbers of CSPs.

To evaluate the performance of the TrustyFeer trust management system in a P2P federated cloud environment, we simulated a representative sample of federated clouds of different scales and conducted a well-controlled experiment using the parameters shown in Table 4.

We identified the critical elements inherent in the design of a trust management system, which are the number of CSPs and the percentages and models of malicious CSPs and controlled their values as follows:

- (1) Number of CSPs was selected in the range $[20 \cdots 70]$.
- (2) Percentage of malicious CSPs was selected in the range $[10 \cdots 60]$.

TABLE 4: The simulation parameters.

Simulation parameter	Values
Application model	File sharing
Network model	Closed network
Numbers of CSPs	20, 30, 40, 50, 60, 70
Percentages of malicious CSPs	10%, 20%, 30%, 40%, 50%, 60%
Number of transactions	750

(3) Models of CSPs were four different models:

- Good providers, which provide honest feedback and a rate of authentic services between 90% and 100%, as we cannot expect good providers to be ideal ones.
- Purely malicious providers that provide inauthentic services or services that do not conform with SLAs or that lie about the feedback on the services they have received from other CSPs regarding the service quality (whether or not they conform to SLAs).
- Malicious feedback by providers that conform to SLAs but consistently lie about the quality of the services they receive.
- A malicious collective consisting of a group of cooperating malicious CSPs.

The extraneous variables, such as the number of transactions, were controlled using randomization to ensure a representative sample in all experiments. Two outstanding reputation algorithms, EigenTrust [30] and TNA-SL [28], were selected as benchmarks. The following two sets of experiments were designed, as shown in Table 5:

- In the first set, the percentage of malicious CSPs was constant at 20%, while the values for the number of CSPs were selected in the range of 20, 30, 40, 50, 60, and 70 CSPs.
- In the second set, the number of CSPs was constant at 40, while the percentage of malicious CSPs was selected in the range of 10%, 20%, 30%, 40%, 50%, and 60% malicious CSPs.

For each set, two mixed models of malicious CSPs were used, with two experiments taking different strategies: *naïve* and *collective*. Moreover, to increase the accuracy of this experimental study, we repeated each experiment ten times and calculated the average of the outcomes. The number of transactions was constant at 750.

5. Results and Discussion

A summary of the results of running 750 transactions for all sets of experiments is shown in Figures 6–13. Each set of experimental results represents an average of 10 simulation runs. As discussed previously, we used the success rate and percentage of invalid services to evaluate the efficiency of our proposed system.

FIGURE 6: Success rate when different numbers of CSPs are considered (collective).

FIGURE 7: Success rate when different numbers of CSPs are considered (naïve).

FIGURE 8: Success rate when different percentages of malicious CSPs are considered (collective).

TABLE 5: Experimental settings.

	Experiments	Number of CSPs	Number of malicious CSPs	Malicious model	Strategy
Set 1: different numbers of CSPs at a constant percentage of malicious CSPs	Ex 1.1	20	4 (20%)	Purely and feedback	Naïve/collective
	Ex 1.2	30	6 (20%)	Purely and feedback	Naïve/collective
	Ex 1.3	40	8 (20%)	Purely and feedback	Naïve/collective
	Ex 1.4	50	10 (20%)	Purely and feedback	Naïve/collective
	Ex 1.5	60	12 (20%)	Purely and feedback	Naïve/collective
	Ex 1.6	70	14 (20%)	Purely and feedback	Naïve/collective
Set 2: constant number of CSPs at different percentages of malicious CSPs	Ex 2.1	40	4 (10%)	Purely and feedback	Naïve/collective
	Ex 2.2	40	8 (20%)	Purely and feedback	Naïve/collective
	Ex 2.3	40	12 (30%)	Purely and feedback	Naïve/collective
	Ex 2.4	40	16 (40%)	Purely and feedback	Naïve/collective
	Ex 2.5	40	20 (50%)	Purely and feedback	Naïve/collective
	Ex 2.6	40	24 (60%)	Purely and feedback	Naïve/collective

FIGURE 9: Success rate when different percentages of malicious CSPs are considered (naïve).

FIGURE 10: Percentage of invalid services when different numbers of CSPs are considered (collective).

FIGURE 11: Percentage of invalid services when different numbers of CSPs are considered (naïve).

5.1. Success Rate. Figures 6–9 illustrate the success rates for good CSPs, calculated as the number of valid services received by good CSPs divided by the number of transactions attempted by good CSPs for each trust system.

In Figures 6 and 7, the success rate of good CSPs is plotted against the number of CSPs in the system. They clearly show that, at a constant percentage, 20%, of malicious CSPs, TrustyFeer and TNA-SL success rates remain almost constant as the number of CSPs increases. However, the TNA-SL success rate is much lower than that of TrustyFeer for all numbers of CSPs. This shows the scalability of TrustyFeer for any number of CSPs. In contrast, EigenTrust has an inconsistent success rate when the number of CSPs is considered. Comparing Figures 7 and 8 suggests that, in contrast to EigenTrust, TrustyFeer and TNA-SL are more powerful in handling complex collective malicious behaviors than naïve peers. Such ability has its root in the subjective logic robust feature where the trust value is calculated based on four factors rather than simply aggregating the rating from previous transitions which is the case with EigenTrust. The additional feature of weighting the feedback of each CSP by

FIGURE 12: Percentage of invalid services when different percentages of malicious CSPs are considered (collective).

FIGURE 13: Percentage of invalid services when different percentages of malicious CSPs are considered (naïve).

its reputation value might be the main driving force behind the superior performance achieved by TrustyFeer.

Figures 8 and 9 plot the success rate of good CSPs against the percentage of malicious CSPs in the system, at a fixed number of 40 CSPs. Despite showing a slight decrease in the success rate as the percentage of malicious CSPs in the system increases, TrustyFeer still outperforms TNA-SL and EigenTrust in all scenarios. When the percentage of malicious CSPs is low, the difference is small. However, the gap increases as the percentage of malicious CSPs in the system increases. Such an observation might be strongly correlated with the way the opinion is calculated in TrustyFeer, where the trust value of a CSP is weighted by its global reputation. This weighting strategy marginalizes the effect of malicious CSPs, especially of those forming collectives, as illustrated by Figure 8. In contrast, the centered strategy of the EigenTrust algorithm, around certain peers, proved to be inefficient when some of them turned out to be malicious which can easily happen in the malicious collectives model.

From the above discussion, we can clearly see that TrustyFeer surpasses TNA-SL and EigenTrust in helping good

CSPs that conform to SLAs in exchanging services. This can be attributed to the fact that TrustyFeer uses a hybrid approach in calculating trust, while TNA-SL and EigenTrust use the peer-based approach.

5.2. Percentage of Invalid Services. In this set of results, the percentage of invalid services exchanged by good CSPs is evaluated for each reputation system.

Figures 10 and 11 illustrate the relationship between the percentage of invalid services exchanged by good CSPs and the number of CSPs in the system. We can see that TrustyFeer outperforms TNA-SL and EigenTrust in terms of the percentage of invalid services exchanged by good CSPs. However, in all scenarios, the percentage of invalid services exchanged by good CSPs remains steady when TrustyFeer is used. Regardless of the P2P federated cloud scale, TrustyFeer has successfully identified the majority of malicious providers and isolated them, so, few good peers mistakenly exchange services with them. This behavior of TrustyFeer has its root in its weighting strategy where the opinion of any CSP is multiplied by its global reputation value. Additionally, the new proposed subjective logic formulas should also have a strong connection to this superior behavior as the SLA attribute values of previous transactions of each CSP are taken into consideration.

Figures 12 and 13 show the success of TrustyFeer in decreasing the percentage of invalid services compared to TNA-SL and EigenTrust. The percentage of invalid services increases steadily for all systems when the percentage of malicious CSPs increases. However, TrustyFeer manages to maintain its success despite such a large percentage of malicious CSPs. This is because TNA-SL and EigenTrust evaluate the quality of peers only when calculating trust values, as both are classified as peer-based reputation management systems. In contrast, being a hybrid system, TrustyFeer evaluates the quality of both the service and peers when calculating trust values, resulting in a more robust reputation management system. Two other factors can also be considered as driving this outstanding performance which are weighting peer's opinion by its reputation values and using new subjective logic formulas that take SLA parameters into consideration.

6. Conclusion

In P2P federated clouds, multiple CSPs can share their resources to facilitate the provision of efficient services to citizens in smart cities. However, associated with the interactions between these individual CSPs are several trust issues that affect the QoS of the federated cloud. In this paper, we have proposed TrustyFeer, a trust management system to enhance the QoS in P2P federated cloud environments. The TrustyFeer system proposes the use of subjective logic equations based on SLAs and the global reputation of CSPs to calculate trust values. The system was evaluated using a strictly controlled simulation environment with varying numbers of CSPs and threat models. In addition, the performance of TrustyFeer was compared against the TNA-SL and EigenTrust reputation management systems. The simulation results showed that TrustyFeer has a positive impact on reducing

the percentage of services that do not conform to SLAs and increasing the success rate of services that do conform to SLAs between good CSPs.

In future work, we plan to implement a P2P trust simulator especially for a P2P cloud environment. This will include a number of virtual machine requests between CSPs with different characteristics, based on a real data repository. This will help us to gain deep insights into system performance when evaluating a P2P cloud ecosystem.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by Saudi Aramco, under the “Saudi Aramco Ibn Khaldun Fellowship for Saudi Women,” in partnership with the Center for Clean Water and Clean Energy at MIT, and the Deanship of Scientific Research at King Saud University through Research Group no. RG-1438-002.

References

- [1] M. J. Kaur and P. Maheshwari, “Building smart cities applications using IoT and cloud-based architectures,” in *Proceedings of the International Conference on Industrial Informatics and Computer Systems, CIICS 2016*, UAE, March 2016.
- [2] A. Zanella, N. Bui, A. P. Castellani, L. Vangelista, and M. Zorzi, “Internet of things for smart cities,” *IEEE Internet of Things Journal*, vol. 1, no. 1, pp. 22–32, 2014.
- [3] P. Mell and T. Grance, “The NIST definition of cloud computing,” *National Institute of Standards and Technology*, vol. 53, no. 6, 2009.
- [4] T. H. Noor, Q. Z. Sheng, S. Zeadally, and J. Yu, “Trust management of services in cloud environments: obstacles and solutions,” *ACM Computing Surveys*, vol. 46, no. 1, article 12, 2013.
- [5] A. J. Ferrer, F. Hernández, J. Tordsson et al., “OPTIMIS: a holistic approach to cloud service provisioning,” *Future Generation Computer Systems*, vol. 28, no. 1, pp. 66–77, 2012.
- [6] P. Riteau, “Building dynamic computing infrastructures over distributed clouds,” in *Proceedings of the 1st IEEE Symposium on Network Cloud Computing and Applications, NCCA 2011*, pp. 127–130, fra, November 2011.
- [7] M. R. M. Assis, L. F. Bittencourt, R. Tolosana-Calasanz, and C. A. Lee, “Cloud federations: requirements, properties, and architectures,” in *Developing Interoperable and Federated Cloud Architecture*, Kecskemeti, G., A. Kertesz, and Z. Nemeth, Eds., Chapter 1, pp. 1–41, IGI Global, Hershey, Pa, USA, 2016.
- [8] O. Shareef and A. Kayed, “A survey on federated clouds environment,” *Journal of Advanced Research in Computer Science and Software Engineering*, vol. 5, no. 2, pp. 83–92, 2015.
- [9] O. Babaoglu, M. Marzolla, and M. Tamburini, “Design and implementation of a P2P Cloud system,” in *Proceedings of the 27th Annual ACM Symposium on Applied Computing, SAC 2012*, pp. 412–417, Italy, March 2012.
- [10] T. Hardjono, D. Shrier, and A. Pentland, *TRUST::DATA: A New Framework for Identity and Data Sharing*, Visionary Future LLC, 2016.
- [11] U. Kaur and D. Singh, “Trust: models and architecture in cloud computing,” *International Journal of Computer Science and Information Security (IJCSIS)*, vol. 13, no. 12, 150 pages, 2015.
- [12] M. Feldman, C. Papadimitriou, J. Chuang, and I. Stoica, “Free-riding and whitewashing in peer-to-peer systems,” *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 5, pp. 1010–1018, 2006.
- [13] F. Z. Filali and B. Yagoubi, “Global trust: a trust model for cloud service selection,” *International Journal of Computer Network and Information Security*, vol. 7, no. 5, pp. 41–50, 2015.
- [14] P. Resnick and R. Zeckhauser, “Trust among strangers in internet transactions: empirical analysis of eBay’s reputation system,” *Advances in Applied Microeconomics*, vol. 11, pp. 127–157, 2002.
- [15] S. Brin and L. Page, “The anatomy of a large-scale hypertextual web search engine,” *Computer Networks*, vol. 56, no. 18, pp. 3825–3833, 2012.
- [16] J. Huang, F. Nie, H. Huang, and Y.-C. Tu, “Trust prediction via aggregating heterogeneous social networks,” in *Proceedings of the 21st ACM International Conference on Information and Knowledge Management, CIKM 2012*, pp. 1774–1778, USA, November 2012.
- [17] K. K. Bharadwaj and M. Y. H. Al-Shamri, “Fuzzy computational models for trust and reputation systems,” *Electronic Commerce Research and Applications*, vol. 8, no. 1, pp. 37–47, 2009.
- [18] N. Iltaf and A. Ghafoor, “A fuzzy based credibility evaluation of recommended trust in pervasive computing environment,” in *Proceedings of the 2013 IEEE 10th Consumer Communications and Networking Conference, CCNC 2013*, pp. 617–620, USA, January 2013.
- [19] S. Song, K. Hwang, and Y.-K. Kwok, “Risk-resilient heuristics and genetic algorithms for security-assured grid job scheduling,” *IEEE Transactions on Computers*, vol. 55, no. 6, pp. 703–719, 2006.
- [20] S. Song, K. Hwang, R. F. Zhou, and Y.-K. Kwok, “Trusted P2P transactions with fuzzy reputation aggregation,” *IEEE Internet Computing*, vol. 9, no. 6, pp. 24–34, 2005.
- [21] P. Sarathi Banerjee, J. Paulchoudhury, and S. R. Bhadra Chaudhuri, “Fuzzy membership function in a trust based AODV for MANET,” *International Journal of Computer Network and Information Security*, vol. 5, no. 12, pp. 27–34, 2013.
- [22] M. Tavakolifard and S. J. Knapkog, “A probabilistic reputation algorithm for decentralized multi-agent environments,” *Electronic Notes in Theoretical Computer Science*, vol. 244, pp. 139–149, 2009.
- [23] W. T. Teacy, M. Luck, A. Rogers, and N. R. Jennings, “An efficient and versatile approach to trust and reputation using hierarchical Bayesian modelling,” *Artificial Intelligence*, vol. 193, pp. 149–185, 2012.
- [24] Y. Zhang and Y. Fang, “A fine-grained reputation system for reliable service selection in peer-to-peer networks,” *IEEE Transactions on Parallel and Distributed Systems*, vol. 18, no. 8, pp. 1134–1145, 2007.
- [25] T. Muller and P. Schweitzer, “On Beta Models with Trust Chains,” in *Proceedings of the Trust Management VII: 7th IFIP WG 11.11 International Conference, IFIPTM*, Malaga, Spain, 2013.
- [26] T. van Deursen, P. Koster, and M. Petković, “Hedaquin: a reputation-based health data quality indicator,” *Electronic Notes in Theoretical Computer Science*, vol. 197, no. 2, pp. 159–167, 2008.
- [27] A. Jøsang, “A logic for uncertain probabilities,” *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 9, no. 3, pp. 279–311, 2001.

- [28] A. Jøsang, R. Hayward, and S. Pope, "Trust network analysis with subjective logic," in *Proceedings of the 29th Australasian Computer Science Conference (ACSC2006)*, CRPIT, Hobart, Australia, January 2006.
- [29] R. Zhou and K. Hwang, "Trust overlay networks for global reputation aggregation in P2P grid computing," in *Proceedings of the 20th IEEE International Parallel Distributed Processing Symposium, Rhodes Island*, 2006.
- [30] S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina, "The EigenTrust algorithm for reputation management in P2P networks," in *Proceedings of the 12th International Conference on World Wide Web (WWW '03)*, pp. 640–651, Budapest, Hungary, May 2003.
- [31] J. Wang and J. Liu, "The comparison of distributed P2P trust models based on quantitative parameters in the file downloading scenarios," *Journal of Electrical and Computer Engineering*, vol. 2016, Article ID 4361719, 2016.
- [32] Z. Su, L. Liu, M. Li, X. Fan, and Y. Zhou, "Reliable and resilient trust management in distributed service provision networks," *ACM Transactions on the Web (TWEB)*, vol. 9, no. 3, article no. 14, 2015.
- [33] N. Alhadad, Y. Busnel, P. Serrano-Alvarado, and P. Lamarre, "Trust evaluation of a system for an activity with subjective logic," in *Proceedings of the International Conference on Trust, Privacy and Security in Digital Business*, vol. 8647, pp. 48–59, Munich, Germany, 2014.
- [34] E. Choo, J. Jiang, and T. Yu, "COMPARS: toward an empirical approach for comparing the resilience of reputation systems," in *Proceedings of the 4th ACM Conference on Data and Application Security and Privacy, CODASPY 2014*, pp. 87–98, USA, March 2014.
- [35] H. A. Kurdi, "HonestPeer: An enhanced EigenTrust algorithm for reputation management in P2P systems," *Journal of King Saud University—Computer and Information Sciences*, vol. 27, no. 3, pp. 315–322, 2015.
- [36] "QTM: P2P Trust Simulator, 2009," <https://rtg.cis.upenn.edu/qtm/p2psim.php3>.
- [37] L. Xiong and L. Liu, "PeerTrust: supporting reputation-based trust for peer-to-peer electronic communities," *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, no. 7, pp. 843–857, 2004.
- [38] R. Zhou and K. Hwang, "PowerTrust: a robust and scalable reputation system for trusted peer-to-peer computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 18, no. 4, pp. 460–473, 2007.
- [39] R. Chen, X. Zhao, L. Tang, J. Hu, and Z. Chen, "CuboidTrust: a global reputation-based trust model in peer-to-peer networks," in *Autonomic and Trusted Computing*, vol. 4610, pp. 203–215, Springer, Hong Kong, China, 2007.
- [40] E. Ayday and F. Fekri, "BP-P2P: Belief propagation-based trust and reputation management for P2P networks," in *Proceedings of the 2012 9th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks, SECON 2012*, pp. 578–586, Republic of Korea, June 2012.
- [41] R. Zhou and K. Hwang, "Gossip-based reputation aggregation for unstructured peer-to-peer networks," in *Proceedings of the 21st International Parallel and Distributed Processing Symposium, IPDPS 2007*, USA, March 2007.
- [42] L. Huaiqing, W. Xuezhi, and L. Haitao, "Hierarchical fuzzy trust management for peer-to-peer network," in *Proceedings of the 2009 Second ISECS International Colloquium on Computing, Communication, Control, and Management, CCCM 2009*, pp. 377–380, China, August 2009.
- [43] H. Zhao and X. Li, "VectorTrust: Trust vector aggregation scheme for trust management in peer-to-peer networks," in *Proceedings of the 18th International Conference on Computer Communications and Networks (ICCCN '09)*, IEEE, August 2009.
- [44] E. D. Canedo, R. Junior, and R. Albuquerque, "Trust model for reliable file exchange in cloud computing," *International Journal of Computer Science & Information Technology (IJCSIT)*, vol. 4, no. 1, 2012.
- [45] N. Dladlu and O. O. Ekabua, "Implementation of a novel peer-to-peer reputation-based trust management model in a cloud service provisioning environment," in *The International Conference on Digital Information Processing, E-Business and Cloud Computing*, Kuala Lumpur, Malaysia, 2016.
- [46] I. Petri, O. F. Rana, Y. Rezgui, and G. C. Silaghi, "Trust modelling and analysis in peer-to-peer clouds," *International Journal of Cloud Computing*, vol. 1, no. 2/3, p. 221, 2012.
- [47] J. Abawajy, "Determining service trustworthiness in intercloud computing environments," in *Proceedings of the 10th International Symposium on Pervasive Systems, Algorithms, and Networks, I-SPAN 2009*, pp. 784–788, Taiwan, December 2009.
- [48] S. Y. Lee, O.-H. Kwon, J. Kim, and S. J. Hong, "A reputation management system in structured peer-to-peer networks," in *Proceedings of the 14th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WET ICE '05)*, pp. 362–367, June 2005.
- [49] H. Kurdi, S. Alnasser, and M. Alhelal, "AuthenticPeer: a reputation management system for peer-to-peer wireless sensor networks," *International Journal of Distributed Sensor Networks*, vol. 11, no. 11, Article ID 637831, 2015.
- [50] A. Jøsang, R. Ismail, and C. Boyd, "A survey of trust and reputation systems for online service provision," *Decision Support Systems*, vol. 43, no. 2, pp. 618–644, 2007.
- [51] S. Ruohomaa, L. Kutvonen, and E. Koutrouli, "Reputation management survey," in *Proceedings of the 2nd International Conference on Availability, Reliability and Security, ARES 2007*, pp. 103–111, Australia, April 2007.
- [52] D. Kaur and J. S. Gupta, "Proposed P2P reputation-based model to secure grid," in *Proceedings of the International Conference on Recent Advances Trends in Information Technology (iRAFIT)*, Patiala, Punjab, India, 2012.
- [53] F. G. Mármol and G. M. Pérez, "Security threats scenarios in trust and reputation models for distributed systems," *Computers & Security*, vol. 28, no. 7, pp. 545–556, 2009.
- [54] B. Wickremasinghe, R. N. Calheiros, and R. Buyya, "CloudAnalyst: a cloudsims-based visual modeller for analysing cloud computing environments and applications," in *Proceedings of the 24th IEEE International Conference on Advanced Information Networking and Applications (AINA '10)*, pp. 446–452, Perth, Wash, USA, April 2010.