Credibility-Based Trust Management for Services in Cloud Environments

Talal H. Noor and Quan Z. Sheng

School of Computer Science,
The University of Adelaide, Adelaide SA 5005, Australia
{talal,qsheng}@cs.adelaide.edu.au

Abstract. Trust management is one of the most challenging issues in the emerging cloud computing. Although many approaches have been proposed recently for trust management in cloud environments, not much attention has been given to determining the credibility of trust feedbacks. Moreover, the dynamic nature of cloud environments makes guaranteeing the availability of trust management services a difficult problem due to the unpredictable number of cloud consumers. In this paper, we propose a framework to improve ways on trust management in cloud environments. In particular, we introduce a credibility model that not only distinguishes between credible trust feedbacks, but also has the ability to detect the malicious trust feedbacks from attackers. We also present a replication determination model that dynamically decides the optimal replica number of the trust management service so that the trust management service can be always maintained at a desired availability level. The approaches have been validated by the prototype system and experimental results.

Keywords: Trust Management, Cloud Computing, Credibility Model, Service Availability.

1 Introduction

In recent years, cloud computing has been receiving much attention as a new computing paradigm for providing flexible and on-demand infrastructures, platforms and software as services [2,6]. Both the public and the private sectors can benefit from the adoption of cloud services. For instance, it only took 24 hours, at the cost of merely $240, for the New York Times to archive its 11 million articles (1851-1980) using Amazon Web Services[1].

Given the fact of the accelerated adoption of cloud computing in the industry, there is a significant challenge in managing trust among cloud providers, service providers, and service requesters. Indeed, trust is one of the top obstacles for the adoption and the growth of cloud computing [2,6,14]. Recently, a considerable amount of research works have recognized the significance of trust management

[1] http://open.blogs.nytimes.com/2007/11/01/self-service-prorated-super-computing-fun/

and proposed several solutions to assess and manage trust based on trust feedbacks collected from participants [14,7,26,9]. However, one particular problem has been mostly neglected: to what extent can these trust feedbacks be credible. On the one hand, it is not unusual that a trust management system will experience malicious behaviors from its users. On the other hand, the quality of the trust feedbacks differs from one person to another, depending on how experienced she is. This paper focuses on improving ways on the trust management in cloud environments. In particular, we distinguish the following key issues of the trust management in cloud environments:

– **Trust Robustness.** Determining the credibility of trust feedbacks is a significant challenge due to the overlapping interactions between service requesters, service providers, and cloud providers. This is true because cloud service interactions are dynamic. It is more likely that a cloud consumer has many interactions with the same cloud service, leading to multiple trust feedbacks to the cloud service. In addition, it is difficult to know how experienced a cloud consumer is and from whom malicious trust feedbacks are expected. Indeed, the trust management protection still requires extensive probabilistic computations [29,16] and trust participants' collaboration by manually rating trust feedbacks [19].
– **Availability of the Trust Management Service.** In a cloud environment, guaranteeing the availability of the trust management service is a difficult problem due to the unpredictable number of cloud consumers and the highly dynamic nature of the cloud environment. Consequently, approaches that requires understanding of the trust participants' interests and capabilities through similarity measurements [25] are inappropriate in the cloud environment. Trust management systems should be adaptive and highly scalable.
– **Trust Feedback Assessment and Storage.** The trust assessment of a service in existing techniques is usually centralized, whereas the trust feedbacks come from distributed trust participants. Trust models that follow a centralized architecture are more prone to several problems including scalability, availability, and security (e.g., Denial of Service (DoS) attack) [13]. Given the open and distributed nature of cloud environments, we believe that centralized solutions are not suitable for trust feedback assessment and storage.

In this paper, we overview the design and the implementation of the trust management framework. This framework helps distinguish between credible trust feedbacks and malicious trust feedbacks through a credibility model. It also guarantees high availability of the trust management service. In a nutshell, the salient features of the framework are:

– **A Credibility Model.** We develop a credibility model that not only distinguishes between trust feedbacks from experienced cloud consumers and amateur cloud consumers, but also has the ability to detect the malicious

trust feedbacks from attackers (i.e., who intend to manipulate the trust results by giving multiple trust feedbacks to a certain cloud service in a short period of time).
- **A Replication Determination Model.** High availability is an important requirement to the trust management service. We propose to spread replicas of the trust management service and develop a *replication determination model* that dynamically determines the optimal number of trust management service replicas, which share the trust management workload, thereby always maintaining the trust management service at a desired availability level.
- **Distributed Trust Feedback Assessment and Storage.** To avoid the drawbacks of centralized architectures, our trust management service allows trust feedback assessment and storage to be managed in a distributed way. Each trust management service replica is responsible for trust feedbacks given to a set of cloud services.

The remainder of the paper is organized as follows. Section 2 overviews the related work. Section 3 briefly presents the design of the trust management framework. Section 4 details the trust management service, including the distributed trust feedback collection and assessment, as well as the replication determination model for high availability of the trust management service. Section 5 describes the details of our credibility model. Finally, Section 6 reports the implementation and several experimental evaluations and Section 7 provides some concluding remarks.

2 Related Work

Several research works recognized the significance of trust management [15,28,13]. In particular, trust management is considered as one of the critical issues in cloud computing and is becoming a very active research area in recent years [17,21,14,5].

Several trust management approaches were proposed as policy-based trust management. For instance, Hwang et al. [14] proposed a security aware cloud architecture that uses VPN or SSL for communication security, focusing on both the cloud provider's and the cloud consumer's perspectives. In the cloud provider's perspective, the proposed architecture uses the trust negotiation approach and the data coloring (integration) using fuzzy logic techniques. In the cloud consumer's perspective, the proposed architecture uses the Distributed-Hash-Table (DHT)-based trust-overlay networks among several data centers to deploy a reputation-based trust management technique. Brandic et al. [5] proposed a novel approach for compliance management in cloud environments to establish trust between different parties. The centralized architecture focuses on the cloud consumer's perspective that uses compliant management to help cloud consumers to have proper choices when selecting cloud services. Unlike previous works that use centralized architecture, we present a credibility model supporting distributed trust feedback assessment and storage. This credibility model also distinguishes between trustworthy and malicious trust feedbacks.

Other trust management approaches were proposed as reputation-based trust management. For example, Conner et al. [9] proposed a trust management framework for the service-oriented architecture (SOA) that focuses on the service provider's perspective to protect resources from unauthorized access. This framework has a decentralized architecture that offers multiple trust evaluation metrics, allowing service providers to have customized evaluation of their clients (i.e., service requesters). Malik and Bouguettaya [20] proposed reputation assessment techniques based on the existing quality of service (QoS) parameters. The proposed framework supports different assessment metrics such as majority rating, past rating history, personal experience for credibility evaluation, etc. Unlike previous works that require extensive computations or trust participants' collaboration by rating the trust feedbacks, we present a credibility model that include several metrics namely the *Majority Consensus* and the *Feedback Density* which facilitates the determination of credible trust feedbacks. We were inspired by Xiong and Liu who differentiate between the credibility of a peer and the credibility of a feedback through distinguishing several parameters to measure the credibility of the trust participants feedbacks [30]. However, their approach is not applicable in cloud environments because peers supply and consume services and they are evaluated on that base. In other words trust results are used to distinguish between credible and malicious feedbacks.

3 The Trust Management Framework

We propose a trust management framework based on the service-oriented architecture (SOA). In particular, our framework uses Web services to span several distributed trust management service nodes. Trust participants (i.e., the cloud consumers) can give their trust feedbacks or inquire about a certain cloud service's trust results using Simple Object Access Protocol (SOAP) or REST [24] messages. We design our framework in this way because of the dynamic nature of cloud environments (e.g., new cloud consumers can join while others might leave around the clock). This requires the trust management service to be adaptive and highly scalable in order to collect the trust feedbacks and update the trust results constantly. Figure 1 depicts the main components of the trust management framework, which consists of two different layers, namely the *Service Layer* and the *Service Requester Layer.*

The Service Layer. This layer represents the big umbrella which includes cloud services (i.e., IaaS (Infrastructure as a Service), PaaS (Platform as a Service), and SaaS (Software as a Service)), e-services (e.g., booking a flight) and the trust management service where a service requester can give trust feedbacks to a particular service. Interactions within this layer are considered as *Cloud Service Interaction* and *Trust Interaction.*

The Service Requester Layer. This layer consists of different service requesters who consume services in the service layer. For example, a user can book a flight

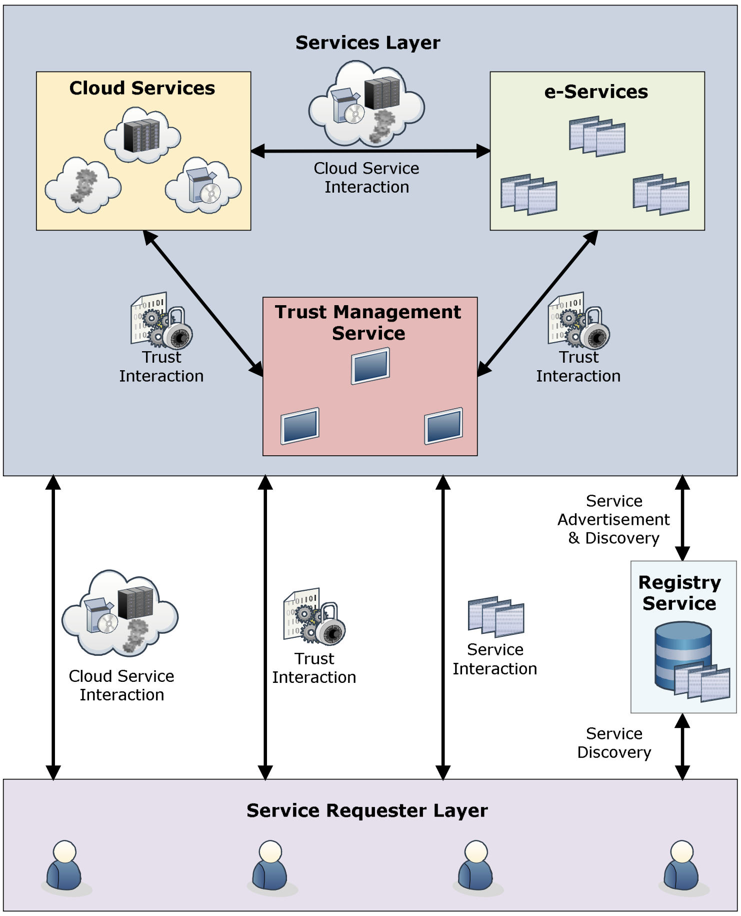


Fig. 1. Architecture of the Trust Management Framework

through an e-service provided by a certain airline company. A new startup that has limited funding can consume cloud services (e.g., hosting their services in Amazon S3). Service requesters can give trust feedbacks of a particular cloud service by invoking the trust management service (see Section 4).

Our framework also contains a *Registry Service* (see Figure 1) that has the following responsibilities:

– *Service Advertisement.* Both cloud providers and service providers are able to advertise their services through the *Service Registry.*
– *Service Discovery.* Service providers, cloud providers, and service requesters are able to access the *Service Registry* to discover services.

3.1 Assumptions and Attack Models

We assume that communications are secure. Attacks that occur in the *communication security level* such as *Man-in-the-Middle* (MITM) attack [3] are beyond the scope of this work. We also assume that cloud consumers have unique identities. Attacks that use the notion of multiple identities (i.e., the *Sybil* attack [12]) or *Whitewashing* attack that occur when the malicious cloud consumers (i.e., attackers) desperately seek new identities to clean their negative history records [18] are also beyond the scope of this work. In this paper, we only consider two types of malicious behaviors including *Self-promoting* attack and *Slandering* attack.

Self-promoting Attack. This attack arises when the malicious cloud consumers attempt to increase their trust results [10] or their allies in order to achieve their common interests. In the proposed framework this type of attack can happen in two cases. The first case (*Individual Collusion*) occurs when a certain malicious cloud consumer gives numerous fake or misleading trust feedbacks to increase the trust results of a certain cloud service. The second case (*Collaborative Collusion*) occurs when several malicious cloud consumers collaborate to give numerous fake or misleading trust feedbacks.

Slandering Attack. This attack is considered as the opposite of the *Self-promoting* attack that happens when the malicious cloud consumers try to decrease the trust results of certain cloud service [4]; this aggressive behavior is taken because of jealousy from competitors. In the proposed framework this type of attack can also happen either through *Individual Collusion* or *Collaborative Collusion.*

Service requesters can give trust feedbacks for a certain cloud service or send a query to the trust management service regarding a certain cloud service. In the following sections, we will focus on introducing our design of the trust management service.

4 Trust Management Service

4.1 Trust Feedback Collection and Assessment

In our framework, the trust behavior of a cloud service is represented by a collection of invocation history records denoted as $\mathcal{H}$. Each cloud consumer c holds her point of view regarding the trustworthiness of a specific cloud service s in the invocation history record which is managed by a trust management service. Each invocation history record is represented in a tuple that consists of the cloud consumer primary identity $\mathcal{C}$, the cloud service identity $\mathcal{S}$, a set of trust feedbacks $\mathcal{F}$ and the aggregated trust feedbacks weighted by the credibility $\mathcal{F}_c$ (i.e., $\mathcal{H} = (\mathcal{C}, \mathcal{S}, \mathcal{F}, \mathcal{F}_c)$). Each feedback in $\mathcal{F}$ is represented in numerical form with the range of [0, 1], where 0, +1, and 0.5 means negative feedback, positive feedback, and neutral respectively.

Whenever a cloud consumer inquires the trust management service regarding the trustworthiness of a certain cloud service s, the trust result, denoted as $\mathcal{T}r(s)$, is calculated as the following:

$$\mathcal{T}r(s) = \frac{\sum_{l=1}^{|\mathcal{V}(s)|} \mathcal{F}_c(l, s)}{|\mathcal{V}(s)|} \tag{1}$$

where $\mathcal{V}(s)$ is all of the feedbacks given to the cloud service s and $|\mathcal{V}(s)|$ represents the length of the $\mathcal{V}(s)$ (i.e., the total number of feedbacks given to the cloud service s). $\mathcal{F}_c(l, s)$ are the trust feedbacks from the l^{th} cloud consumer weighted by the credibility.

The trust management service distinguishes between credible trust feedbacks and malicious trust feedbacks through assigning the credibility aggregated weights $\mathcal{C}r(l, s)$ to the trust feedbacks $\mathcal{F}(l, s)$ as shown in Equation 2, where the result $\mathcal{F}_c(l, s)$ is held in the invocation history record h and updated in the assigned trust management service. The details on how to calculate $\mathcal{C}r(l, s)$ is described in Section 5.

$$\mathcal{F}_c(l, s) = \mathcal{F}(l, s) * \mathcal{C}r(l, s) \tag{2}$$

4.2 Availability of the Trust Management Service

Guaranteeing the availability of the trust management service is a significant challenge due to unpredictable number of invocation requests the service has to handle at a time, as well as the dynamic nature of the cloud environments. An emerging trend for solving the high-availability issue is centered on replication. In our approach, we propose to spread trust management service replicas over various clouds and dynamically direct requests to appropriate clouds (e.g., with lower workload), so that its desired availability level can be always maintained.

However, there is clearly a trade-off between high availability and replication cost. On the one hand, more clouds hosting trust management service means better availability. On the other hand, more replicas residing at various clouds means higher overhead (e.g., cost and resource consumption such as bandwidth and storage space). Thus, it is essential to develop a mechanism that helps to determine the optimal number of the trust management service replicas in order to meet the trust management service's availability requirement.

We propose a replication determination model to allow the trust management service to know how many replicas are required to achieve a certain level of availability. Given the trust management service s_{tms} failure probability denoted p that ranges from 0 to 1, the total number of s_{tms} replicas denoted r, and the availability threshold denoted e_a that also ranges from 0 to 1. The desired goal of the replication is to ensure that at least one replica of the trust management service is available, represented in the following formula:

$$e_a(s_{tms}) < 1 - p^{r(s_{tms})} \tag{3}$$

where $p^{r(s_{tms})}$ represents the probability that all trust management service replicas are failed, and $1-p^{r(s_{tms})}$ represents the opposite (i.e., the probability of at least one trust management replica is available). As a result, the optimal number of trust management service replicas can be calculated as follows:

$$r(s_{tms}) > log_p(1 - e_a(s_{tms})) \tag{4}$$

For example, if the availability threshold $e_a(s_{tms}) = 0.9999$ and the failure probability of the trust management service $p = 0.2$ (low), $r(s_{tms}) > 5.723$, meaning that at least 6 trust management service replicas are needed. Similarly, if $e_a(s_{tms})$ $= 0.9999$ and the failure probability of the trust management service $p = 0.8$ (high), $r(s_{tms}) > 41.28$ which means at least 42 replicas are required.

Whenever a cloud consumer needs to send the invocation history record or query the trust result of a certain cloud service, $h(c, s)$ can be sent to a particular trust management service decided by using a consistent hash function (e.g., sha-256) as follows:

$$Tms_{id}(s) = \left(\sum_{i=1}^{|hash(s)|} byte_i \left(hash(s)\right) \right) mod\ r(s_{tms}) \tag{5}$$

where the first part of the equation represents the sum of each byte of the hashed cloud service identity $hash(s)$. The second part of the equation represents the optimal number of the trust management service replicas $r(s_{tms})$. This insures that the chosen trust management service replica is within the optimal number range.

5 The Credibility Model

Since the trust behavior of a cloud service in our framework is represented by a collection of invocation history records that contain cloud consumers trust feedbacks, there is a considerable possibility of the trust management service receiving *inaccurate* or even *malicious* trust feedbacks from amateur cloud consumers (e.g., who lack experience) or vicious cloud consumers (e.g., who submit lots of negative feedbacks in a short period in order to disadvantage a particular cloud service). To overcome these issues, we propose a *credibility model*, which considers several factors including the *Majority Consensus* and the *Feedback Density*.

5.1 Majority Consensus

It is well-known that the majority of people usually agree with experts' judgments about what is good [8]. Similarly, we believe that the majority of cloud consumers agree with *Expert Cloud Consumers'* judgments. In other words, any cloud consumer whose trust feedback is close to the majority trust feedbacks is considered as an *Expert Cloud Consumer*, *Amateur Cloud Consumers* otherwise.

In order to measure how close the cloud consumer's trust feedbacks to the majority trust feedbacks (i.e., the *Majority Consensus*, $\mathcal{J}(c)$), we use the slandered deviation (i.e., the root-mean-square) which is calculated as follows:

$$\mathcal{J}(c) = 1 - \sqrt{\frac{\sum_{h \in \mathcal{V}c(c)} \left(\sum_{k=1}^{|\mathcal{V}c(c,k)|} \left(\frac{\mathcal{F}(c,k)}{|\mathcal{V}c(c,k)|} - \left(\frac{\sum_{l \neq c, l=1}^{|\mathcal{V}c(l,k)|} \mathcal{F}(l,k)}{|\mathcal{V}(k)| - |\mathcal{V}c(c,k)|} \right) \right) \right)^2}{|\mathcal{V}c(c)|}} \tag{6}$$

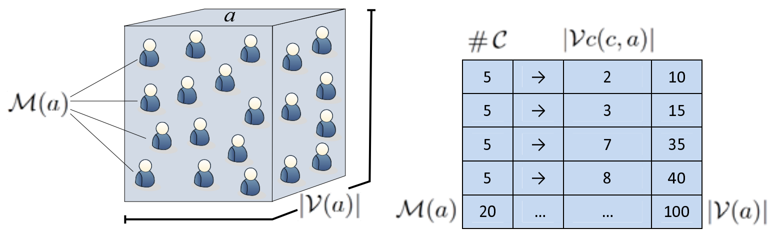
where the first part of the numerator represents the mean of the cloud consumer c's trust feedbacks $\mathcal{F}(c,k)$ for the k^{th} cloud service. The second part of the numerator represents the mean of the majority trust feedbacks given by other cloud consumers denoted $\mathcal{F}(l,k)$ (i.e., the l^{th} cloud consumer trust feedbacks, except the cloud consumer c's trust feedbacks) to the k^{th} cloud service. This procedure is done for all cloud services to which cloud consumer c gives trust feedbacks (i.e., $\mathcal{V}c(c)$).

5.2 Feedback Density

Some malicious cloud consumers may give numerous fake or misleading trust feedbacks to increase or decrease the trust result for a certain cloud service in order to achieve their personal interests (i.e., *Self-promoting* and *Slandering* attacks). Several online reputation-based systems such as auction systems (e.g., eBay [11], and Amazon [1]), have tried to help their consumers to overcome such attacks based on revealing the aggregated trust feedbacks as well as the number of trust feedbacks. The number of trust feedbacks gives the evaluator a hint in determining how credible the trust feedback is, which is supported by the research findings in [30,27].

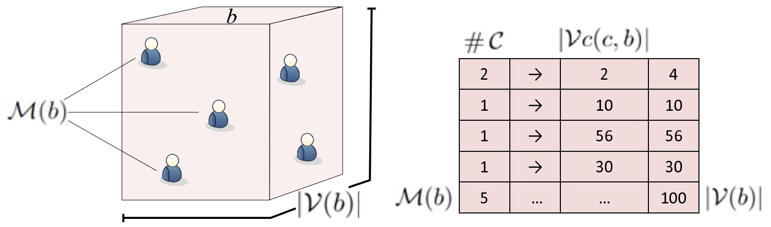
However, the number of trust feedbacks is not enough in determining how credible the aggregated trust feedbacks are. For instance, suppose there are two different cloud services a and b as shown in Figure 2. The aggregated trust feedbacks of the both cloud services are high (i.e., a has 90% positive feedbacks from 100 feedbacks, b has 93% positive feedbacks from 100 feedbacks). Intuitively, cloud consumers should proceed with the cloud service that has the highest aggregated trust feedbacks (e.g., cloud service b in our case). However, *Self-promoting* attack might has been performed on cloud service b, which clearly should not be selected by cloud consumers.

In order to overcome this problem, we introduce the concept of *Feedback Density*, which facilitates the determination of credible trust feedbacks. Specifically, we consider the total number of cloud consumers who gave trust feedbacks to a particular cloud service as the *Feedback Mass*, the total number of trust feedbacks given to the cloud service as the *Feedback Volume*. The feedback volume is influenced by the *Feedback Volume Collusion* factor which is controlled by a specified volume collusion threshold. This factor regulates the multiple trust feedbacks extent that could collude the overall trust feedback volume. For instance, if the volume collusion threshold is set to 5 feedbacks, any cloud consumer



	$\#\mathcal{C}$		$\lvert\mathcal{V}c(c,a)\rvert$		
	5	→	2	10	
	5	→	3	15	
	5	→	7	35	
	5	→	8	40	
$\mathcal{M}(a)$	20	…	…	100	$\lvert\mathcal{V}(a)\rvert$

(a) Cloud Service a



	$\#\mathcal{C}$		$\lvert\mathcal{V}c(c,b)\rvert$		
	2	→	2	4	
	1	→	10	10	
	1	→	56	56	
	1	→	30	30	
$\mathcal{M}(b)$	5	…	…	100	$\lvert\mathcal{V}(b)\rvert$

(b) Cloud Service b

Fig. 2. Trust Feedback Density Determination

c who gives more than 5 feedbacks is considered to be suspicious of involving in a feedback volume collusion. The feedback density of a certain cloud service s, $\mathcal{D}(s)$, is calculated as follows:

$$\mathcal{D}(s) = \frac{\mathcal{M}(s)}{|\mathcal{V}(s)| * \left(\left(\frac{\sum_{h\in\mathcal{V}(s)}\left(\sum_{l=1}^{|\mathcal{V}(l,s)|}\left(\sum_{|\mathcal{V}c(l,s)|>e_v(s)}|\mathcal{V}c(l,s)|\right)\right)}{|\mathcal{V}(s)|}\right)+1\right)} \tag{7}$$

where $\mathcal{M}(s)$ denotes the total number of cloud consumers who gave trust feedbacks to the cloud service s (i.e., the *Feedback Mass*). $|\mathcal{V}(s)|$ represents the total number of trust feedbacks given to the cloud service s (i.e., the *Feedback Volume*). The second part of the denominator represents the *Feedback Volume Collusion* factor. This factor is calculated as the ratio of the number of trust feedbacks given by the cloud consumers who give feedbacks more than the specified volume collusion threshold (i.e., $e_v(s)$) over the total number of feedbacks received by the cloud service (i.e., $|\mathcal{V}(s)|$). The idea behind adding 1 to this ratio is to reduce the value of the multiple trust feedbacks which are given diversely from the same cloud consumer.

Figure 2 depicts the same example mentioned before where the first row in the table on the right side of Figure 2(a) shows that 5 particular cloud consumers gave 2 feedbacks to the cloud service a in which the total number of those trust feedbacks is 10. The last row shows the total number of cloud consumers (i.e., $\mathcal{M}(a) = 20$) and the total number of trust feedbacks given to the cloud service a (i.e., $|\mathcal{V}(a)| = 100$). Both cloud services a and b have the same total number of trust feedbacks (i.e., $|\mathcal{V}(a)| = 100$ and $|\mathcal{V}(b)| = 100$) and very close aggregated feedbacks (e.g., a has 90% positive feedbacks and b has 93% positive feedbacks).

However, the *Feedback Mass* of the cloud service a is higher than the Feedback Mass of the cloud service b (i.e., $\mathcal{M}(a) = 20$ and $\mathcal{M}(b) = 5$). If the volume collusion threshold e_v is set to 3 feedbacks per cloud consumer, 15 cloud consumers gave more than 3 feedbacks to the cloud service a where the total amount of trust feedbacks' lengths $|\mathcal{V}c(c, a)| = 70$ feedbacks; while 3 cloud consumers gave more than 3 feedbacks to the cloud service b where the total amount of trust feedbacks' lengths $|\mathcal{V}c(c, b)| = 80$ feedbacks. According to Equation 7, the *Feedback Density* of the cloud service a is higher than the cloud service b (i.e., $\mathcal{D}(a) = 0.118$ and $\mathcal{D}(b) = 0.028$). In other words, the higher the *Feedback Density*, the more credible the aggregated feedbacks are. The lower the *Feedback Density*, the higher possibility of collusion in the aggregated feedbacks.

Based on the specified trust feedbacks credibility factors (i.e., majority consensus and feedback density), the trust management service distinguishes between trust feedbacks from experienced cloud consumers and the ones from amateur or even vicious cloud consumers through assigning the credibility aggregated weights $\mathcal{C}r(c, s)$ to each of the cloud consumers trust feedbacks as shown in Equation 2. The credibility aggregated weights $\mathcal{C}r(c, s)$ is calculated as follows:

$$\mathcal{C}r(c, s) = \frac{\mu * \mathcal{J}(c) + \rho * \mathcal{D}(s)}{\lambda} \tag{8}$$

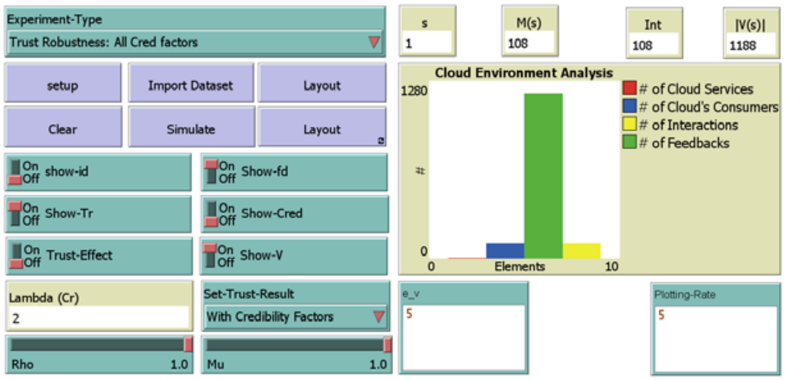
where μ and $\mathcal{J}(c)$ denote the *Majority Consensus* factor's normalized weight (i.e., parameter) and the factor's value respectively. The second part of the equation represents the *Feedback Density* factor where ρ denotes the factor's normalized weight and $\mathcal{D}(s)$ denotes the factor's value. λ represents the number of factors used to calculate $\mathcal{C}r(c, s)$. For example, if we only consider majority consensus, $\lambda = 1$; if we consider both the majority consensus and the feedback density, $\lambda = 2$.

6 Implementation and Experimental Evaluation

In this section, we report the implementation and preliminary experimental results in validating the proposed approach. Our implementation and experiments were developed based on the NetLogo platform [23], which was used to simulate the cloud environments. We particularly focused on validating and studying the performance of the proposed credibility model (see Section 5).

Since it is hard to find some publicly available real-life trust data sets, in our experiments, we used Epinions[2] rating data set which was collected by Massa and Avesani [22]. The reason that we chose Epinions data set is due to its similar data structure (i.e., consumers' opinions and reviews on specific products and services) with our cloud consumer trust feedbacks. In particular, we considered `user_id` in Epinions as the cloud consumer primary identity $\mathcal{C}$, `item_id` as the cloud service identity $\mathcal{S}$, and we normalized the `rating_value` as the cloud

[2] http://www.trustlet.org/wiki/Downloaded_Epinions_dataset


Experiment-Type
Trust Robustness: All Cred factors

s
1

M(s)
108

Int
108

|V(s)|
1188

Lambda (Cr)
2

Set-Trust-Result
With Credibility Factors

Rho 1.0

Mu 1.0

e_v
5

Plotting-Rate
5

Fig. 3. Netlogo-based Prototype System's GUI

consumers trust feedbacks $\mathcal{F}$ to scale of 0 to 1. The data set has 49,290 users, 139,738 items, and 664,824 trust feedbacks. Figure 3 depicts the Graphical User Interface (GUI) for a cloud service. We imported the Epinions data set to create the cloud environment that we are intending to analyze.

We evaluate the trust robustness of our credibility model using both *analytical analysis* and *empirical analysis*. The analytical analysis focuses on measuring the trust result robustness (i.e., with respect to *Malicious Behavior Rate* of malicious cloud consumers) when using the credibility model and without using the credibility model. The analytical model calculates the trust results without weighting the trust results (i.e., we turn the $\mathcal{C}r(c,s)$ to 1 for all trust feedbacks). The empirical analysis focuses on measuring the trust result robustness for each factor in our credibility model including the *Majority Consensus* and the *Feedback Density*. The parameters setup for each corresponding experiment factor are depicted in Table 1.

Table 1. Experiment Factors and Parameters Setup

Experiment Design	μ	ρ	λ	$\mathcal{C}r(c,s)$
With Credibility factors	1	1	2	
Without Credibility factors				1
Majority Consensus factor	1	0	1	
Feedback Density factor	0	1	1	

Figure 4 depicts the analytical analysis of the trust results for a particular cloud service. From the figure, it can be seen that the higher the malicious behavior rate the lower the trust results are when considering to calculate the trust with all credibility factors. On the other hand, the trust results shows nearly no response to the malicious behavior rate when considering to calculate the trust without credibility factors. This demonstrates that our credibility model is robust and more sensitive in detecting malicious behaviors.

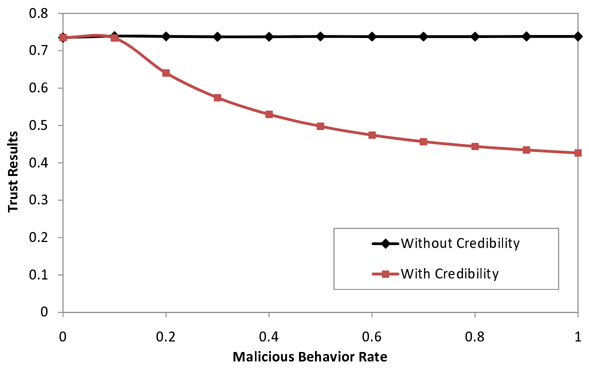


Fig. 4. Trust Robustness: With Credibility Vs. Without Credibility

Figure 5 shows the empirical analysis of the trust results for the same cloud service. It is clear that the trust results obtained by only considering the majority consensus factor are more accurate than the trust results obtained by only considering the feedback density factor when the malicious behavior rate is low (e.g., when the malicious behavior rate = 0.1, $\mathcal{T}r(s) = 0.59$ if we consider the majority consensus factor only while $\mathcal{T}r(s) = 0.73$ if we consider the feedback density factor only). This is true because there is still not many vicious cloud consumers (e.g., who submit lots of positive feedbacks in a short period in order to advantage a particular cloud service) during the trust aggregation. However, the trust results obtained by only considering the feedback density factor significantly response more when the malicious behavior rate become higher (e.g., when the malicious behavior rate = 0.9, $\mathcal{T}r(s) = 0.61$ for the majority consensus factor, $\mathcal{T}r(s) = 0.13$ for the feedback density factor). As a result, we can consider the majority consensus factor as a trust accuracy factor while the feedback density factor as a trust robustness factor (i.e., the feedback density factor is responsible for the robustness and the sensitiveness of our credibility model).

7 Conclusions and Future Work

Given the fact of the accelerated adoption of cloud computing in the recent years, there is a significant challenge in managing trust among cloud providers, service providers, and service requesters. In this paper, we present a trust management framework to manage trust in cloud environments. We introduce a credibility model that assesses cloud services' trustworthiness by distinguishing between credible trust feedbacks and amateur or malicious trust feedbacks. Also, the credibility model has the ability to detect the malicious trust feedbacks from attackers (i.e., who intend to manipulate the trust results by giving multiple trust feedbacks to a certain cloud service in a short period of time). We particularly introduce two trust parameters including the *Majority Consensus* factor and the *Feedback Density* factor in calculating the trust value of a cloud service.

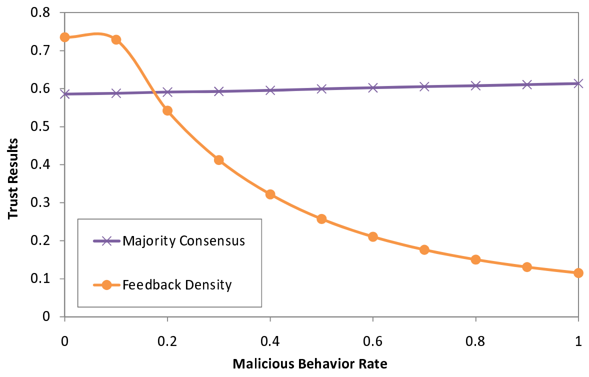


Fig. 5. Trust Robustness: Credibility Factors

In addition, our trust management service allows trust feedback assessment and storage to be managed in a distributed way.

In the future, we plan to deal with more challenging problems such as the *Sybil* attack and the *Whitewashing* attack. Performance optimization of the trust management service is another focus of our future research work.

References

1. Amazon: Amazon.com: Online shopping for electronics, apparel, computers, books, dvds & more (2011),`http://www.amazon.com/` (accessed March 01, 2011)
2. Armbrust, M., et al.: A View of Cloud Computing. Communiaction of the ACM 53(4), 50–58 (2010)
3. Aziz, B., Hamilton, G.: Detecting Man-in-the-Middle Attacks by Precise Timing. In: Proc. of the 3rd Int. Conf. on Emerging Security Information, Systems and Technologies (SECURWARE 2009). Athens, Glyfada, Greece (June 2009)
4. Ba, S., Pavlou, P.: Evidence of the Effect of Trust Building Technology in Electronic Markets: Price Premiums and Buyer Behavior. MIS Quarterly 26(3), 243–268 (2002)
5. Brandic, I., Dustdar, S., Anstett, T., Schumm, D., Leymann, F., Konrad, R.: Compliant Cloud Computing (C3): Architecture and Language Support for User-Driven Compliance Management in Clouds. In: Proc. of IEEE 3rd Int. Conf. on Cloud Computing (CLOUD 2010), Miami, Florida, USA (July 2010)
6. Buyya, R., Yeo, C., Venugopal, S.: Market-oriented Cloud Computing: Vision, Hype, and Reality for Delivering it Services as Computing Utilities. In: Proc. of IEEE 10th Int. Conf. on High Performance Computing and Communications (HPCC 2008), Dalian, China (September 2008)
7. Chen, K., Hwang, K., Chen, G.: Heuristic Discovery of Role-Based Trust Chains in Peer-to-Peer Networks. IEEE Transactions on Parallel and Distributed Systems 20(1), 83–96 (2008)
8. Child, I.: The Psychological Meaning of Aesthetic Judgments. Visual Arts Research 9(2(18)), 51–59 (1983)

9. Conner, W., Iyengar, A., Mikalsen, T., Rouvellou, I., Nahrstedt, K.: A Trust Management Framework for Service-Oriented Environments. In: Proc. of the 18th Int. Conf. on World Wide Web (WWW 2009), Madrid, Spain (April 2009)
10. Douceur, J.R.: The Sybil Attack. In: Druschel, P., Kaashoek, M.F., Rowstron, A. (eds.) IPTPS 2002. LNCS, vol. 2429, pp. 251–260. Springer, Heidelberg (2002)
11. eBay: ebay - new & used electronics, cars, apparel, collectibles, sporting goods & more at low prices (2011), `http://www.ebay.com/` (accessed March 01, 2011)
12. Friedman, E., Resnick, P., Sami, R.: Manipulation-Resistant Reputation Systems. In: Algorithmic Game Theory, pp. 677–697. Cambridge University Press, New York (2007)
13. Hoffman, K., Zage, D., Nita-Rotaru, C.: A Survey of Attack and Defense Techniques for Reputation Systems. ACM Computing Surveys (CSUR) 42(1), 1–31 (2009)
14. Hwang, K., Li, D.: Trusted Cloud Computing with Secure Resources and Data Coloring. IEEE Internet Computing 14(5), 14–22 (2010)
15. Jøsang, A., Ismail, R., Boyd, C.: A Survey of Trust and Reputation Systems for Online Service Provision. Decision Support Systems 43(2), 618–644 (2007)
16. Jøsang, A., Quattrociocchi, W.: Advanced Features in Bayesian Reputation Systems. In: Fischer-Hübner, S., Lambrinoudakis, C., Pernul, G. (eds.) TrustBus 2009. LNCS, vol. 5695, pp. 105–114. Springer, Heidelberg (2009)
17. Krautheim, F.J., Phatak, D.S., Sherman, A.T.: Introducing the Trusted Virtual Environment Module: A New Mechanism for Rooting Trust in Cloud Computing. In: Acquisti, A., Smith, S.W., Sadeghi, A.-R. (eds.) TRUST 2010. LNCS, vol. 6101, pp. 211–227. Springer, Heidelberg (2010)
18. Lai, K., Feldman, M., Stoica, I., Chuang, J.: Incentives for Cooperation in Peer-to-Peer Networks. In: Proc. of the 1st Workshop on Economics of Peer-to-Peer Systems, Berkeley, CA, USA (June 2003)
19. Malik, Z., Bouguettaya, A.: Rater Credibility Assessment in Web Services Interactions. World Wide Web 12(1), 3–25 (2009)
20. Malik, Z., Bouguettaya, A.: RATEWeb: Reputation Assessment for Trust Establishment Among Web services. The VLDB Journal 18(4), 885–911 (2009)
21. Manuel, P., Thamarai Selvi, S., Barr, M.E.: Trust Management System for Grid and Cloud Resources. In: Proc. of the 1st Int. Conf. on Advanced Computing (ICAC 2009), Chennai, India (December 2009)
22. Massa, P., Avesani, P.: Trust Metrics in Recommender Systems. In: Computing with Social Trust. Human-Computer Interaction Series, pp. 259–285. Springer, London (2009)
23. NetLogo: Netlogo home page (2011), `http://ccl.northwestern.edu/netlogo/` (accessed March 1, 2011)
24. Sheth, A.P., Gomadam, K., Lathem, J.: SA-REST: Semantically Interoperable and Easier-to-Use Services and Mashups. IEEE Internet Computing 11(6), 84–87 (2007)
25. Skopik, F., Schall, D., Dustdar, S.: Start Trusting Strangers? Bootstrapping and Prediction of Trust. In: Vossen, G., Long, D.D.E., Yu, J.X. (eds.) WISE 2009. LNCS, vol. 5802, pp. 275–289. Springer, Heidelberg (2009)
26. Skopik, F., Schall, D., Dustdar, S.: Trustworthy Interaction Balancing in Mixed Service-Oriented Systems. In: Proc. of ACM 25th Symp. on Applied Computing (SAC 2010), Sierre, Switzerland (March 2010)
27. Srivatsa, M., Liu, L.: Securing Decentralized Reputation Management Using TrustGuard. Journal of Parallel and Distributed Computing 66(9), 1217–1232 (2006)

28. Wang, Y., Vassileva, J.: Toward Trust and Reputation Based Web Service Selection: A Survey. International Transactions on Systems Science and Applications 3(2), 118–132 (2007)
29. Weng, J., Miao, C.Y., Goh, A.: Protecting Online Rating Systems from Unfair Ratings. In: Katsikas, S.K., López, J., Pernul, G. (eds.) TrustBus 2005. LNCS, vol. 3592, pp. 50–59. Springer, Heidelberg (2005)
30. Xiong, L., Liu, L.: Peertrust: Supporting Reputation-based Trust for Peer-to-Peer Electronic Communities. IEEE Transactions on Knowledge and Data Engineering 16(7), 843–857 (2004)