CLOUD ARMOR: Supporting Reputation-Based Trust Management for Cloud Services

Dasari Swapna1, J. Ramesh Babu Varugu2, B. Ravinder Reddy3

1 PG Scholar, Dept of CSE, Annamacharya Institute of Technology and Science, Hyderabad, TS, India,
E-mail: dasariswapna912@gmail.com.
2 Associate Professor & HOD, Dept of CSE, Annamacharya Institute of Technology and Science, Hyderabad, TS, India,
E-mail: hodeceact@gmail.com.
3 Assistant Professor, Dept of CSE, Annamacharya Institute of Technology and Science, Hyderabad, TS, India,
E-mail: p.satish605@gmail.com.

Abstract: In cloud computing growth, the management of trust element is most challenging issue. Cloud computing has produce high challenges in security and privacy by the changing of environments. Trust is one of the most concerned obstacles for the adoption and growth of cloud computing. Although several solutions have been proposed recently in managing trust feedbacks in cloud environments, how to determine the credibility of trust feedbacks is mostly neglected. In this project the system proposed a Cloud Armor, a reputation-based trust management framework that provides a set of functionalities to deliver Trust as a Service (Taas). “Trust as a Service” (TaasS) framework to improve ways on trust management in cloud environments. The approaches have been validated by the prototype system and experimental results.

Keywords: Cloud Computing, Credibility, Credentials, Trust Management, Reputation, Security, Privacy, Availability.

I. INTRODUCTION
The highly dynamic, distributed, and nontransparent nature of cloud services make the trust management in cloud environments a significant challenge. According to researchers at Berkeley, trust and security is ranked one of the top 10 obstacles for the adoption of cloud computing. Indeed, Service-Level Agreements (SLAs) alone are inadequate to establish trust between cloud consumers and providers because of its unclear and inconsistent clauses. Consumers’ feedback is a good source to assess the overall trustworthiness of cloud services. Several researchers have recognized the significance of trust management and proposed solutions to access and manage trust based on feedbacks collected from participants. In reality, it is not unusual that a cloud service experiences malicious behaviors (e.g., collusion or Sybil attacks) from its users. This paper focuses on improving trust management in cloud environments by proposing novel ways to ensure the credibility of trust feedbacks. In particular we distinguish the following key issues of the trust management in cloud environments: Consumers’ Privacy. The adoption of cloud computing raise privacy concerns. Consumers can have dynamic interactions with cloud providers, which may involve sensitive information. There are several cases of privacy breaches such as leaks of sensitive information (e.g., date of birth and address) or behavioral information (e.g., with whom the consumer interacted, the kind of cloud services the consumer showed interest, etc.).

Undoubtedly, services which involve consumers data (e.g., interaction histories) should preserve their privacy. Cloud Services Protection. It is not unusual that a cloud service experiences attacks from its users. Attackers can disadvantage a cloud service by giving multiple misleading feedbacks (i.e., collusion attacks) or by creating several accounts (i.e., Sybil attacks). Indeed, the detection of such malicious behaviors poses several challenges. Firstly, new users join the cloud environment and old users leave around the clock. This consumer dynamism makes the detection of malicious behaviors (e.g., feedback collusion) a significant challenge. Secondly, users may have multiple accounts for a particular cloud service, which makes it difficult to detect Sybil attacks. Finally, it is difficult to predict when malicious behaviors occur (i.e., strategic VS. occasional behaviors). Trust ManagementService’s Availability. A trust management service (TMS) provides an interface between users and cloud services for effective trust management. However, guaranteeing the availability of TMS is a difficult problem due to the unpredictable number of users and the highly dynamic nature of the cloud environment. Approaches that require understanding of users’ interests and capabilities through similarity measurements or operational availability measurements (i.e., uptime to the total time) are inappropriate in cloud environments. TMS should be adaptive and highly scalable to be functional in cloud environments.

II. EXISTING AND PROPOSED SYSTEMS

A. Existing System
According to researchers at Berkeley, trust and security is ranked one of the top 10 obstacles for the adoption of cloud computing. Indeed, Service-Level Agreements (SLAs). Consumers’ feedback is a good source to assess the overall
trustworthiness of cloud services. Several researchers have recognized the significance of trust management and proposed solutions to assess and manage trust based on feedbacks collected from participants.

B. Proposed System

Cloud service users’ feedback is a good source to assess the overall trustworthiness of cloud services. In this paper, we have presented novel techniques that help in detecting reputation based attacks and allowing users to effectively identify trustworthy cloud services. We introduce a credibility model that not only identifies misleading trust feedbacks from collusion attacks but also detects Sybil attacks no matter these attacks take place in a long or short period of time (i.e., strategic or occasional attacks respectively). We also develop an availability model that maintains the trust management service at a desired level. We also develop an availability model that maintains the trust management service at a desired level.

Advantages of Proposed System:
- Trust Cloud framework for accountability and trust in cloud computing. In particular, Trust Cloud consists of five layers including workflow.
- Propose a multi-faceted Trust Management (TM) system architecture for cloud computing to help the cloud service users to identify trustworthy cloud service providers.

Fig.1. System Architecture.

III. METHODOLOGIES

A. Detection of Service

This layer consists of different users who use cloud services. For example, a new startup that has limited funding can consume cloud services. Interactions for this layer include: i) service discovery where users are able to discover new cloud services and other services through the Internet, ii) trust and service interactions where users are able to give their feedback or retrieve the trust results of a particular cloud service, and iii) registration where users establish their identity through registering their credentials in IdM before using TMS.

B. Trust Communication

In a typical interaction of the reputation based TMS, a user either gives feedback regarding the trustworthiness of a particular cloud service or requests the trust assessment of the service 1. From users’ feedback, the trust behavior of a cloud service is actually a collection of invocation history records, represented by a tuple H=(C, S, F, T f), where C is the user’s primary identity, S is the cloud service’s identity, and F is a set of Quality of Service (QOS) feedbacks (i.e., the feedback represent several QOS parameters including availability, security, response time, accessibility, price).

C. IDM Registration

The system proposes to use the Identity Management Service (IdM) helping TMS in measuring the credibility of a consumer’s feedback. However, processing the IdM information can breach the privacy of users. One way to preserve privacy is to use cryptographic encryption techniques. However, there is no efficient way to process encrypted data. Another way is to use anonymization techniques to process the IdM information without breaching the privacy of users. Clearly, there is a trade-off between high anonymity and utility.

D. Service Announcement and Communication

This layer consists of different cloud service providers who offer one or several cloud services, i.e., IaaS (Infrastructure as a Service), PaaS (Platform as a Service), and SaaS (Software as a Service), publicly on the Web (more details about cloud services models and designs can be found). These cloud services are accessible through Web portals and indexed on Web search engines such as Google, Yahoo, and Baidu. Interactions for this layer are considered as cloud service interaction with users and TMS.

IV. IMPLEMENTATION AND EXPERIMENTAL EVALUATION

In this section, we report the implementation and experimental results in validating the proposed approach. Our implementation and experiments were developed to validate and study the performance of both the credibility model and the availability model.

A. System Implementation

The trust management service’s implementation is part of our large research project, named Cloud Armor, which offers a platform for reputation-based trust management of cloud services [10]. The platform provides an environment where users can give feedback and request trust assessment for a particular cloud service. Specifically, the trust management service (TMS) consists of two main components: the Trust Data Provisioning and the Trust Assessment Function.

1. The Trust Data Provisioning: This component is responsible for collecting cloud services and trust information. We developed the Cloud Services Crawler module based on the Open Source Web Crawler for Java (crawler4j) and extended it to allow the platform to automatically discover
cloud services on the Internet. We implemented a set of functionalities to simplify the crawling process and made the crawled data more comprehensive (e.g., add Seeds (), select Crawling Domain (), add Crawling Time ()). In addition, we developed the Trust Feedbacks Collector module to collect feedbacks directly from users in the form of history records and stored them in the Trust Feedbacks Database. Indeed, users typically have to establish their identities for the first time they attempt to use the platform through registering their credentials at the Identity Management Service (IdM) which stores the credentials in the Trust

2. Identity Registry: Moreover, we developed the Identity Info Collector module to collect the total number of established identities among the whole identity behavior (i.e., all established identities for users who gave feedbacks to a particular cloud service).

3. The Trust Assessment Function: This function is responsible for handling trust assessment requests from users where the trustworthiness of cloud services are compared and the factors of trust feedbacks are calculated (i.e., the credibility factors). We developed the Factors Calculator for attacks detection based on a set of factors (more details on how the credibility factors are calculated can be found). Moreover, we developed the Trust Assessor to compare the trustworthiness of cloud services through requesting the aggregated factors weights from the Factors Calculator to weigh feedbacks and then calculate the mean of all feedbacks given to each cloud service. The trust results for each cloud service and the factors’ weights for trust feedbacks are stored in the ‘Trust Results and Factors Weights Storage’.

B. Experimental Evaluation

We particularly focused on validating and studying the robustness of the proposed credibility model against different malicious behaviors, namely collusion and Sybil attacks under several behaviors, as well as the performance of our availability model.

C. Credibility Model Experiments

We tested our credibility model using real world trust feedbacks on cloud services. In particular, we crawled several review websites such as cloud-computing.findthebest.com, cloudstorageprovidersreviews.com, and Cloud Hosting Reviewer.com, and where users give their feedbacks on cloud services that they have used. The collected data is represented in a tuple $H$ where the feedback represents several QoS parameters as mentioned earlier and augmented with a set of credentials for each corresponding consumer. We managed to collect 10,076 feedbacks given by 6,982 users to 113 real-world cloud services. The collected dataset has been released to the research community via the project website. For experimental purposes, the collected data was divided into six groups of cloud services, three of which were used to validate the credibility model against collusion attacks, and the other three groups were used to validate the model against Sybil attacks where each group consists of 100 users. Each cloud service group was used to represent a different attacking behavior model, namely: Waves, Uniform and Peaks as shown in Fig.2. The behavior models represent the total number of malicious feedbacks introduced in a particular time instance (e.g., $|V(s)| = 60$ malicious feedbacks).

![Fig.2. Attacking Behavior Models](image)

when $T_f = 40$, Fig.2(a)) when experimenting against collusion attacks. The behavior models also represent the total number of identities established by attackers in a period of time (e.g., $|I(s)| = 78$ malicious identities when $T_f = 20$, Fig.2(c)) where one malicious feedback is introduced per identity when experimenting against Sybil attacks. In collusion attacks, we simulated malicious feedback to increase trust results of cloud services (i.e., self-promoting attack) while in Sybil attacks we simulated malicious feedback to decrease trust results (i.e., slandering attack). To evaluate the robustness of our credibility model with respect to malicious behaviors (i.e., collusion and Sybil attacks), we used two experimental settings: I) measuring the robustness of the credibility model with a conventional model $Cont(s, I_0, t)$ (i.e., turning $C_t(c, s, I_0, t)$ to 1 for all trust feedbacks), and II) measuring the performance of our model using two measures namely precision (i.e., how well TMS did in detecting attacks) and recall (i.e., how many detected attacks are actual attacks). In our experiments, TMS started rewarding cloud services that had been affected by malicious behaviors when the attacks percentage reached 25% (i.e., $et(s) = 25\%$), so the rewarding process would occur only when there was a significant damage in the trust result. We conducted 12 experiments where six of which were conducted to evaluate the robustness of our credibility model against collusion attacks and the rest for Sybil attacks. Each experiment is denoted by a letter from A to F, as shown in Table 1.

<table>
<thead>
<tr>
<th>TABLE I. Behavior Experimental Design</th>
</tr>
</thead>
<tbody>
<tr>
<td>Malicious Behaviors</td>
</tr>
<tr>
<td>---------------------</td>
</tr>
<tr>
<td>Collusion</td>
</tr>
<tr>
<td>Attacks</td>
</tr>
<tr>
<td>Sybil</td>
</tr>
<tr>
<td>Attacks</td>
</tr>
</tbody>
</table>

International Journal of Innovative Technologies
Volume.04, Issue No.13, September-2016, Pages: 2335-2340
1. Robustness against Collusion Attacks

For the collusion attacks, we simulated malicious users to increase trust results of cloud services (i.e., self promoting attack) by giving feedback with the range of [0.8, 1.0]. Fig.3 depicts the analysis of six experiments which were conducted to evaluate the robustness of our model with respect to collusion attacks. In Fig.3, A, B, and C show the trust result for experimental setting I, while A’, B’, and C’ depict the results for experimental setting II. We note that the closer to 100 the time instance is, the higher the trust results are when the trust is calculated using the conventional model. This happens because malicious users are giving misleading feedback to increase the trust result for the cloud service. On the other hand, the trust results show nearly no change when calculated using the proposed credibility model (Fig.3 A, B and C). This demonstrates that our credibility model is sensitive to collusion attacks and is able to detect such malicious behaviors. In addition, we can make an interesting observation that our credibility model gives the best results in precision when the Uniform behavior model is used (i.e., 0.51, see Fig.3 B’), while the highest recall score is recorded when the Waves behavior model is used (i.e., merely 0.9, see Fig.3 A’). Overall, recall scores are fairly high when all behavior models are used which indicate that most of the detected attacks are actual attacks. This means that our model can successfully detect collusion attacks (i.e., whether the attack is strategic such as in Waves and Uniform behavior models or occasional such as in the Peaks behavior model) and TMS is able to dilute the increased trust results from self-promoting attacks using the proposed credibility factors.

2. Robustness against Sybil Attacks

For the Sybil attacks experiments, we simulated malicious users to decrease trust results of cloud services (i.e., slandering attack) by establishing multiple identities and giving one malicious feedback with the range of [0, 0.2] per identity. Fig.4 depicts the analysis of six experiments which were conducted to evaluate the robustness of our model with respect to Sybil attacks. In Fig.4, D, E, and F show the trust results for experimental setting I, while D’, E’, and F’ depict the results for experimental setting II. From Fig.4, we can observe that trust results obtained by using the conventional model decrease when the time instance becomes closer to 100. This is because of malicious users who are giving misleading feedback to decrease the trust result for the cloud service. On the other hand, trust results obtained by using our proposed credibility model are higher than the ones obtained by using the conventional model (Fig.4 D, E and F). This is because the cloud service was rewarded when the attacks occurred. We also can see some sharp drops in trust results obtained by considering our credibility model where the highest number of drops is recorded when the Peaks behavior model is used (i.e., we can see 5 drops in Fig.4 F which actually matches the drops in the Peaks behavior model in Fig.2(c)). This happens because TMS will only reward the affected cloud services if the percentage of attacks during the same period of time has reached the threshold (i.e., which is set to 25% in this case). This means that TMS has rewarded the affected cloud service using the change rate of trust results factor.
CLOUD ARMOR: Supporting Reputation-Based Trust Management for Cloud Services

Fig. 5. Availability Prediction and Caching Accuracy

Moreover, from Fig.4 D', E' and F', we can see that our credibility model gives the best results in precision when the Waves behavior model is used (i.e., 0.47, see Fig.3 D'), while the highest recall score is recorded when the Uniform behavior model is used (i.e., 0.75, see Fig.3 A'). This indicates that our model can successfully detect Sybil attacks (i.e., either strategic attacks such as in Waves and Uniform behavior models or occasional attacks such as in the Peaks behavior model) and TMS is able to reward the affected cloud service using the change rate of trust results factor.

D. Availability Model Experiments

We tested our availability model using the same dataset we collected to validate the credibility model. However, for the availability experiments, we focused on validating the availability prediction accuracy, trust results caching accuracy, and reallocation performance of the availability model (i.e., to validate the three proposed algorithms including Particle Filtering based Algorithm, Trust Results & Credibility Weights Caching Algorithm, and Instances Management Algorithm).

1. Availability Prediction Accuracy

To measure the prediction accuracy of the availability model, we simulated 500 nodes hosting TMS instances and set the failure probability for the nodes as 3.5 percent, which complies with the findings. The motivation of this experiment is to study the estimation accuracy of our approach. We simulated TMS nodes’ availability fluctuation and tracked their fluctuation of availability for 100 time steps (each time step counted as an epoch). The actual availability of TMS nodes and corresponding estimated availability using our particle filter approach were collected and compared. Fig.5 (a) shows the result of one particular TMS node. From the figure, we can see that the estimated availability is very close to the actual availability of the TMS node. This means that our approach works well in tracing and predicting the availability of TMS nodes.

2. Trust Results Caching Accuracy

To measure the caching accuracy of the availability model, we varied the caching threshold to identify the optimal number of new trust feedbacks that TMS received to recalculate the trust result for a particular cloud service without having a significant error in the trust results. The trust result caching accuracy is measured by estimating the root-mean-square error (RMSE) (denoted caching error) of the estimated trust result and the actual trust result of a particular cloud service. The lower the RMSE value means the higher accuracy in the trust result caching. Fig.5 (b) shows the trust result caching accuracy of one particular cloud service. From the figure, we can see that the caching error increases almost linearly when the caching threshold increases. The results allow us to choose the optimal caching threshold based on an acceptable caching error rate. For example, if 10% is an acceptable error margin, the caching threshold can be set to 50 feedbacks. It is worth mentioning that the caching error was measured on real users’ feedbacks on real-world cloud services.

V. CONCLUSION

From this Cloud Armor Supporting Reputation-based Trust Management for Cloud Services has been implemented. In cloud computing growth, the management of trust element is
most challenging issue. Cloud computing has produce high challenges in security and privacy by the changing of environments. Trust is one of the most concerned obstacles for the adoption and growth of cloud computing. Although several solutions have been proposed recently in managing trust feedbacks in cloud environments, how to determine the credibility of trust feedbacks is mostly neglected. Additionally in future, we also enhance the performance of cloud as well as the security.

VI. REFERENCES


Author’s Profile: Dasari Swapna, PG Scholar, Dept of CSE, Annamacharya Institute of Technology and Science, Hyderabad, TS, India, E-mail: dasariswapna912@gmail.com.

Mr. Ramesh Babu Varugu, received the Master of Technology degree in Information Technology from the Gurunanak Institute Of Science Andtechnology-JNTUH, he received the Bachelor of Technology degree from Lakireddy Balireddy College of Engineering, JNTUK. He is currently working as Associate Professor and a Head of the Department of CSE with Annamacharya Institute Of Technology And Sciences, Hyderabad. His interest subjects are operating Systems, Cloud Computing and etc. Email: ramesh.vnl@gmail.com.

Mr. B. Ravinder Reddy, received the Master of Technology degree in Computer Science and Engineering from the Vasavya Engineering College-JNTUH, he received the Bachelor of Engineering degree from Jyothi Engineering College-JNTUH. He is currently working as Assistant Professor of CSE with Annamacharya Institute Of Technology And Sciences, Hyderabad. His interest subjects are Data Mining, Software Engineering and etc. Email: ravi@reddie@gmail.com.