View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by UUM Repository

Jurnal Teknologi

Full paper

Robust Multi-Dimensional Trust Computing Mechanism for Cloud Computing

Mohamed Firdhous[a*], Osman Ghazali[b], Suhaidi Hassan[b]

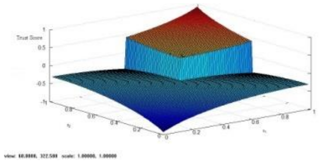
[a]*Faculty of Information Technology, University of Moratuwa, Moratuwa 10400, Sri Lanka*
[b]*InterNetWorks Research Lab, School of Computing, CAS, Universiti Utara Malaysia, 06010 UUM Sintok, Kedah, Malaysia*

*Corresponding author: Mohamed.Firdhous@uom.lk

Article history

Received :1 January 2014
Received in revised form :
15 February 2014
Accepted :18 March 2014

Graphical abstract



Abstract

Cloud computing has become the most promising way of purchasing computing resources over the Internet. The main advantage of .cloud computing is its economic advantages over the traditional computing resource provisioning. For cloud computing to become acceptable to wider audience, it is necessary to maintain the quality of service (QoS) commitments specified in the service level agreement. In this paper, the authors propose a robust multi-level trust computing mechanism that can be used to track the performance of cloud systems using multiple QoS attributes. In addition, tests carried out show that the proposed mechanism is more robust than the ones published in the literature.

Keywords: Cloud computing; quality of service; trust computing

© 2014 Penerbit UTM Press. All rights reserved.

■1.0 INTRODUCTION AND BACKGROUND

Electricity, water, gas and telephony are commonly known as utilities where the users are totally isolated from the nitty-gritty of the production process and pay only for the services they consume. Similarly cloud computing also makes the computing resources including infrastructure, development environment and applications available over the Internet and requires them to pay for the resources accessed. This has earned cloud computing the nick name "5th utility" [1].

Cloud systems have been hosted as virtual system on top of the physical hardware [2]. Thus hardware virtualization is the enabling technology for cloud computing. The virtual systems thus hosted The virtual machine manager installed on the bare metal hardware divides the physical hardware into multiple computing units either using the time division technology, space division or combination of both [3]. The space division virtualization technology assigns dedicated hardware such as CPU cores, memory and i/o devices to various processes, when available. On the other hand, time division virtualization technology divides all the hardware into multiple time slots and assigns them to different processes on a time shared basis [4]. These virtualized systems can be brought up and removed on demand [2]. Cloud computing services such as Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS) are hosted on top of the virtualized systems as shown in Figure 1.

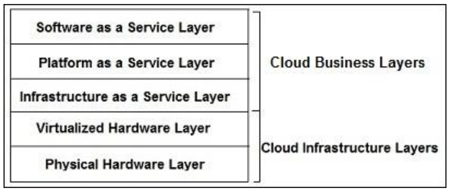

Software as a Service Layer
Platform as a Service Layer
Infrastructure as a Service Layer
Virtualized Hardware Layer
Physical Hardware Layer
Cloud Business Layers
Cloud Infrastructure Layers

Figure 1 Cloud computing layered model

In addition to the cloud computing business layers shown in Figure 1, different researchers and vendors have come up with other applications and solutions that are also marketed as services. These services include: Communication as a Service (CaaS), Data as a Service (DaaS), Network as a Service (NaaS) and Identity and Policy Management as a Service (IPaaS) are some of the other services that are available in the cloud arena, in addition to the cloud business services described earlier [5]. In addition, new services under new name have been introduced to the market daily by service providers. Some researchers have combined all these services under a single name XaaS-Anything as a Service [6].

The advantages of cloud computing over traditional computing can be easily explained by comparing the resource allocation patterns under both schemes. Figure 2 shows the capacity utilization curve developed by the Amazon Web Services (AWS) for a demand and allocation of storage capacity under cloud computing and traditional resources allocation schemes [7].

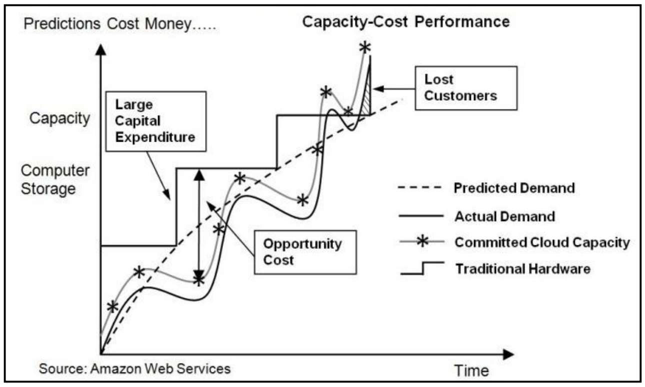


Figure 2 Capacity utilization curve [7]

Based on Figure 2, it can be seen that the actual demand for computer storage is not smooth but goes through fluctuations with ups and downs. The fluctuations in demand may be due to various reasons such as time of day, weekly or seasonal demand variations etc. In order to satisfy the changes in demand, under the traditional hardware provisioning scheme, it is required to invest on new hardware time to time as shown by step wise curve in the figure. Irrespective of how much is invested, traditional hardware provisioning cannot follow the demand pattern resulting in losses due to both under provisioning and over provisioning. On the hand, cloud computing based resource provisioning can closely follow the demand patterns during both short term as well as long term fluctuations. Hosting the resources on virtual platforms provides the cloud computing the ability to follow the demand changes as the virtual systems can be created and removed on the fly. When a virtual system has been removed, it releases all the resources that had been allocated for it, so that it can be allocated to another virtual system [8]. This helps the service provider to increase the utilization of the systems and profitability by allocating the same resources to multiple clients. On the other hand cloud computing benefits the clients by enabling them to pay only for the resources consumed and protecting them from resource starvation during high demand periods.

The attractiveness of cloud computing due to its efficiency and profitability, it has attracted many service providers [9]. These service providers host their services and make them available over the Internet for customers to access. The quality of services provided by these providers would heavily depend on the capacity of the physical resources and the number of clients accessing them concurrently. At the commencement of services, the service providers and the clients enter into a Service Level Agreement (SLA) that specifies conditions and commitments to be satisfied by both parties [10]. In these agreements, the Quality of Service (QoS) to be satisfied by the provider would occupy an important place [11]. Thus the quality of service of the service providers would play an important role in identifying the right service provider. QoS is characterized generally with the attributes such as response time, delay, service time and preferred values for these attributes. Also, the dynamic nature of cloud computing requires continuous monitoring of these attributes [10].

Due to the similarity and multi-faceted nature of trust and service quality, trust computing mechanisms can be used to quantify the QoS of cloud systems [12]. Several trust computing mechanisms based on different criteria and functions have been reported in the literature [13-19]. Though, these mechanisms are based on strong algorithms and functions, they mainly suffer from that shortcoming that they take only one input attribute for computing the trust score. Thus, the multi-faceted nature of trust as well as the user requirement for quantifying QoS on multiple attributes are totally ignored by these mechanisms. Hence, the practical use of these mechanism in a business cloud system is limited. In order to fill this shortcoming, the authors propose a multi-dimensional trust computing mechanism that incorporates statistical verification and non-linear hysteresis function. The robustness of the mechanism is enhanced by the statistical verification of the inputs and the non-linear hysteresis function in the events of short term temporary fluctuations and malicious attacks on the system [17-18].

This paper is into five main sections as follows: Section 1 provides the introduction and background information on the issues handled in the paper and the proposed solution. Section 2 critically analyzes the trust computing mechanisms proposed in the literature with special reference to their shortcomings. Section 3 introduces the proposed robust multi-dimensional trust computing mechanism for cloud computing. Section 4 describes the experimental setup used for testing the proposed mechanism along with an in depth analysis on the results. Finally Section 5 concludes the paper summarizing the findings with reference to the objectives set in Section 1.

■2.0 RELATED WORK

This section takes an in-depth look at the related studies carried out by other researchers and published in journals, conference proceedings and technical reports. A critical analysis is carried out on two main areas of interest: QoS in cloud computing and trust computing in distributed systems.

2.1 Quality of Service in Cloud Computing

Real world business cloud systems have been housed in large datacenters. These datacenters have large number of servers that have been installed with virtual machine managers in order to create even a larger set of virtual servers that can be brought up and removed on demand in an instant. The customer base of the large popular service providers is also large as they can easily attract them due to their previous track records [20].

Though cloud computing has taken the distributed systems market by storm, still there are many issues need to be addressed before completer acceptance of it by the user community [21]. One of the important issue that requires immediate attention is monitoring and management of QoS guarantees. The management of QoS in cloud computing becomes more complex compared to other distributed systems as cloud datacenters may host a diverse set of applications and systems possessing wide range of requirements [22]. For example, real time applications require faster response times and better throughputs and on the other hand non real time batch jobs are more concerned with accuracy and total processing times [23].

2.2 Trust Computing in Distributed Systems

Researchers in social sciences who studied the nature and behavior of human societies were initially interested in investigating the nature of trust and reputation [24]. Trust is a mental attitude for psychologists who investigate what happens in a human mind when one trusts or distrusts another [25]. Based on this notion, several cognitive trust models have been developed by researchers [26]. The sociologists study trust from the angle of social relationship between people in a community. This community relationship has been the foundation for building trust between different entities in multi agent systems and social networks [27]. Utility is the basis for studying trust by economists

[28]. All these studies carried out in diverse fields have enabled computer scientists to gain an in depth insight into human behavior under different circumstances and they have developed computational models based on them [29].

Trust and reputation systems have been developed and incorporated into various distributed systems including e-commerce, peer to peer networks, grid computing, semantic web, web services, and mobile networks [30]. These mechanisms and systems employ a well known mathematical function to compute the trust score for a given entity based on the results of transactions between two or more peers.

Chen and Ye have selected the fuzzy decision making for developing a trust computing mechanism for peer to peer computing systems [13]. The main advantages of this mechanism is the ability of handling of uncertainty and imprecision along with combining both direct trust and recommendation trust. The main shortcomings of this mechanism include the way trust is evolved initially using recommendation trust and then using direct trust as two distinct phases, the way the recommendation from multiple intermediaries are combined and computation of direct trust using a single parameter as input. Taking the recommendation trust and direct trust as distinct phases of trust computation makes the mechanism essentially single dimensional as they happen in sequence rather than taken together. The combination of recommendations by multiple intermediaries are carried out by taking the average value. No weight is given to the trustworthiness of the recommender, this makes the mechanism vulnerable to attacks by malicious nodes that spread false information. Hence this is a single dimensional trust computing mechanism.

The trust model proposed for P2P system by Tian *et al.* is based on recommendation evidence [14]. The proposed model has the advantage of modeling dynamic trust relationship using the aggregation of recommendation information. It also possesses the special capability of filtering out corrupted recommendation information. The downside of this model is that it takes only the recommendation information as the sole parameter for modeling trust. Hence, it is also essentially a single attribute based trust modeling system.

Dai *et al.* have proposed a trust computing mechanism employing the entropy function as the core for wireless sensor networks [15]. The main advantage of the proposed mechanism is the successful modeling of trust in an uncertain environment. Entropy is the measure of average uncertainty in a random variable. Also, the trust score computed reflects the results of the previous direct interactions of a given node with another. Hence, this is essentially a direct trust computing mechanism based on the nodes own experience. The main shortcoming of this mechanism is that it expects every node to have personal interactions with other nodes to build its own trust database and also the trust score computed is based on a single attribute, namely the success of failure of the previous interactions. Further, entropy is a monotonous function which changes its value for every input changes.

The trust computing mechanism proposed for cloud computing by Firdhous *et al.* in is based on a simple function that modifies the final score for every small change in the input [16]. The proposed mechanism is very simple but it can be easily exploited by the malicious attackers. Also, this one is also incapable of handling the user requirements based on multiple attributes.

The multilevel thresholding based trust computing mechanism proposed by Firdhous *et al.* in is also a single dimensional monotonous trust computing mechanism [17]. The main advantage of this algorithm is the modification of multiple related trust scores together when a change in a more stringent attribute occurs. The same advantage can be turned to disadvantage by a malicious attacker as the function used for computing the trust score is a monotonous one without any guard against momentary fluctuations.

The other hysteresis based trust computing mechanism proposed by Firdhous *et al.* in is more rugged in the events of malicious attacks and momentary fluctuations as the mathematical function used for computing trust is immune to these changes [18]. But this is also a single parameter based trust computing mechanism.

The memory-less trust computing mechanism proposed by Firdhous *et al.* in is very robust in the events of malicious attacks as the computed trust value does not depend on the previous interactions with any system [19]. But, this mechanism is also a single attribute based one as the mathematical function takes only one attribute as input.

Table 1 summarizes the trust computing mechanisms discussed above with respect to the functions used, their advantages and disadvantages. From the table, it can be seen that all these trust computing mechanisms are single dimensional ones incapable of handling multiple input parameters.

3.0 PROPOSED TRUST COMPUTING MECHANISM

Trust computing mechanism mainly concentrates on trust evolution where the trust scores are either improved or worsened based on the results of the interactions [31]. Figure 3 shows the block diagram of the trust computing system proposed in the paper. The trust computing unit and the QoS monitoring unit make the trust computing system. The cloud provider is external to the system, but provides the actual QoS information after every interaction.

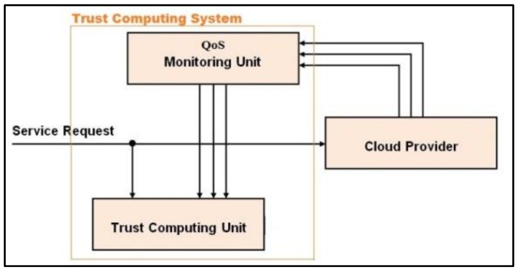


Figure 3 Trust computing system

When a client sign a service level agreement with a service provider, he or she also signs up with a trust provider who is independent of both the service provider and the client. The client provides the trust provider with a committed QoS values along with the weights and confidence level for each attribute depending on the stringency of the service quality required. When the client request reaches the service provider, it is also given to the trust computing system. The trust computing system, then extracts the expected QoS parameters and expected values (specified in the SLA) from its database for the particular request. When the service is completed, the QoS monitoring units follows the actual performance values and supplies them to the trust computing unit.

Table 1 Comparison of trust computing mechanisms

Paper	Mechanism	Function	Advantages	Disadvantages
[13])	Fuzzy decision making	Single input parameter, monotonous	Ability to handle uncertainty and imprecise information. Combines both direct and recommendation trusts.	Single attribute. No special weight for the trustworthiness of different recommenders. Vulnerable to attack.
[14]	Recommendation evidence	Single input parameter, monotonous	Models dynamic trust relationships between nodes. Has the ability to filter noisy recommendation information.	Single attribute.
[15]	Entropy based	Single input parameter, monotonous	Capable of modeling trust in uncertain environments.	Depends only on the direct interaction between nodes. Single attribute. Monotonously modifies the scores.
[16]	Incremental	Single input parameter, monotonous	Simple.	Single attribute. Vulnerable to attack.
[17]	Multi-level thresholding	Single input parameter, monotonous	Fast convergence as multiple trust scores are modified simultaneously.	Single attribute. Vulnerable to attack.
[18]	Hysteresis based	Single input parameter, hysteresis function	Robust in the events of attacks and momentary fluctuations.	Single attribute.
[19]	Memoryless	Single input parameter, Sigmoid function	Robust in the events of attacks and momentary fluctuations.	Single attribute.

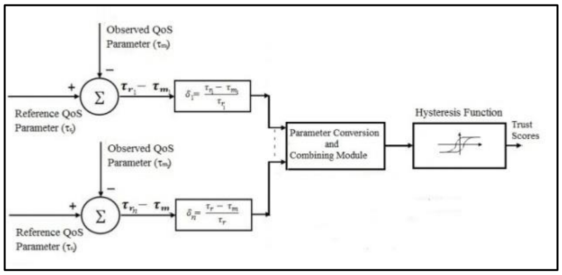

Observed QoS Parameter (τ_m)
+
−
Σ
$\tau_{r_1} - \tau_{m_1}$
$\delta_1 = \frac{\tau_{r_1} - \tau_{m_1}}{\tau_{r_1}}$
Reference QoS Parameter (τ_r)
Observed QoS Parameter (τ_m)
$\tau_{r_n} - \tau_m$
$\delta_n = \frac{\tau_r - \tau_m}{\tau_r}$
Reference QoS Parameter (τ_r)
Parameter Conversion and Combining Module
Hysteresis Function
Trust Scores

Figure 4 Trust computing unit

Figure 4 shows the trust computing unit in detail. The summer computes the difference between the actual value and the expected value for every attribute and supplies those differences to the next stage for computing the normalized attribute value. The normalization process removes any skewness in results due to the domination of a single attribute over the others. The parameter conversion and combining unit creates a single value by combining all the input parameters into a single value that can be supplied to the hysteresis function for computing the trust score.

The parameter conversion and combination is one of the main components of this mechanism that makes it multi-dimensional as opposed to all the other mechanisms. All the input parameters are converted to a single (combined) parameter as follows:

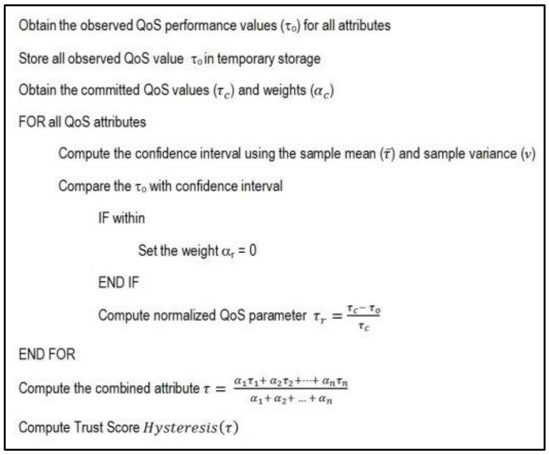
$$\tau = \frac{\alpha_1\tau_1 + \alpha_2\tau_2 + \ldots + \alpha_n\tau_n}{\alpha_1 + \alpha_2 + \ldots + \alpha_n} \qquad (1)$$

and

$$\alpha_1 + \alpha_2 + \ldots + \alpha_n = 1$$

where τ_r is the r^{th} parameter and$\alpha_r$is the weight applied to it.

The weights are selected depending on the importance of the parameter for the performance of the application. When an attribute does not play any role in the performance, its weight would be made equal to zero which essentially eliminates it from the trust computation process. Once the actual performance values (τ_o) are received, they are stored in the temporary storage for the purpose of computing the confidence interval. If the performance of any attribute falls within the confidence interval, the system performance is taken as satisfactory and eliminated from the computation of trust by making its weight (α) equal to zero. Figure 5 shows the trust computing algorithm employed in this mechanism.



```
Obtain the observed QoS performance values ($\tau_o$) for all attributes
Store all observed QoS value $\tau_o$ in temporary storage
Obtain the committed QoS values ($\tau_c$) and weights ($\alpha_c$)
FOR all QoS attributes
    Compute the confidence interval using the sample mean ($\bar{\tau}$) and sample variance ($v$)
    Compare the $\tau_o$ with confidence interval
        IF within
            Set the weight $\alpha_r = 0$
        END IF
        Compute normalized QoS parameter $\tau_r = \frac{\tau_c - \tau_o}{\tau_c}$
END FOR
Compute the combined attribute $\tau = \frac{\alpha_1\tau_1 + \alpha_2\tau_2 + \cdots + \alpha_n\tau_n}{\alpha_1 + \alpha_2 + \ldots + \alpha_n}$
Compute Trust Score $Hysteresis(\tau)$
```

Figure 5 Trust computing algorithm

■4.0 RESULTS AND DISCUSSION

The proposed mechanism was its functionality and accuracy with simulations. The simulation environment was created with Mat lab by creating every functional unit, independently and combining them together to form the complete system. The hysteresis function in the trust computing unit was constructed as follows:

$$hysteresis(x) = \begin{cases} sigm\,(x - k) & for\ x_n > x_{n-1} \\ sigm(x + k) & for\ x_n < x_{n-1} \end{cases} \qquad (2)$$

where k - is the horizontal shift and

$$sigm(x) = \frac{1 - e^{-x}}{1 + e^{+x}}$$

$sigm(x)$ is known as the sigmoid function that has an odd symmetry about the y-axis. The hysteresis loop thus created is shown in Figure 6.

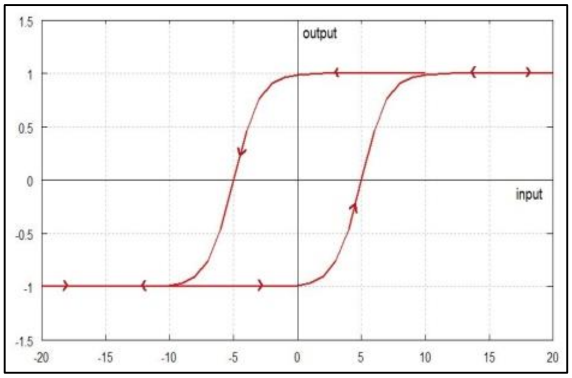


Figure 6 Hysteresis loop

Figure 7 shows the trust scores computed using two attributes along with the effect of weights applied on the input parameters. From the figure, it can be seen that the final trust score is more aligned towards the parameter that is applied a higher weight.

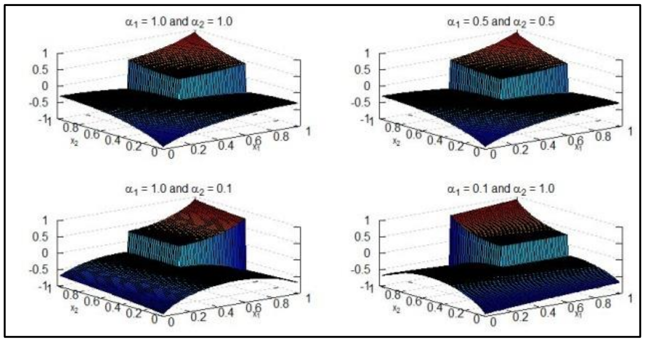


Figure 7 Effect of multiple attributes on trust score

Figure 8 shows the trust values computed using the proposed mechanism along with that of the entropy based mechanism. The proposed mechanism was also tested using statistically validated (@95%) inputs and non validate inputs. The statistically validation checks if the change in the attribute is due to a temporary fluctuation or due to system degradation. If the observed input value falls within the confidence interval, it was taken as a temporary fluctuation and the effect of the attribute on the trust score was eliminated by making the weight (α) equal to zero. This way, if all the QoS attributes fall within their respective confidence intervals, then the trust score will not be modified from the previous value as there is no observable change in performance. From Figure 8, it can be seen that the performance of the proposed mechanism is better and subject to less fluctuations compared to the entropy based mechanism proposed by Dai *et al.* in [15]. Also it could be seen that when the statistically validated input is applied to the proposed mechanism it shows more robust performance as small fluctuations in the performance is suppressed by the statistical validation process.

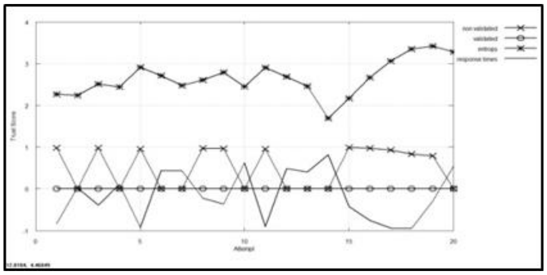


Figure 8 Comparison of trust scores computed

Figure 9 shows the effect of the confidence level on the trust scores computed. From this figure, it could be seen that the trust scores computed using 90% confidence level shows more fluctuations than the one computed using 95% confidence level. This is due to the reason that at 95% confidence level, the expectation of the client on performance is more stringent.

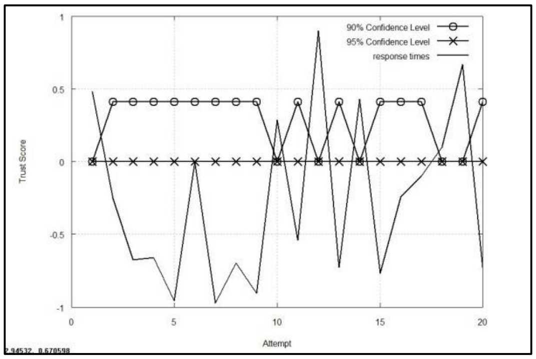


Figure 9 Effect of confidence level on trust score

Hence it can be concluded that the proposed mechanism performs better and more robust than the entropy based mechanism in the events of temporary fluctuations. Also it cannot be attacked by adversaries by continuous bombardments. Figure 10 shows trust scores computed using the same methods when the fluctuations are large. From Figure 10, it can be seen that when the fluctuations are large trust scores show the same performance for both validated and non-validated inputs. This is due to the reason that when the fluctuations are large, they are due to actual system degradation than temporary ones.

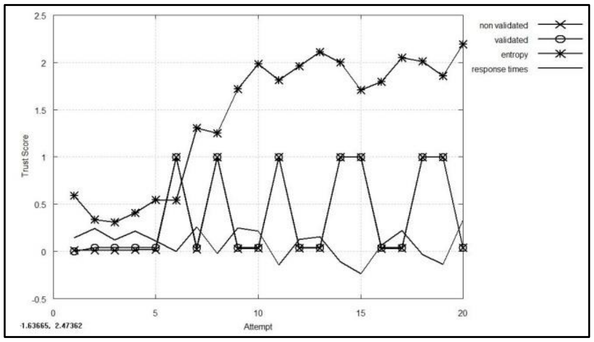


Figure 10 Effect of large fluctuations on trust scores

■5.0 CONCLUSION

In this paper, the authors presented a robust multi-dimensional trust computing mechanism that can track the performance of a cloud system using more than on QoS parameter. The mechanisms proposed in the literature so far are all single dimension as they compute the trust score using only one input parameter. More over the proposed mechanism shows more robust performance than the ones that are implemented using monotonously changing functions. When the proposed mechanism is equipped with additional statistical validation of inputs, its performance becomes better due to double protection provided by statistical validation and hysteresis loop both are immune to small changes in inputs.

References

[1] Buyya, R., C. S Yeo, S. Venugopal, J. Broberg, and I. Brandic. 2009. Cloud Computing and Emerging IT Platforms: Vision, Hype and Reality for Delivering Computing as the 5th Utility. *Journal of Future Generation Computer Systems*. 25(6): 599–616.

[2] Siddhisena, B., L. Warusawithana, and M. Mendis. 2011. Next Generation Multi-Tenant Virtualization Cloud Computing Platform. *Proceedings of the 13th International Conference on Advanced Communication Technology*. Seoul, South Korea.

[3] Semnanian, A. A., J. Pham, B. Englert, and X. Wu. 2011. Virtualization Technology and Its Impact on Computer Hardware Architecture. *Proceedings of the 8th International Conference on New Generation Information Technology*. Las Vegas, NV, USA.

[4] Zaman, S. and D. Grosu. 2010. Combinatorial Auction-Based Allocation of Virtual Machine Instances in Clouds. *Proceedings of the 2nd IEEE International Conference on Cloud Computing Technology and Science*. Indianapolis, IN, USA.

[5] Zhou, M., R. Zhang, D. Zeng and W. Qian. 2010. Services in the Cloud Computing Era: A Survey. *Proceedings of the 4th Fourth International Universal Communication Symposium*. Beijing, China.

[6] Rao, M. and S. Vijay. 2009. Cloud Computing and the Lessons from the Past. *Proceedings of the 18th IEEE International Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises*. Groningen, The Netherlands.

[7] AWS. 2012. Capacity Utilization Curve. *Amazon Web Services Economics Center*. Retrieved on 05/01/2012 from http://aws.amazon.com/economics/.

[8] Zhang, B., X. Wang, R. Lai, L. Yang, Y. Luo, X. Li, X and Z. Wang. 2010. A Survey on I/O Virtualization and Optimization. *Proceedings of the 5th Annual ChinaGrid Conference*. Guangzhou, China.

[9] Rimal, B. P., E. Choi and I. Lumb. 2009. A Taxonomy and Survey of Cloud Computing Systems. *Proceedings of the 5th International Joint Conference on INC, IMS and IDC*. Seoul, Korea.

[10] Patel, P., A. Ranabahu and A. Sheth. 2009. Service Level Agreement in Cloud Computing. *Proceedings of the ACM SIGPLAN International Conference on Object- Oriented Programming, Systems, Languages, and Applications*. Orlando, FL, USA.

[11] Wu, L. and R. Buyya. 2012. Service Level Agreement in Utility Computing Systems. In V. Cardellini, E. Casalicchio, K. Castelo Branco, J. Estrella, and F. Monaco (Eds.). *Performance and Dependability in Service Computing: Concepts, Techniques and Research Directions*. Information Science Reference.

[12] Firdhous, M., O. Ghazali, S. Hassan, N. Z. Harun and A. Abas. 2011. Honey Bee Based Trust Management System for Cloud Computing. *Proceedings of the 3rd International Conference on Computing and Informatics*. Bandung, Indonesia.

[13] Chen, H. and Z. Ye. 2008. Research of P2P Trust Based on Fuzzy Decision Making. *Proceedings of the 12th International Conference on Computer Supported Cooperative Work in Design*. Xi'an, China.

[14] Tian, C. Q., S. H. Zou, W. D. Wang and S. D. Cheng. 2008. A New Trust Model Based on Recommendation Evidence for P2P Networks. *Chinese Journal of Computers*. 31(2): 270–281.

[15] Dai, H., Z. Jia and X. Dong. 2008. An Entropy-Based Trust Modeling and Evaluation for Wireless Sensor Networks. *Proceedings of the International Conference on Embedded Software and Systems*. Chengdu, Sichuan, China.

[16] Firdhous, M., O. Ghazali and S. Hassan. 2011. A Trust Computing Mechanism for Cloud Computing. *Proceedings of the 4th ITU Kaleidoscope Academic Conference*. Cape Town, South Africa.

[17] Firdhous, M., O. Ghazali and S. Hassan. 2011. A Trust Computing Mechanism for Cloud Computing with Multilevel Thresholding. *Proceedings of the 6th International Conference on Industrial and Information Systems (ICIIS2011)*. Kandy, Sri Lanka.

[18] Firdhous, M., O. Ghazali and S. Hassan. 2012. Hysteresis-Based Robust Trust Computing Mechanism for Cloud Computing. *Proceedings of the IEEE Region 10 Conference*. Cebu, the Philippines.

[19] Firdhous, M., O. Ghazali and S. Hassan. 2012. A Memoryless Trust Computing Mechanism for Cloud Computing. *Proceedings of the 4th International Conference on Networked Digital Technologies*. Dubai.

[20] Garg, S. K., S. K. Gopalaiyengar and R. Buyya. 2011. SLA-Based Resource Provisioning for Heterogeneous Workloads in a Virtualized Cloud Datacenter. *Proceedings of the 11th International Conference on Algorithms and Architectures for Parallel Processing*. Melbourne, Australia.

[21] Yeo, C. S. and R. Buyya. .2005. Service Level Agreement Based Allocation of Cluster Resources: Handling Penalty to Enhance Utility. *Proceedings of the 7th IEEE International Conference on Cluster Computing*. Boston, MA, USA.

[22] Quiroz, A., H. Kim, M. Parashar, N. Gnanasambandam and N. Sharma. 2009. Towards Autonomic Workload Provisioning for Enterprise Grids and Clouds. *Proceedings of the 10th IEEE/ACM International Conference on Grid Computing*. Banff, AL, Canada.

[23] Carrera, D., M. Steinder, I. Whalley, J. Torres and E. Ayguade. 2008. Enabling Resource Sharing between Transactional and Batch Workloads using Dynamic Application Placement. *Proceedings of the 9th ACM/IFIP/USENIX International Conference on Middleware*. Leuven, Belgium.

[24] Yu, H., Z. Shen, C. Miao, C. Leung and D. Niyato. 2010. A Survey of Trust and Reputation Management Systems in Wireless Communications. *Proceedings of the IEEE*. 98(10): 1755–1772.

[25] McKnight, D. H. and N. L. Chervany. 2001. Conceptualizing trust: A Typology and E-commerce Customer Relationships Model. *Proceedings of the 34th Hawaii International Conference on System Sciences*. Island of Maui, HI, USA.

[26] Wang, W. and G. S. Zeng. 2010. Bayesian Cognitive Trust Model Based Self-Clustering Algorithm for MANETs. *Science China Information Sciences*. 53(3): 494–505.

[27] Gan, Z., J. He and Q. Ding. 2009. Trust Relationship Modeling in E-Commerce-Based Social Network. *Proceedings of the International Conference on Computational Intelligence and Security*. Beijing, China.

[28] Menkes, R. A. 2007. An Economic Analysis of Trust, Social Capital and the Legislation of Trust. *LLM Thesis*. University of Ghent, Ghent, Belgium.

[29] Mui, L. 2002. Computational Models of Trust and Reputation: Agents, Evolutionary Games, and Social Networks. *PhD Thesis*. Massachusetts Institute of Technology, Cambridge, MA, USA.

[30] Momani, M. and S. Challa. 2010. Survey of Trust Models In Different Network Domains. *International Journal of Ad hoc, Sensor and Ubiquitous Computing*. 1(3): 1–19.

[31] Abari, A. S. and T. White. 2012. DART: A Distributed Analysis of Reputation and Trust Framework. *Computational Intelligence*. 28(4): 642–682.