2015 International Conference on Cloud and Autonomic Computing

Monitoring and Management of Service Level Agreements in Cloud Computing

Anithakumari S
Dept. of CSE, NITK Surathkal
Karnataka, India
Email: lekshmi03@gmail.com

K Chandrasekaran
Dept. of CSE, NITK Surathkal
Karnataka, India
Email:kch@nitk.ac.in

***Abstract*—Cloud computing environment consists of various interactive entities like cloud service providers, cloud service brokers, cloud customers and end-users with different objectives and expectations. Service Level Agreements (SLAs) manage the relationship among cloud service providers and cloud consumers by defining the terms of the agreement for the participating entities and provide the basic ground for interactions among both the parties. In this work we proposed a framework to efficiently monitor and analyze the SLA parameters and tried to find out the possibility of occurrence of SLA violations. Also we implemented an adaptive resource allocation system by utilizing the results of predicted SLA violations. Our adaptive resource allocation system allocates computing resources to cloud applications and tries to reduce the occurrence of SLA violations, by allocating additional resources on the detection of possibility of occurrence of a violation. The experimental studies show that our proposed system works well in private cloud computing environment and gives more efficient results.**

I. Introduction

Cloud computing is a newly emerging paradigm of computing in which customers can use highly scalable computing resources or services according to their requirements in a pay-per use basis. The provisioning of cloud services such as: applications, platforms and infrastructure (storage, processing power and network bandwidth) has been done by cloud providers based on a predefined set of properties called quality of service (QoS). The negotiation of QoS properties has been done through an agreement called Service Level Agreement (SLA). SLA is viewed as a legal contract between cloud providers and customers to assure that QoS properties of the cloud services are met. If any violation happens in QoS the defaulter has to pay penalty as per the terms specified in the SLA. SLAs play a major role in defining the expectations of both the parties by specifying the nature and type of the service.

Reliable monitoring and management of SLA is equally important to both cloud providers and cloud consumers since it is the dominant mechanism to maintain healthy business between both the parties. SLA monitoring can contribute more to guarantee the Quality of service (QoS) of cloud applications. SLA monitoring system actually detects the SLA objective violations and helps to provision the resources and services in an adaptive manner.

In this work, we proposed an architecture for detecting SLA violations and used the results from this violation detection system to develop adaptive resource allocation which will dynamically allocate resources and services according to the changing requirements in the environment. Here the possibility of SLA violations are predicted (ie, before the occurrence of violation) by monitoring the run time data and we use these predictions to initiate adaptive provisioning of the cloud service. This preventive and adaptive approach contributes high performance and efficiency to cloud service provisioning. The adaptive resource allocation system proposed here uses the method of early detection of SLA violations and it yields a better allocation for almost 75 % of the total cases. The remaining sections of the paper is organized as: Section II describes the related works in the field of SLA monitoring and adaptive approaches. Section III presents the SLA management mechanism and in section IV we present adaptive resource management through SLA. In section V implementation and results are discussed and finally section VI concludes the paper.

II. Related Work

SLAs are treated as a medium for specifying the required levels of performance and it also assures the provisioning of computing resources negotiated initially among various cloud providers. Research works like QoSMONaaS [1] and SLAMonADA [2] present SLA monitoring system to detect the SLA violations, but no major focus has given towards the reactive actions. In [3] Koller and Schubert used a proxy-like approach to implement autonomic QoS management in web service based agreement and SLAs are used to define QoS parameters. Frutos and Kotsiopoulos [4] discuss the concept of a framework for computational Grids using the project BREIN [5]. BREIN project [5] explores the management of SLAs in Grids and not in clouds. A. Kertesz et. al. [6] and P. Varalakshmi et. al. [7] propose some systems to detect the violations by monitoring SLA. They tried the detection of SLA violations for computing the penalty the service provider has to give to the service consumer on each violation. The resource allocation mechanisms discussed in papers like [8], [9] focus job scheduling/ task scheduling techniques for allocation of resources. In [10] the resource allocation is organized by considering criteria like geographical distance and workload of the data center. But none of the works in these existing literature have attempted to relate the resource allocation and SLA violation. Here in this work our main focus is to detect the possibility of SLA violations and reduce the occurrence of violation using some adaptive techniques in resource allocation.

978-1-4673-9566-3/15 $31.00 © 2015 IEEE
DOI 10.1109/ICCAC.2015.28

III. SLA MANAGEMENT

SLA management system enforces the agreement terms specified in the SLA by going through several steps, in accordance with the various phases of the SLA life cycle. The initial step is establishment of an SLA, where both cloud service provider and consumer do negotiations on significant SLA parameters by considering needed guarantees and third party or broker services. After the negotiation and establishment of a standard SLA, the SLA deployment component will take care of the checking and monitoring of the SLA. The quality dimension of an SLA is visible as an SLO (Service Level Objective), and the penalty to SLA violation is for the non-attainment of SLOs. In our SLA monitoring engine the real values of SLOs are monitored at runtime and the values are measured in a per-instance or per-aggregated manner. Figure 1 illustrates the domain model of SLA monitoring where both per instance and per aggregated mode of monitoring are used collectively to get SLO values. To clarify the concept, here we present a set of SLOs (Table I) by assuming the SLA of an online shopping service as a sample. Here we have

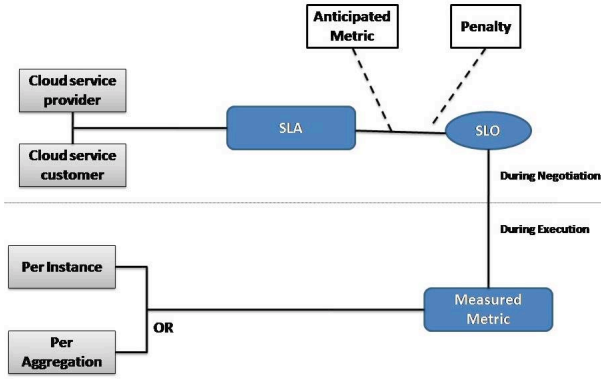

Fig. 1. Domain model of SLA

Type of SLO	SLO	Anticipated metric	Penalty	Interval
Instance level	Time to order	$\leq$ 36Hrs	3% discount	Not applicable
	Time to delivery	$\leq$ 4 days	5% discount	Not Applicable
	Delay in end to end transaction	$\leq$7days	10% discount	Not Applicable
Aggregated	Quality of good	High	80% discount	Not Applicable
	Failed service request rate	$\leq$1%	20% discount on next purchase	Bi-weekly
	Availability of service rate	$\geq$ 99%	20% discount on next purchase	Monthly

TABLE I. SAMPLE SLO

presented a runtime approach for detecting SLA violation based on event monitoring. Runtime detection and prediction is appreciative because it yields high effect in adaptive resource allocation. To materialize detection of SLA violation, runtime data is monitored and analyzed thoroughly to trigger variations in SLOs. The provision to perform monitoring, analysis and verification are included in the prototype implementation of our architecture.

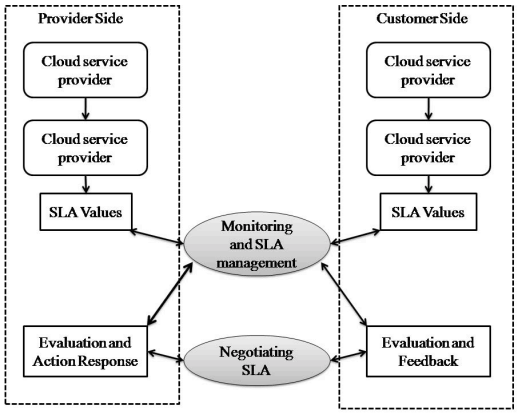

Fig. 2. SLA management system

The overall architecture of our SLA monitoring and management system is given in figure 2. The detailed view of SLA monitoring and management system is shown in figure 3. The

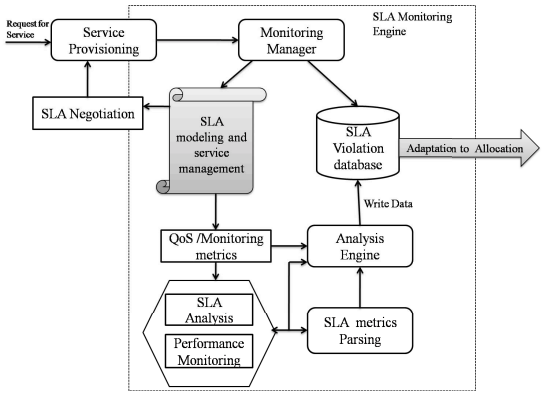

Fig. 3. SLA Monitoring engine

monitoring engine mainly contains two different phases: the runtime monitoring of SLA parameters and the analysis and verification of SLOs. The runtime monitoring is coordinated by the monitoring manager which continuously monitor the application performance. The service management unit extracts the parameters corresponding to QoS terms specified in the SLA and made it available to the next unit, the analysis engine. The analysis engine verifies the values through a matching between the needed value, ie., the threshold limit, and measured values. If the measured value exceeds the threshold limit then it is identified as a violation and the corresponding data values are passed to the violation database for next level processing. The data values contained in the violation database are decoded properly for making adaptive decisions at the time of adaptive resource management. For making adaptive actions in resource management a separate database is maintained in the resource allocation unit, called database for adaptive actions, which contain the list of all possible adaptive actions.

IV. ADAPTIVE RESOURCE MANAGEMENT THROUGH SLA

Monitoring and analysis of SLA parameters gives a clear picture of the timely requirements of the service consumer

and if the resource allocation system is capable of accommodating the changes dynamically according to the changing expectations of the user, then the resource management system becomes more adaptive and effective. Our work focusses on the management of resource allocation by providing additional images to executing jobs, which reduces the number of SLA violations in an opennebula cloud environment. The algorithm we developed for implementing adaptive resource management in cloud computing is depicted here.

Algorithm 1 Detection of SLA violation

Input: Monitored SLA parameters.
Output: Flag status of SLA violation.
1. Start
2. Let *response time* and *job execution time* be two SLO values. Let *flag value* be a boolean value for controlling the trigger signal in adaptive resource allocation.
3.
a. Initialize SLA violation limit and SLA threat limit.
b. Reset the *flag value* to zero.
4.
a. Monitor SLA parameters using monitoring tool.
b. Measure SLO values like *response time*, *job execution time* and *number of resources*.
5. Compare the measured SLO values with SLA threat limit.
6. If the SLO value $\geq$ threat limit then
a. Detect it as a possibility of SLA violation.
b. Set the value of *flag* to 1.
c. Give trigger signal to dynamic resource allocation.
7. Stop

Algorithm 2 Adaptive Resource Allocation

Input: Computing resources like CPU, RAM and Memory.
Output: Computing resources after allocation.
1. Start
2. Read the availability of resources like CPU, RAM, Memory, VM image, Network etc.
3. Allocate normal resources to the job.
4. Check whether any trigger signal at adaptive resource allocation.
5. If yes
a. submit resource request to resource provisioning unit.
b. Release and allocate extra resources by providing VM images.
c. Update estimation of job execution time, response time and resource availability.
d. Convey back to SLA negotiator and resource provider about the resource allocation/re-allocation.
6. Stop

In Algorithm 1, input is the measured SLA values received through the monitoring tools and output is a *flag value* showing the status of SLA violation. The trigger signal used in Algorithm 2 for controlling adaptive resource allocation is operated by this *flag value*. The *flag value* represents the possibility of SLA violation as: if the *flag* is set, the violation possibility is detected and if the flag value is not changed, there is no possibility of violation. The measured SLO values like 'response time' and 'job execution time' are compared with *threat threshold limits* of violation and if the SLO values are going beyond the threshold limits, it is identified as a possibility of violation. Algorithm 2 takes the available resources like CPU, RAM, memory, VM (Virtual Machine) images etc. as the input and gives the same set of resources (after utilization) as the output. For allocating additional resources to jobs a request is given to the resource provisioning unit, which is in charge of resource allocation and re-allocation. After allocation the resource availability is re-estimated and conveyed back to SLA negotiator and Resource provider units. Time complexity of both algorithms is O(n) where 'n' is the number of jobs executed .

V. Experimental Analysis

Results are derived from a private cloud environment set up using opennebula and some monitoring tools like Ganglia. The private cloud setup comprise of multiple host machines which are controlled by a front-end controller where each host is capable of generating multiple VM images. The SLA parameter considered here for detection of SLA violations is response time. The experiments are conducted on a private cloud set up using opennebula 4.6. Computing services are hosted in virtual machines of multiple host machines and are submitted, monitored and controlled by using the GUI package Sunstone. Runtime monitoring and measurement have done by GMOND module provided by Ganglia [11] monitoring tool. The ganglia tool is installed on the front end controller which serves as cloud provider for the setup. The measured metrics are forwarded to event stream processing where Esper engine [12] is utilized for event processing. The metrics and events are passed among different units using Java Messaging Service(JMS). SLA parameters and corresponding SLO values are extracted from the document containing negotiated SLA using an XML parser. Database management is established using MySQL database. We fixed the threshold limit, for the SLA parameter (response time) as 2 sec, for SLA violation and if the measured parameter values are going beyond this violation threshold limit, it is detected as a violation. Similarly for prediction we fixed the threat threshold limit as 1.7 sec.

A. Results and Analysis

The preliminary results obtained for detection and prediction of SLA violation is shown in figure 4. The detection module counted the number of violations and predictions for several days and all the three parameters are depicted in the graph. The obtained results and the analysis on these results are as follows: If there is no SLA violation identified means all hosts have enough computing resources to process their applications without compromising the quality of service. But if there is any violation or possibility of violation it is purely an indication of the compromise on quality of service. For our proposed module, number of SLA violations obtained is between 20 to 25 and the number of predictions is 45 to 50 per month on an average. From this results, it is clear that our detection module has succeeded in predicting majority of SLA violations. The results obtained for our adaptive resource management system is shown in figure 5. The job execution time of 4 different jobs, allocated to different VMs, with adaptive resource allocation and without adaptive resource allocation are shown. Our adaptive resource allocation works

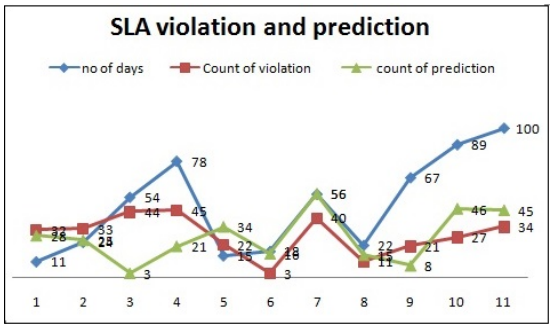

Fig. 4. SLA violation and prediction

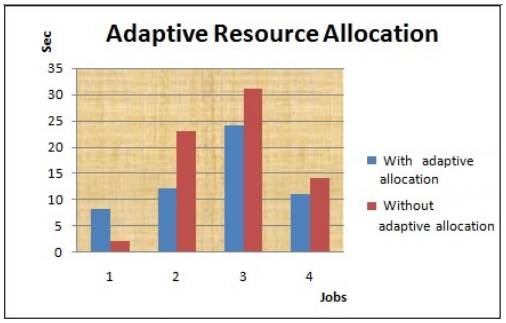

Fig. 5. Adaptive Resource allocation

well by taking less time for job completion in three cases and in one case it is taking more time than the normal approach. That is if we generalize the criteria, our algorithm is giving better performance for around 75 % of the total cases. The overall results show that our SLA associated adaptive resource management system works well in private cloud environment and generates more efficiency to resource allocation and processing.

VI. Conclusion

Defining proper SLA and maintaining SLA management systems are the major things to take care for assuring the quality requirements while achieving business in cloud computing. The business goals and performance are characterized by QoS and other contextual information in cloud systems. In this work we monitored the SLA parameters and detected the possibility of occurrence of SLA violation. The resource allocation mechanism is controlled properly by using these results and done well in reducing the number of SLA violations. This SLA associated adaptive resource allocation surely improved the efficiency of resource allocation in the private cloud computing environment set up using opennebula. The attraction of our proposed model is it gives attention to both cloud providers and consumers and helps to continue the business transactions for a long time.

References

[1] G. Cicotti, S. DAntonio, R. Cristaldi, and A. Sergio, "How to monitor qos in cloud infrastructures: The qosmonaas approach," in *Intelligent Distributed Computing VI*. Springer, 2013, pp. 253–262.

[2] C. Muller, M. Oriol, M. Rodríguez, X. Franch, J. Marco, M. Resinas, and A. Ruiz-Cortes, "Salmonada: A platform for monitoring and explaining violations of ws-agreement-compliant documents," in *Principles of Engineering Service Oriented Systems (PESOS), 2012 ICSE Workshop on*. IEEE, 2012, pp. 43–49.

[3] B. Koller and L. Schubert, "Towards autonomous sla management using a proxy-like approach," *Multiagent and Grid Systems*, vol. 3, no. 3, pp. 313–325, 2007.

[4] H. M. Frutos and I. Kotsiopoulos, "Brein: Business objective driven reliable and intelligent grids for real business." *IBIS*, vol. 8, pp. 39–41, 2009.

[5] M. Comuzzi, C. Kotsokalis, G. Spanoudakis, and R. Yahyapour, "Establishing and monitoring slas in complex service based systems," in *Web Services, 2009. ICWS 2009. IEEE International Conference on*. IEEE, 2009, pp. 783–790.

[6] A. Kertesz, G. Kecskemeti, and I. Brandic, "Autonomic sla-aware service virtualization for distributed systems," in *Parallel, Distributed and Network-Based Processing (PDP), 2011 19th Euromicro International Conference on*. IEEE, 2011, pp. 503–510.

[7] P. Varalakshmi, K. Priya, J. Pradeepa, and V. Perumal, "Sla with dual party beneficiality in distributed cloud," in *Advances in Computing and Communications*. Springer, 2011, pp. 471–479.

[8] M. Stillwell, D. Schanzenbach, F. Vivien, and H. Casanova, "Resource allocation using virtual clusters," in *Cluster Computing and the Grid, 2009. CCGRID'09. 9th IEEE/ACM International Symposium on*. IEEE, 2009, pp. 260–267.

[9] J. Li, M. Qiu, J.-W. Niu, Y. Chen, and Z. Ming, "Adaptive resource allocation for preemptable jobs in cloud systems," in *Intelligent Systems Design and Applications (ISDA), 2010 10th International Conference on*. IEEE, 2010, pp. 31–36.

[10] G. Jung and K. M. Sim, "Agent-based adaptive resource allocation on the cloud computing environment," in *Parallel Processing Workshops (ICPPW), 2011 40th International Conference on*. IEEE, 2011, pp. 345–351.

[11] M. L. Massie, B. N. Chun, and D. E. Culler, "The ganglia distributed monitoring system: design, implementation, and experience," *Parallel Computing*, vol. 30, no. 7, pp. 817–840, 2004.

[12] T. Bernhardt and A. Vasseur, "Esper: Event stream processing and correlation," *ONJava, in http://www. onjava. com/lpt/a/6955, O'Reilly*, 2007.

[13] P. Leitner, J. Ferner, W. Hummer, and S. Dustdar, "Data-driven and automated prediction of service level agreement violations in service compositions," *Distributed and Parallel Databases*, vol. 31, no. 3, pp. 447–470, 2013.

[14] C. Reich, K. Bubendorfer, M. Banholzer, and R. Buyya, "A sla-oriented management of containers for hosting stateful web services," in *e-Science and Grid Computing, IEEE International Conference on*. IEEE, 2007, pp. 85–92.

[15] A. L. Freitas, N. Parlavantzas, and J.-L. Pazat, "A qos assurance framework for distributed infrastructures," in *Proceedings of the 3rd International Workshop on Monitoring, Adaptation and Beyond*. ACM, 2010, pp. 1–8.

[16] V. C. Emeakaroha, I. Brandic, M. Maurer, and S. Dustdar, "Low level metrics to high level slas-lom2his framework: Bridging the gap between monitored metrics and sla parameters in cloud environments." in *HPCS*, 2010, pp. 48–54.

[17] M. Maurer, I. Breskovic, V. C. Emeakaroha, and I. Brandic, "Revealing the mape loop for the autonomic management of cloud infrastructures," in *Computers and Communications (ISCC), 2011 IEEE Symposium on*. IEEE, 2011, pp. 147–152.

[18] D. Verma, *Supporting service level agreements on IP networks*. Sams Publishing, 1999.

[19] V. C. Emeakaroha, M. A. Netto, R. N. Calheiros, I. Brandic, R. Buyya, and C. A. De Rose, "Towards autonomic detection of sla violations in cloud infrastructures," *Future Generation Computer Systems*, vol. 28, no. 7, pp. 1017–1029, 2012.

[20] M. Boniface, S. C. Phillips, A. Sanchez-Macian, and M. Surridge, "Dynamic service provisioning using gria slas," in *Service-Oriented Computing-ICSOC 2007 Workshops*. Springer, 2009, pp. 56–67.