Resource Allocation Based on Agreement with Data Security in Cloud Computing

Aparna D. Deshmukh, Archana Nikose

Abstract- Cloud computing has been envisioned as the next-generation architecture of IT enterprise. In contrast to traditional solutions, where the IT services are under proper physical, logical and personnel controls, cloud computing moves the application software and databases to the large data centers, where the management of the data and services may not be fully trustworthy. cloud migrating from traditional software to Cloud enables on-going revenue for software providers. However, in order to deliver hosted services to customers, SaaS companies have to either maintain their own hardware or rent it from infrastructure providers. This requirement means that SaaS providers will incur extra costs. In order to minimize the cost of resources, it is also important to satisfy a minimum service level to customers. Therefore, this paper proposes resource allocation algorithms for SaaS providers who want to minimize infrastructure cost and SLA violations. An SLA is a formal contract used to guarantee that consumers‟ service quality expectation can be achieved. Cloud Computing moves the application software and databases to the large data centers, where the management of the data and services may not be fully trustworthy. So in this paper focus on cloud data storage security, which has always been an important aspect of quality of service.

Keywords- Cloud computing; Service Level Agreement (SLA); Resource Allocation; Scheduling; Software as a Service.

I. INTRODUCTION

Traditionally the shrink-wrapped software sales model dominated the market. This model requires customers are required to purchase per petual or subscription-based license and manage the deployment themselves, including transitioning between different versions. Hence, customers need technical expertise and high initial investment for buying software. They also need to pay for upgrades as annual maintenance fee.

The software services are provisioned on a pay-as-you-go basis to overcome the limitation of the traditional software sales model. Using the SaaS model, providers gain steady, on-going revenue from their customers. In exchange for the on-going charges, the customers get the benefit of continuously maintained software. Hence, there is no additional license fee for new versions and the complexity of transitioning to new releases is managed by SaaS providers.

SaaS providers such as Computer Associates (CA) derive their profits from the margin between the operational cost of infrastructure and the revenue generated from customers. Therefore, SaaS providers are looking into solutions that minimize the overall infrastructure cost without adversely affecting the customers. Hence, the focus of this paper is on exploring policies to minimize the required infrastructure to meet customer demand in the context of SaaS providers offering hosted software services.

Manuscript received on July, 2013.

Aparna D. Deshmukh, M.Tech.-CSE, Smt. Bhagwati Chaturvedi College of Engg. Nagpur, Maharashtra, India.

Archana Nikose, CSE Department, Smt. Bhagwati Chaturvedi College of Engg. Nagpur, Maharashtra, India.

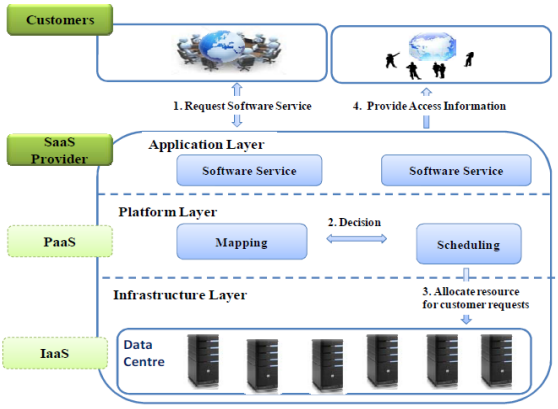


Fig. 1. A system model of SaaS layer structure

A SaaS model for serving customers in Cloud is shown in Fig. 1. A customer sends requests for utilizing enterprise software services offered by a SaaS provider, who uses three layers, namely application layer, platform layer and infrastructure layer, to satisfy the customer's request. The application layer manages all application services that are offered to customers by the SaaS provider. The platform layer includes mapping and scheduling policies for translating the customer's Quality of Service (QoS) requirements to infrastructure level parameters and allocating Virtual Machines (VMs) to serve their requests. The infrastructure layer controls the actual initiation and removal of VMs.

The current works in Cloud computing [10][2] are focused mostly on maximizing the profit of IaaS providers, but works related to the SaaS provider considering SLAs are still in their infancy. Many works do not consider the customer driven management, where resources have to be dynamically rearranged based on customers' demands. Thus, in this paper, we examine the resource allocation strategies, which allow a cost effective usage of resources in Clouds to satisfy dynamically changing customer demands in line with SLAs.

From the perspective of data security, which has always been an important aspect of quality of service, Cloud Computing inevitably poses new challenging security threats for number of reasons. Firstly, traditional cryptographic primitives for the purpose of data security protection can not be directly adopted due to the users' loss control of data under Cloud Computing. Therefore, verification of correct data storage in the cloud must be conducted without explicit knowledge of the whole data. Secondly, Cloud Computing is not just a third party data warehouse. The data stored in the cloud may be frequently updated by the users, including insertion, deletion, modification, appending, reordering, etc.

In this paper, we propose an effective and flexible distributed scheme with explicit dynamic data support to ensure the correctness of users' data in the cloud. We rely on erasure correcting code in the file distribution preparation to provide redundancies and guarantee the data dependability. This construction drastically reduces the communication and storage overhead as compared to the traditional replication based file distribution techniques. By utilizing the homomorphic token with distributed verification of erasure-coded data, our scheme achieves the storage correctness insurance as well as data error localization: whenever data corruption has been detected during the storage correctness verification, our scheme can almost guarantee the simultaneous localization of data errors, i.e., the identification of the misbehaving server(s).

II. ALGORITHMS

The main objective of our work is to maximize the profit for a SaaS provider by minimizing the cost of VMs using effective platform layer resource allocation strategies.

Algorithm: Resource Allocation Algorithm

A SaaS provider can maximize the profit by minimizing the resource cost, which depends on the number and type of initiated VMs. Therefore, this algorithm is designed to minimize the number of VMs by utilizing already initiated VMs. *Algorithm describes* the resource allocation algorithm, which involves two main request types: a) first time rent and b) update service.

The algorithm checks the request type, if the request type is 'first time rent' then there are new user want to buy the services then he can create there account by entering username and password and click on signup button. Then one form is coming in that enter name, password, is admin or not, security question and answer the question. Then account of user has to be created and register amount 1000 is added in his account, then he login. But if same name of user try creating an account then account not created and give message that 'username already exist try crating an account with another username'.

After login token is generated and all services are display for user for 24 hour. For security purpose every time new token is generated. If user have balance less than or equal to service cost then he can buy it otherwise account was updated by contacting with admin. Admin can enter id of user and amount to be added and add amount to user account. After buy services, cost of services reduced from user account and service available for 24 hour and after 24 hour user can again buy it.

If user has forget his password then he must enter username and click on forget password button then security question come which is asked at the time of creating account and answering the question we get password.

ALGORITHMS

First Time Rent *{*

1. **If (press button signup) *{***
2. *(Enter name, password, is admin, security question, answer)*
3. *Register amount 1000*
4. **If (press login button*) {***
5. *Welcome user, token generated, all services display for user*
6. **If(user amount is >= service cost) *{***
7. *Buy service and reduced cost*
8. ***}***
9. ***Else {***
10. *Contact with admin to add amount*
11. ***}***
12. *Else **{***
13. **If (press forget password)*{***
14. *Answer security question*
15. *Get password}*
16. ***}***
17. **If (same name of user)*{***
18. *Username Already Exists Try Creating An Account With Another Username*
19. ***}***
20. ***}***
21. *Logout*
22. ***}***

If the request type is 'update', then the type of update is checked. If user have balance less than or equal to service cost then he buy it but if he want to buy the services which having more cost than available balance of user then he contact with admin to add amount in his account.

If update type is 'add account', then it first checks the user id and amount to be added. Then press button update amount then amount was updated in user account and user buy services that user want.

But update type is manage service then admin can enter service id, service cost, service name, service url. If admin press add then service is added or if he press update then service was updated or if he press delete then service was deleted.

Update *{*

1. **If (admin is login) {**
2. **If (if update type is update amount) {**
3. get Id *i and amount to be added*
4. ***Account** was updated* **}**
5. **Else {**
6. Logout }
7. }
8. **Else** { **If (update type is manage service){**
9. **If (press button add)**
10. *get service id, service cost, service name, service url*
11. **service added }**
12. **Else**
13. **If (press button update){**
14. *get service id, service cost, service name, service url*
15. **services updated}**
16. **Else If(press button delete){**
17. *get service id, service cost, service name, service url*
18. **service delete}**
19. **}**
20. *Logout*
21. **}**

These algorithms maximize the profit of saas provider and minimize the infrastructure cost and also provide data security by generating homomorphc token.

III. EXPERIMENTAL RESULT

In this paper, we propose an effective and flexible distributed scheme with explicit dynamic data support to ensure the correctness of users' data in the cloud. In cloud data storage system, users store their data in the cloud and no longer possess the data locally.

One of the key issues is to effectively detect any unauthorized data modification and corruption, possibly due to server compromise. Then, the homomorphic token is introduced. In this there are three modules are present as mentioned below with details.

- **Client Module:**

In this module, the client sends the query to the server. Based on the query the server sends the corresponding file to the client. Before this process, the client authorization step is involved. In the server side, it checks the client name and its password for security process. If it is satisfied and then received the queries from the client and search the corresponding files in the database. Finally, find that file and send to the client. If the server finds the intruder means, then message come can't login.

In this module there are token based login that means, login with system with the help of token ,so that data security on cloud will maintained.

- **SLA module**

SLA means service level agreement. An SLA is a formal contract used to guarantee that consumers will get service quality as per expectation what he paid for. Fundamental issue is the management of SLAs, including SLA autonomy management or trade off among multiple Quality of Service (QoS) pa-rameters. In this project resource can be allocated to the software as service provider based on service level agreement. In this module customer request submitted to server based on his requirement where SLA logic implemented. Each user would have a balance and there are 'N' number of services each for different balance. After a login with the system user would be shown a service screen with 'N' service depending upon balance.

Depending upon the service selector balance would be deduced from the user account and service available for 24 hour and new token for that service would be generated for the user. After everyday user are logged out to ensure cloud reliability.

- **Service module.**

Give the service to customer according to his request. In this there are Calculator, Currency Convertor, Graphing, Google, Open Pdf, and Viewdocsonline service are online and engineering calculator and cost calculator service are offline.. As per project title mainly concentration on first two modules. First is Login module where application will take care of data security from intruder and second is SLA where application will take care to provide service as per agreement by allocating resources to end user.

The first module focuses on login with data security with the help of random token generated by application. when we enter user name and password, application will check user exist or not. If user is not exist it will not give access to application provided on cloud. If any intruder tries to access home page with invalid id and token combination he/she will be restricted and will be given output that your id and token doesn't match. If user exist then user will allow to login with randomly generated token for data security. Also application will maintain token in database so that it will prevent unauthorized access. When authorized user login then every time new token is generated, in this way it will maintained the security of data on cloud.

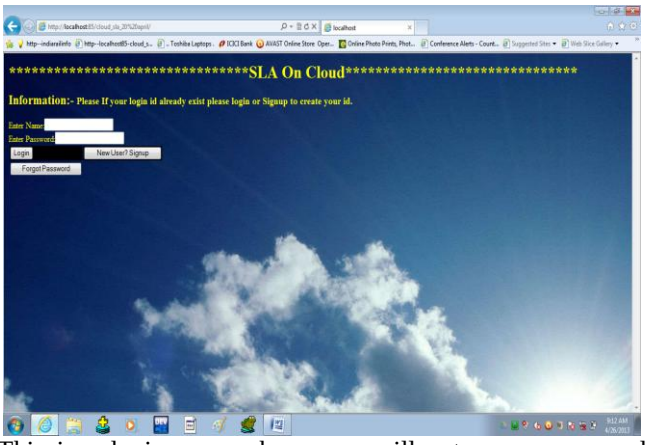
After authorized login service page available to user. If he want buy service then he/she put the id of service and click on buy button then service page available to user. In this project there are six services are available to user and every service have id. If user put 1 in ENTER ID button then calculator service is available to user. If he put 2 then currency convertor, if put 3 then graphing service ,if put 4 then graphing if put 5 then open pdf, if he put 6 then view document service are available to user.

If user amount is less than service cost and he want to buy service which have high cost than available to user then he contact with admin in (admin2013 @gmail.com) and add amount.

Admin can add amount to user account. If user want to add amount then he can put id of user and amount to be added and click on update amount button then amount is added to user account. Admin can manage service by delete, add, update button. He can put service id, service name, service cost, service URL and add, delete or update service.

3.1. Login Page

This is first page of user account. If user want to create there account then click on signup button or if already have account then click on login button.

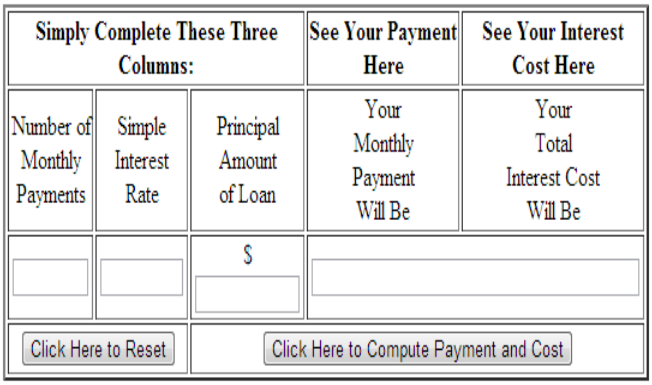


This is a login page where user will enter username and password. Once user enter userid and password application will check whether same user exist in database or not. If exist then he will be routed to next application page.

3.2. Data Security and Service page

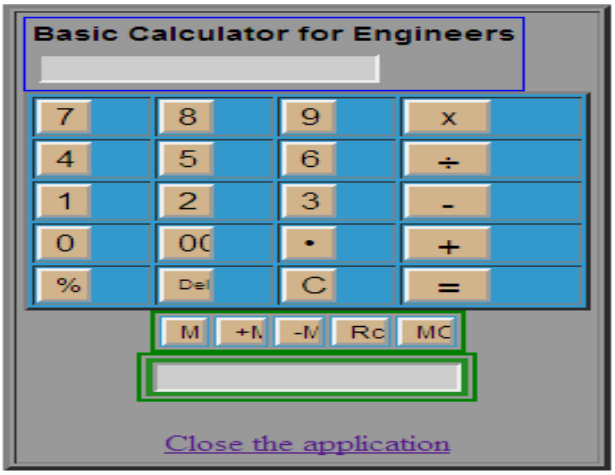
When user is login then security token is generated. Every time new token is generated. So that data security is maintained and services available.

3.3. Cost Calculator



If customer put 9 in service id then click on buy button then cost calculator service is available to user for 24 hour.

3.4. Calculator

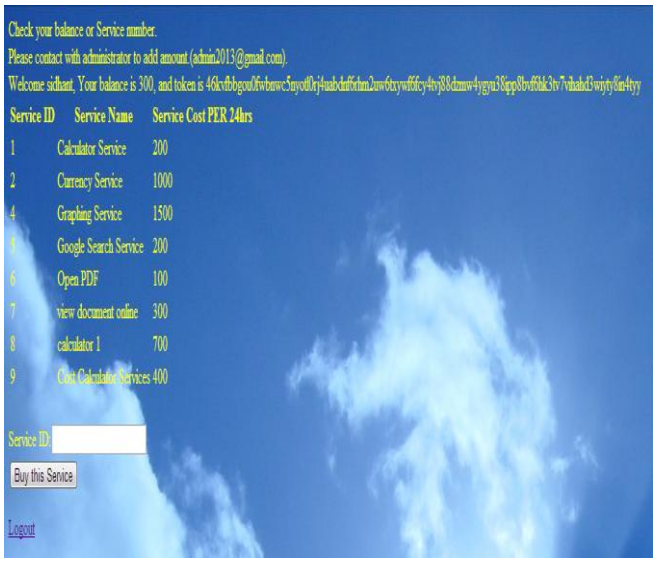


If customer put 8 in service id then calculator service availabale for user for 24 hour.

3.5 Manage service page of admin

Admin can manage all services. If he want add, delete, update service then he can put service id , service cost, service URL, service name of service and manage it.

3.6. Admin contact page



If user have balance less than service cost then he contact with admin in (admin2013@gmail.com) and add amount and buy service that user want. Admin insert id and amount to be added and update the user account. So that customer can buy services for 24 hour after his time slot user agsin buy this service if he want.and cosr of services reduced in his account.

3.7. Update amount page for admin

User have balance less than service cost then user contact with admin to add amount then admin can put id and amount to be added and click on update button then amount is addes to user account.

IV. CONCLUSION

In Cloud computing environments, primarily three types of on-demand services are available for customers i.e. Software as a Service, Infrastructure as
a Service and Platform as a Service. This paper focused on scheduling customer requests for SaaS providers with the explicit aim of cost minimization with dynamic demands handling. In this paper, we investigated the problem of data security in cloud data storage, which is essentially a distributed storage system. To ensure the correctness of users' data in cloud data storage, this paper proposed an effective and flexible distributed scheme with explicit dynamic data support, including service update, delete, and append.

FUTURE SCOPE

In building on the research undertaken in this project in the future, we will analyze ways to increase the efficiency of the algorithms in terms of total profit and shall also consider the SLA negotiation process in Cloud computing environments to improve customer satisfaction levels.

In the next twenty years, service-oriented computing will play an important role in sharing the industry and the way business is conducted and services are delivered and managed. This paradigm is expected to have major impact on service economy; the service sector includes health services (e-health), financial services, government services, etc.

REFERENCES

1. C.S. Yeo, and R. Buyya,"Service level agreement based allocation of cluster resources: Handling penalty to enhance utility". In Proceedings of the 7th IEEE International Conference on Cluster Computing Bostan, MA, USA, (Cluster 2005).
2. Y.C. Lee, C. Wang, A.Y. Zomaya and B.B. Zhou, "Profit-driven Service Request Scheduling in Clouds". In Proceedings of the International Symposium on Cluster and Grid Computing, (CCGrid 2010), Melbourne, Australia.
3. R. Buyya, J. Broberg, and A. Goscinski (eds). *Cloud Computing: Principles and Paradigms*. ISBN-13: 978-0470887998, Wiley Press, USA, February.
4. R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud Computing and Emerging IT Platforms: Vision, Hype, and Reality for Delivering Computing as the 5th Utility, Future Generation Computer Systems", 25(6), (pp. 599-616), Elsevier Science, Amsterdam, The Netherlands.
5. J. Broberg, S. Venugopal, and R Buyya, Market-oriented Grids and Utility Computing: The state-of-the-art and future directions, Journal of Grid Computing, 3(6), (pp.255-276).
6. I. Popovici, and J. Wiles, "Proitable services in an uncertain world". In Proceeding of the18th Conference on Supercomputing (SC 2005), Seattle, WA.
7. D. Parkhill, "The challenge of the computer utility", 1966, Addision-Wesley Educational Publishers Inc., USA.
8. Y. Song, Y. Li, H. Wang, Y. Zhang, B. Feng, H. Zang, Y. Sun, "A Service-Oriented Priority-Based Resource Scheduling Scheme for Virtualized Utility Computing", High Performance Computing-HiPC 2008.
9. S.K. Garg,R.Buyya, and H. J. Siegel, " Time and Cost Trade-off Management for Scheduling Parellel Application on Utility Gride ", Future Generation Computer System.26(8). (pp. 1344-1355).
10. Rodrigo N. Calheiros, Rajiv Ranjan, Anton Beloglazov, Cesar A. F. De Rose, and Rajkumar Buyya, CloudSim: A Toolkit for Modeling and Simulation of Cloud Computing Environments and Evaluation of Resource Provisioning Algorithms, Software: Practice and Experience (SPE), Volume 41, Number 1, Pages: 23-50, ISSN: 0038- 0644, Wiley Press, New York, USA, January, 2011.
11. A. Juels and J. Burton S. Kaliski, "PORs: Proofs of Retrievability for Large Files," Proc. of CCS '07, pp. 584–597, 2007.
12. K. D. Bowers, A. Juels, and A. Oprea, "Proofs of Retrievability: Theory and Implementation," Cryptology ePrint Archive, Report 2008/175, 2008, http://eprint.iacr.org/.
13. G. Ateniese, R. D. Pietro, L. V. Mancini, and G. Tsudik, "Scalable and Efficient Provable Data Possession," *Proc. of SecureComm '08*, pp. 1–10, 2008.