International Journal of Applied Engineering Research ISSN 0973-4562 Volume 13, Number 19 (2018) pp. 14402-14410
 https://dx.doi.org/10.37622/IJAER/13.9.2018.14402-14410

Verification as a Service (VaaS): A Trusted Multipurpose Service for Accounting, Billing Verification and Approval of Consumed Resources at Low Computational Overhead to Enhance the Cloud Usability

Himadri Biswas[1], Debabrata Sarddar[2]

[1]PhD Scholar, Computer Science and Engineering Department, University of Kalyani, Kalyani, Nadia, West Bengal, India.
[2]Assistant Professor in the Department of Computer Science and Engineering, University of Kalyani, Kalyani, Nadia, West Bengal, India.

Abstract

Among numerous computing services in cloud computing Billing is one of the utmost promising field. The users always expect the best services with minimal cost and a trustworthy manner from their service provider. Existing billing systems are limited in terms of verification or computational overhead, even third party allowed the user to interact with the CSP. As a result third party cannot verify that either user's record or the CSP's record is correct.

In this paper, we propose a Verification as a service (VaaS) as a medication for these limitations. Cloud infrastructure for VaaS is solely managed by the Verification Service Provider (VSP) without any intervention of the users, as well as the Service Providers. This service can be used for any type of verification at the user side as well as the provider's side. Here we only implement a part of a service i.e. VAMS (Verification & Approval Management System) module for the verification of the billing at the user side. This module is not only used as a third party for bill verification purposes, but as a faster bill generator with low computational cost and low overheads.

Keywords: Verification as a service, Cloud Service Provider, Verification Service Provider, Service Level Agreement, Inter task communication cost, Inter process communication cost.

INTRODUCTION

In essence Cloud computing is based on a "pay-as-you-use" basis model where computing resources are accessed through Internet technologies. CSPs offer different services based on the pricing category- basically Fixed cost and dynamic cost; Fixed cost billing is quite reasonable, being based simply the customer have to pay the same amount all the time i.e. a subscription basis; dynamic, in which the charging price changes dynamically; or market-dependent, in which the customer have to pay based on the real-time market conditions [1]. On the basis of SLA, CSP may provide the usage details along with the bill to the users for ensuring that which computing resources were consumed and when they were initiated. Unfortunately providers may charge higher bills where fairness is immaterial until clients are able to grasp them accountable and verifiable for it.

Anyway of whether the Cloud Computing focuses on Infrastructure as a Service (IaaS), Platform as a Service (PaaS) or Software as a Service (SaaS), Verification as a service (VaaS) is the missing link, the key challenges quick to react to changing business circumstances.

The proposed model provides the new computing environment- VaaS, which ensures the accountability and verifiability on the data between the user and the CSP because a user pays for its uses, so user needs a protection from any false accusation that may be claimed by the service provider to get illegal compensations.

The VAMS (Verification & Approval Management System) model implementation proposed in this project aims at providing a dynamic billing & Verification mechanism for cloud users and service providers also. The service provider defines a pricing based on the present load on the cloud using the COMPUTE module. It is then the user's choice to continue using the cloud resource by paying the specified amount or to terminate the cloud resource.

RELATED WORKS

Many research work done on billing systems using the computing resources in grid and cloud computing environments [6], [7], and an attempt to alter from grid computing to cloud computing to find out and fulfill the new requirements. Mihailescu and Teo introduced a dynamic pricing scheme for federated clouds, would be beneficial because it would set the price according to the levels of supply and demand in which resources are shared among many cloud service providers [2]. Macias and Guitart, proposed a genetic model for pricing in cloud computing markets [3]. Selecting a good pricing model via their genetic algorithms concerned three main steps: define a chromosome, appraise it, and at last choose the best pairs of chromosomes for reproduction and neglecting those with the worst results. The results of the simulation demonstrated that genetic pricing reached revenues greater than the other dynamic pricing or fixed pricing strategy. Vaquero [8], the service of advanced billing method permit for a pay as you use model for shared multi tenant resources. Stefan Tai and other researchers set up that the Cloud service customers are charged based on actual service usage, though the exact costs, pricing and billing models from both customers and service providers viewpoints are not

mentioned properly. So how do we estimate costs, compare substitute, assess risks and find out value of conventional versus cloud computing clarification? With the help of this approach we cannot estimate the cost [9]. Different authors, like Wang et al. [11] and Li et al. [10], have also observed accounting in a cloud environment. Wang revealed that users are compared with shops among vendors, and Li illustrate that cost and performance for the similar task can differ notably across different cloud vendors. Billing of Cloud storage access should be based on exertion [12], This paper quarrel for exertion-based metrics, like disk time, for accessing the cost component of cloud storage billing. It also consider faces in supporting fair and predictable exertion accounting, such as significant interwork load intervention effects for storage access, and a performance lagging approach to addressing them. A storage exertion based standard billing model and varied rates for it over time and across different providers would introduce market forces [13]. The Reservoir project inspects the use of service level agreements [14] for resource provision in federated cloud scenarios. Georgios Gousios and others also projected the design and implementation of extensible billing service software, Aquarium [15]. This approach offers portability but desired performance cannot be achieved. A complete review of various cloud billing models has been presented in [16]. The authors discuss here most pricing models in cloud computing are not fair toward the service provider and most of them aimed to increase the service provider's revenues and decrease its costs. Attributes regarding the end user, such as user satisfaction level, QoS, end user utility, etc would include a better pricing approach. Work has done on resource accounting and billing in the context of cloud federation [17-18] and (earlier) grid federation projects. In commercial cloud service the transaction of billing and management are both guided by CSP alone rather than mutual verifiability of billing transactions. Another PKI (Public Key Infrastructure) [5] based mechanism for imposing the verifiability of billing system, though the computational complexity of PKI may result in a high computational overhead and unbearable response time of billing due to its asymmetric key operations and for digital signatures need to be carried out with regards to both the customer terminal and CSP. In [4], as a remedy for the above mentioned limitations the authors proposed a secure and non-disruptive billing system called THEMIS. Here the authors describe the trustworthy audit trail used by the CNA (cloud notary authority) as a third party to resolve any disputes between the user and the CSP, because at any time the user or the CSP can modify the billing records even after a mutual contract between them. Another Billing and Verification mechanism represented in [23], where main focus is on systematic record maintenance of resources used by a consumer of cloud computing services, where the price is determined based on the threshold value of a node. Node threshold value is calculated instantly at the time of initiation and price can be determined based on the threshold value falls in which range followed by the price table. In [24], the concept is totally different from Trusted Third Party (TTP), where Security Service Provider (SSP) allows the data owner to outsource data that is responsive to a Cloud Service Provider, and it also ensures that only the authorized users receive the data that is outsourced. Further, it enables direct mutual trust between the Cloud service provider and the data owner, *i.e.* SSP acts as a media which only help to make the secure connection between the communicating nodes or the nodes that want to communicate each other, but not directly store or retrieve or supply of any secured data. But there is no such verification mechanism which may prevent direct intervention between the users and the service providers to avoid false acquisition or blame to each other from the resource allocating phase to bill generating phase of the SLAM (Service Level Agreement Manager) model.

OVERVIEW AND RATIONALE

Cloud Service Models:

The cloud computing service models are –

Software as a Service (SaaS): A readymade application, along with any necessary software, operating system, hardware, and network are provided by this SaaS model.

Platform as a Service (PaaS): Hardware, network and an operating system are provided by this PaaS model, whereas the consumer can install or builds up its own software and applications.

Infrastructure as a Service (IaaS): Only the hardware and networks are supplied by this IaaS model, the customer set up or develops its own operating systems, software and applications.

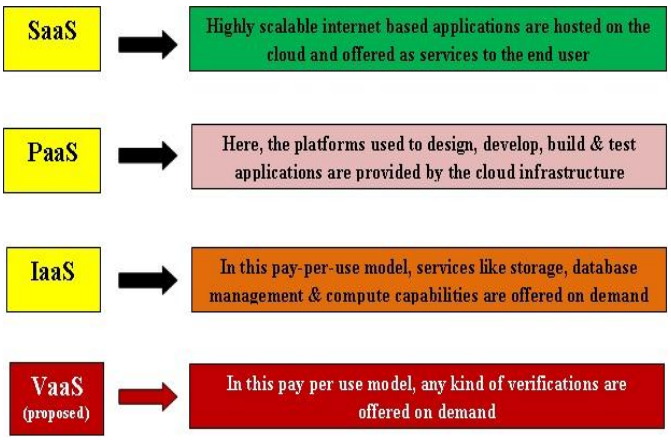

Figure 1. Cloud Service Models

Proposed service model:

Verification-as-a-Service (VaaS): VaaS under Private Cloud provides the security to access the resources on the cloud server and impose the access control on the data. The main concept is to verify the consumed resources by a cloud user without any partiality either or both sides and no chance of intervention between the users and the Service Providers directly. Because once a user or a service provider registered in Vaas, has to go through this service.

International Journal of Applied Engineering Research ISSN 0973-4562 Volume 13, Number 19 (2018) pp. 14402-14410
© Research India Publications. https://dx.doi.org/10.37622/IJAER/13.9.2018.14402-14410

Deployment Models:

Cloud services are usually made available via a private cloud, community cloud, public cloud or hybrid cloud.

Public cloud: Public cloud provides services over the Internet and are owned and directed by a cloud provider. Some examples include services such as e-mail services, online photo storage services or social networking sites intended at the general public,. Still, services for enterprises can also be offered in a public cloud.

Private cloud: Private cloud infrastructure is operated exclusively for a dedicated organization, and is supervised either by the organization or a third party.

Community cloud: Here, the service is shared among several groups and made available only to those groups. The infrastructure may be owned and operated either by the groups or by a cloud service provider.

Hybrid cloud: A hybrid cloud is a mixture of different methods of resource pooling (for example, combining public and community clouds). Except these, another types of clouds are-

Federated clouds: It represents public clouds that use the same cloud infrastructure standard [22]. Therefore, VMs can easily be roamed between the federated clouds possessed by several cloud providers. A cloud federation requires, at least, a contract between cloud providers to commit to a specific cloud infrastructure standard.

Federated hybrid clouds: Here, a cloud user may run some of its services on its private cloud and some others on a federated clouds [22].

Differentiated clouds: It implies the overall cloud market, where cloud providers offer their services using different standards [23]. Consequently, the different public clouds are not interoperable. A cloud user, who wants to use several public clouds, would require to use the standards of each cloud. This category encompasses clouds using the same cloud standard and is possessed by the same provider. Those clouds are to be found at different geographical locations (e.g., Amazon AWS).

Cloud Service Provider (CSP):

Service provider provides the virtual components like computers, disks, file-systems, databases, or even higher level constructs with the assurance of watching out of the virtual components, and the customers just need to upload their program and run it. So, a CSP supervises cloud servers and provides storage space and resources pay and use basis on its infrastructure for storing files and using the resources by the authorized users

Verification Service Provider (VSP):

VSP under VaaS, plays the important role to establish a smooth communication between the user and the CSP. So before storing or accessing the existing data or resources on the cloud server, VSP registers the Users and the CSP's to avoid future disputes. VSP maintains a security log table which holds all identification keys of the users and their respective CSPs.

Service Level Agreement (SLA):

It is a bond and a form of a guarantee between a service provider (either internal or external) and the customer that defines the level, quality, quantity, scope and responsibility of service expected from the service provider. SLAs are based on output in that their aim is specially to describe what the customer will receive.

PROPOSED WORK

In this paper, we present our proposed service model Verification as a Service (VaaS), which keeps track of resources used by a cloud user and has the potential to yield benefits for cloud service providers and consumers alike.. Already we know that there are different cloud service models and their intended service providers available in the market, but most of them do not allow customers to verify their used resources or allowed on request basis. Once allowed, most of the cases they do not realize their return records are correct or not. In this situation they need help of a third party on faith. Our aim in this work is to provide a service model VaaS where the users if they want must be going through this model before getting any services from the CSP's and in that case no CSPs should not provide any final bill to the users without getting approval from VAMS under VaaS. Our proposed VaaS model not only used for verification of consumed resources by a cloud user, it shows how fast to calculate the bill. So any service provider can use this model to generate their bill also.

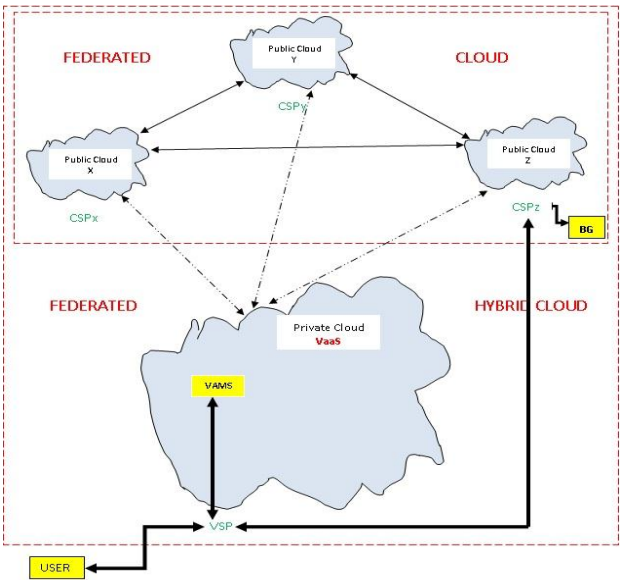

Figure 2. Verification as a Service (VaaS) Model Architecture

International Journal of Applied Engineering Research ISSN 0973-4562 Volume 13, Number 19 (2018) pp. 14402-14410
 https://dx.doi.org/10.37622/IJAER/13.9.2018.14402-14410

Algorithm: Verification as a Service (VaaS)

1. User requests for billing and record details to VSP
2. VSP checks for user's authentication.
3. If user is authentic –
   - 3.1 VSP forwards request to VAMS
   - 3.2 VSP forwards the request to dedicated CSP's.
   - 3.3 CSP check for user authentication & SLA.
   - 3.4 If authentic –
     - 3.4.1 Accepted for services and forwards to VAMS
     - 3.4.2 VAMS collects the record details from its log file.
     - 3.4.3 VAMS generates the bill.
     - 3.4.4 VAMS forwards the bill as well as periodic signals / records to CSP.
     - 3.4.5 CSP forwards to VSP.
     - 3.4.6 VSP forwards to VAMS.
     - 3.4.7 VAMS computes the bill and match with the received bill details.
     - 3.4.8 If matches –
       - 3.4.8.1 VAMS approve the bill and sends to BSP
       - 3.4.8.2 VSP sends the bill as a final to the user.
       - 3.4.8.3 VSP sends an acknowledgement to the CSP.
     - 3.4.9 Else –
       - 3.4.9.1 VAMS forwards the bill to VSP for re computing.
       - 3.4.9.2 Go to Step 3.2
   - 3.5 Else –
     - 3.5.1 Go to Step 2.
4. Else –
   - 4.1 Return a message – "Not accepted for services" to the user.

Flow Chart:

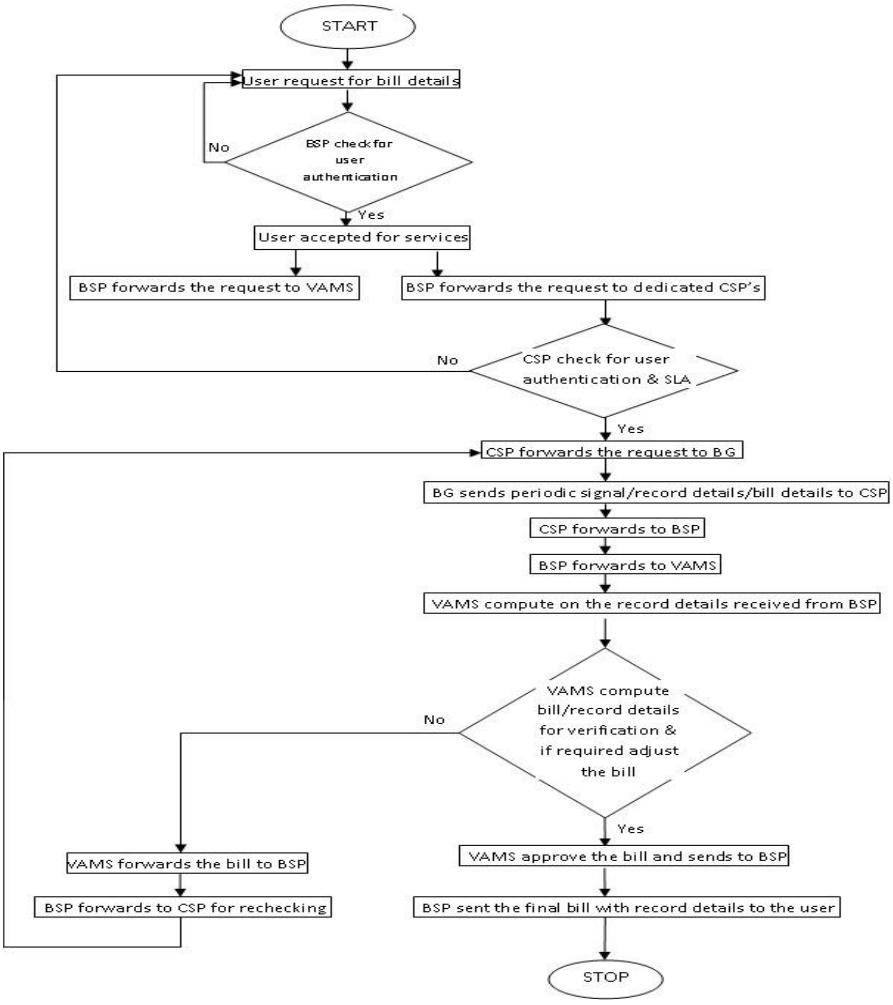

Figure 3. Flow chart

International Journal of Applied Engineering Research ISSN 0973-4562 Volume 13, Number 19 (2018) pp. 14402-14410
 https://dx.doi.org/10.37622/IJAER/13.9.2018.14402-14410

Working Process of Verification & Approval Management System (VAMS):

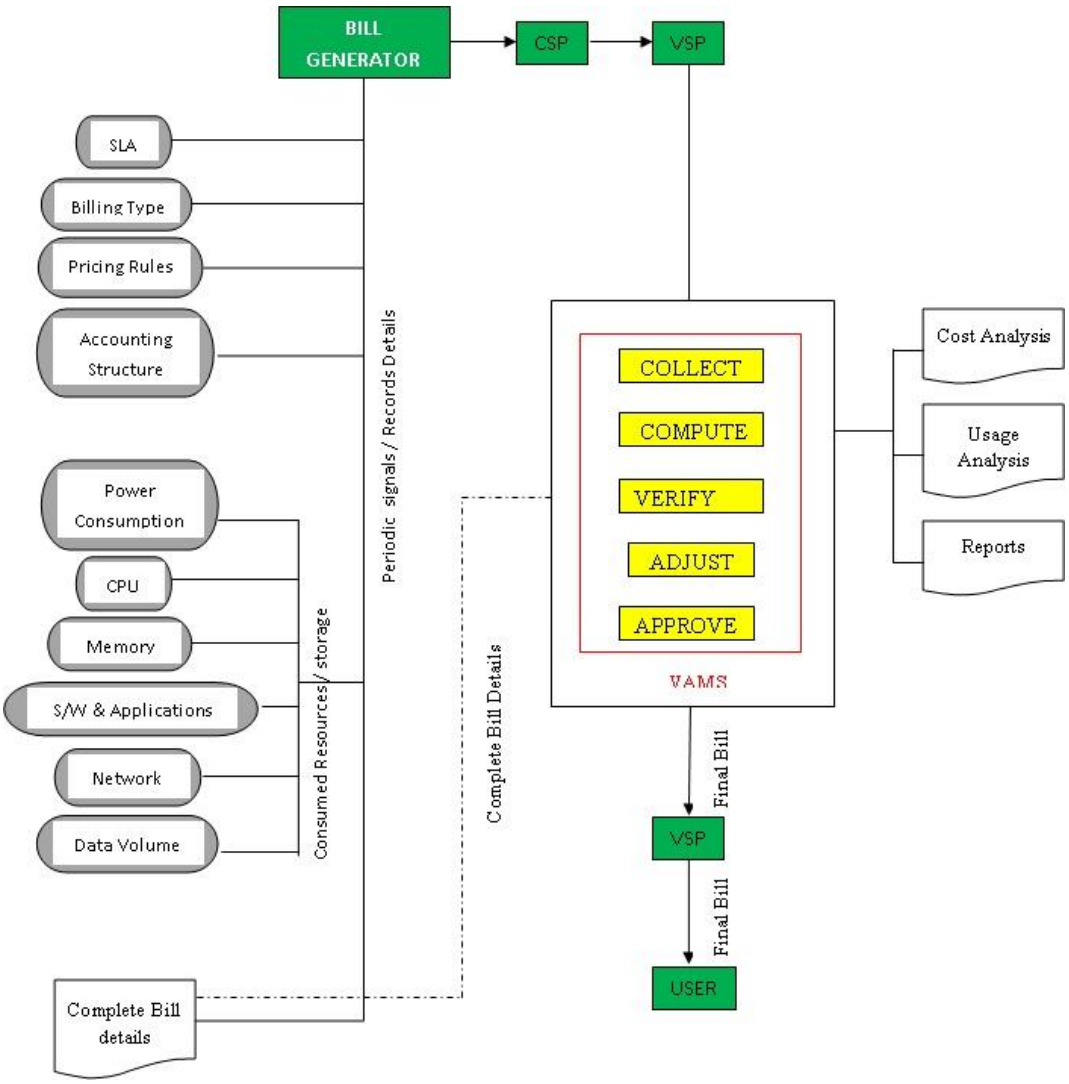

Figure 4. Working Process of Verification & Approval Management System (VAMS)

In a cloud computing environment, the most important hurdle in trying to present an accurate billing statement to a client is to compute the resources consumed in terms of processing and data storage. To be effective, a VAMS has to closely monitor the processing demands placed by a user activity on virtual machines, data storage systems, and network bandwidth. Here VAMS has the different layers-- Collect layer which receives the periodic signals or detailed records of the users and complete accounting & billing records from its respective CSP's through the VSP. Compute layer computes based on the collected records followed by the accounting standards and produce the reports analyzing on cost and usage details. This report compared with the bill generated by the CSP in the Verify layer. If found minor dissimilarities, fine-tuned in Adjustment layer and send to the Approve layer for granting it and finally sent to the user through VSP, otherwise a message forwards to the CSP for rechecking.

Total Cost Estimation by COMPUTE module of VAMS Model:

User submits a number of processes and each process contains a number of tasks. Each process executes under the same node (processor) or different nodes. The process is assigned to the node based on the threshold value of that node, threshold value of a node is calculated as a product of all the node's average workload and the speed of the processor, which is constant and depends on the processing capability of that node relative to the processing capability of all other nodes. So, before calculating the overall cost of all processes we have to calculate the inter task communication cost and inter process communication cost.

International Journal of Applied Engineering Research ISSN 0973-4562 Volume 13, Number 19 (2018) pp. 14402-14410
 https://dx.doi.org/10.37622/IJAER/13.9.2018.14402-14410

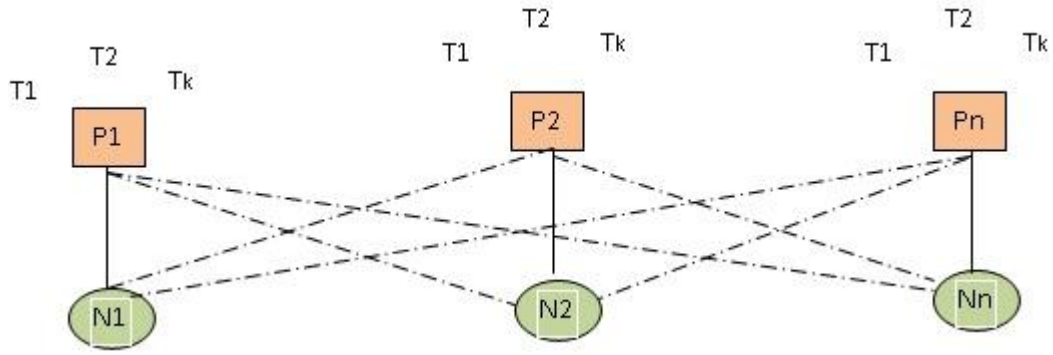

Figure 5. Each process (P1...Pn) contains a number of tasks (T1..Tk) and one process can be assigned to the same node or different nodes (N1...Nn)

Inter task communication cost (ITCc): Communication occurs between a pair of tasks is called Inter task communication. During the execution of the process, if two tasks communicates l times and if the average times for each inter task communication is t, then ITCc for the two tasks is (l × t) × cost/sec.

Inter process communication cost (IPCc): Inter process communication is between a pair of processes. During the execution of the processes, if two processes communicates m times and if the average times for each inter process communication is r, then IPCc for the two processes is (m × r) × cost/sec.

Now, we find out the total cost of all the processes submitted by the user i.e, **C=(n×Pc)+IPCc**, where **C** = Total Cost, n = Number of processes submitted by a user, **Pc** = Cost of a process execution, **IPCc** = Inter process communication cost.

Let us know that, each process contains a number of tasks. So, the cost of processing each task of individual processes on their assigned node(s),

$$Tc = \frac{\text{Amount of computation by each task in bytes}}{\text{Processor Speed}} \times \text{cost/sec} + PWc$$

$$\text{So, } Tc = \frac{\text{Number of instructions executed + CPU Usage + memory usage + S/w and applications Usage + network usage + data storage usage}}{\text{Amount of data in bytes that the node can process/sec}} \times \text{cost/sec} + PWc \text{ ---------- (i)}$$

Where, PWc is the power consumption cost of a task. If the power consumption (in sec) by each task is PW, then power consumption cost by each task is PWc = (PW × cost/sec)

If the number of tasks of a process is k, then total cost of processing of all tasks of a process is PTc = (Tc × k) ---------- (ii)

Inter task communication cost of a process,

$$ITCc = \sum_{i=1}^{k-1} \sum_{j=i+1}^{k} ITCc_{ij}, \quad ITCc_{ij} \begin{cases} 1 & i \to j \\ 0 & i \not\to j \end{cases} \quad \text{(iii)}$$

Where, k is the number of tasks of a process, i and j are the pair of tasks for inter task communication.

So, total cost of a process execution, Pc = PTc + ITCc [from (i), (ii) & (iii)] --------- (iv)

Inter process communication cost,

$$IPCc = \sum_{a=1}^{n-1} \sum_{b=a+1}^{n} IPCc_{ab}, \quad IPCc_{ab} \begin{cases} 1 & a \to b \\ 0 & a \not\to b \end{cases} \quad \text{(v)}$$

Where n is the number of processes submitted by the user;

a,b are pair of processes for inter process communication.

So, the total cost of all the processes submitted by the user,

C = (n × Pc) + IPCc, $n \geq 1$, $IPCc \geq 0$ [from (iv) and (v)] – ---------- (vi)

If n = 0, then IPCc = 0; So, C = 0

If n = 1, then IPCc = 0, So, C = Pc

If n = 2, then C = 2Pc + IPCc

For small number of processes (similar sizes), if we neglect the inter process communication cost (i.e. IPCc), then C = nPc i.e. once the COMPUTE module calculate the cost of a process, it can easily calculate the total cost, because at the time of process submission by the user COMPUTE module keeps track of the number of processes.

Algorithm: Compute_Bill

Notations:

Cost Per Second – Cs

Processor Speed - PS

Task Cost – Tc

Inter Task Communication Cost – ITCc

Inter Process Communication Cost – IPCc

Process's Tasks Cost – PTc

Process Cost – Pc

Total Cost – C

Number of process – n

Number of Tasks of a process - k

Number of communication between two tasks – l

Average time for each inter task communication – t

Number of communication between two processes – m

Average time for each inter process communication – r

Power consumption (in sec) by each task – PW

Power consumption cost by each task – PWc

Begin Compute_Bill

Data: Initialize Tc to Zero

Data: Initialize ITCc to Zero

Data: Initialize IPCc to Zero

Data: Initialize PTc to Zero

Data: Initialize Pc to Zero

Data: Initialize C to Zero

Data: Set_Of_Process

Data: Set_Of_Task

Data: l

Data: t

Data: m

Data: r

Data: Constant Cs

Data: Consumption

For each Set_Of_Process [i=1 to n] do

For each Set_Of_Task [j=1 to k] do

consumption ← *collect*(Resource)

Tc ← *compute*(consumption)

PTc ← PTc +Tc

j ← j+1

Endfor

For each Set_Of_Task [u=1 to k-1] do

For each Set_Of_Task [v=u+1 to k] do

If [u interacts with v]

ITCc_uv ← (l*t)*Cs

ITCc ← ITCc + ITCc_uv

Else

ITCc ← ITCc

v ← v + 1

Endfor

u ← u+1

Endfor

Pc ← PTc+ITCc

Store (Set_Of_Process[i], Pc)

i ← i+1

Endfor

For each Set_Of_Process [a=1 to n-1] do

For each Set_Of_Process [b=a+1 to n] do

If [a interacts with b]

IPCc_ab ← (m*r)*Cs

IPCc ← IPCc + IPCc_ab

Else

IPCc ← IPCc

b ← b + 1

Endfor

International Journal of Applied Engineering Research ISSN 0973-4562 Volume 13, Number 19 (2018) pp. 14402-14410
© Research India Publications. https://dx.doi.org/10.37622/IJAER/13.9.2018.14402-14410

a ← a+1

Endfor

For each Set_Of_Process [i=1 to n] do

C ← C+ (Set_Of_Process[i], Pc)

i ← i+1

Endfor

C ← C+IPCc

Store (C, Total Cost)

End Compute_Bill

Begin *collect*

Data: Resource

Data: Consumption

Resource ← (executed instructions + consumed storage + consumed S/w and

applications + consumed memory + consumed network)

Consumption ← VAMS.collect(Resource)

Return Consumption

Endcollect

Begin *Compute*

Data: Consumption

Data: Tc

Data: PW

Data: PWc

Data: PS

PS ← Amount of data processing in sec.

PWc ← PW*CS

Consumption ← [(Consumption / PS) * CS] + PWc

Tc ← VAMS.Compute(Consumption)

Return Tc

Endcompute

RESULT ANALYSIS

The basic operational cost of a process (based on the process size) is specified by a Serviced Provider or once calculate the cost of a process based on the current work load consider the cost of the process is fixed for that moment. Our COMPUTE module of VAMS model keeps track of the number of submitted processes by a cloud user, so overall cost can easily be calculated and it founds that total cost varies with the number of process execution.

Table 1. Process Cost calculation

No. of processes(n)	cost of a process(Pc)	Total Cost(C)
1	10	10
2	10	20
3	10	30
4	10	40
5	10	50
6	10	60

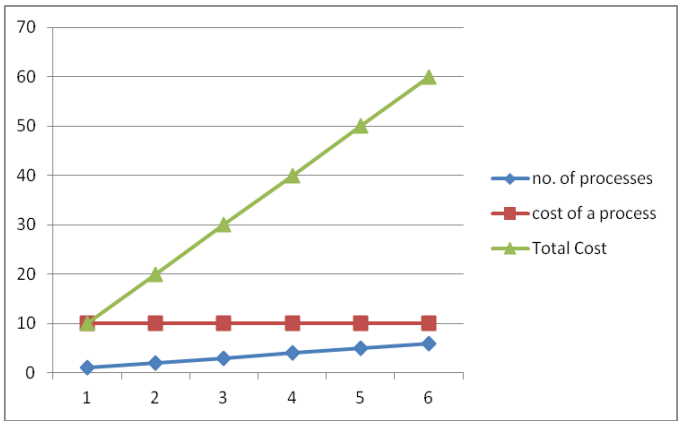

Figure 6. Process Cost Analysis

CONCLUSION

The main focus of this paper is to provide a Compute and verification as a service (VaaS) as an implementation dynamic pricing scheme for federated clouds [2] in which resources are shared among many cloud service providers. Dynamic pricing would be beneficial for calculating the actual price than fixed pricing because it would set the price according to the levels of supply and demand and should improve the reliability and scalability for both users and providers. Users always collect their updated information from VAMS module due to its continuous record collection and verification process. Our VAMS module not only helps to the user for verification of their bill provided by the CSP, but the CSPs would be benefited for preparing their bills with the proper resource utilisation. Our system make different than other systems [20], [4] due to its little user interaction, the whole matter looks into the proposed module at a low computational cost and overheads.

FUTURE SCOPE

Our motto is to enrich the QoS, end user utility, satisfactory level of the users as well as the service providers as far as low investment cost is possible and in future we want to implement other verification modules under VaaS to make cloud computing more accessible.

REFERENCES

[1] White paper "Billing and Revenue Management for Cloud Computing" oracle,2010.

[2] M. Mihailescu and Y. M. Teo, "Dynamic Resource Pricing on Federated Clouds", Proc. 10th IEEE/ACM Int. Symp. on Cluster. Cloud and Grid Computing, (2010).

[3] M. Macias and J. Guitart, "A Genetic Model for Pricing in Cloud Computing Markets", Proc. 26th Symp. of Applied Computing, (2011).

[4] U. Yamini, K. SathiyaPriya," Trusted SLA Monitoring for Billing System in Public Cloud computing Environment", IJMER,2010.

[5] Stefan Kelm, "Public Key Infrastructure", http://www.pki- page.org/, 2009.

[6] D.Banks, J.S.Erickson, and M.Rhodes, "Towards Cloud-based Collaboration Services", Usenix Workshop HotCloud 2009.

[7] N.Santos, K.P.Gummadi, and R.Rodrigues, "Towards Trusted Cloud Computing", Usenix Workshop HotCloud 2009.

[8] L. M. Vaquero, L. Rodero and R. Buyya, "Dynamically Scaling Applications in Cloud", ACM SIGCOMM Computer Communication Review, vol. 41, no. 1, (2011), pp. 45-52.

[9] S. Tai, J. Nimis, A. Lenk and M. Klems, "Cloud Service Engineering", Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering, vol. 2, (2010), pp. 475-476.

[10] A. Li et al. Cloudcmp: Shopping for a cloud made easy. In Hot-Cloud, 2010.

[11] H.Wang et al. Distributed systems meet economics: Pricing in the cloud. In HotCloud, 2010.

[12] M. Wachs, L. Xu, A. Kanevsky and G. R. Ganger, "Exertion-based billing for cloud storage access", Proceeding Hot Cloud'11 Proceedings of the 3rd USENIX conference on Hot topics in cloud computing, (2011), pp. 1-5.

[13] O. Krieger et al. Enabling a marketplace of clouds: Vmware's vcloud director. ACM SIGOPS Operating Systems Review, 44(4):103–114, 2010.

[14] E. Elmroth, F. Marquez, D. Henriksson and D. Ferrera, "Accounting and billing for federated cloud infrastructures", In Eighth International Conference on Grid and Cooperative Computing, IEEE, (2009), pp. 268–275.

[15] G. Gousios, C. K. K. Loverdos, P. Louridas, N. Koziris, "Aquarium: An Extensible Billing Platform for Cloud Infrastructures", https://code.grnet.gr/attachments/download/.../Aquarium-paper.pdf.

[16] Danamma M.Bulla et al, / (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 5 (2) , 2014, 1455-1458.

[17] R. M. Piro, A. Guarise and A. Werbrouck, "Price-sensitive resource brokering with the Hybrid Pricing Model and widely overlapping price domains", Research Articles: Concurrency and Computation Practice and Experience, vol. 18, no. 8, (2006), pp. 837–850.

[18] B. Rochwerger, D. Breitgand, E. Levy, A. Galis, K. Nagin, I. M. Llorente, R. Montero, Y. Wolfsthal, E. Elmroth, J. Caceres, M. Ben-Yehuda, W. Emmerich and F. Galan, "The Reservoir model and Architecture for open federated cloud computing", IBM Journal of Research and Development, vol. 53, no. 4, (2009), pp. 4:1-4:11.

[19] Rajesh Bose, Sandip Roy and Debabrata Sarddar, " User Satisfied Online IaaS Cloud Billing Architecture with the Help of Billboard Manager", International Journal of Grid Distribution Computing Vol.8, No.2 (2015), pp.61-78.

[20] Ahmed Mihoob, Carlos Molina-Jimenez and Santosh Shrivastava, "Consumer-centric resource accounting in the cloud",Journal of Internet Services and Applications 2013, 4:8.

[21] ITEA 2 Project 10014 EASI-CLOUDS - Extended Architecture and Service Infrastructure for Cloud-Aware Software.

[22] M. Maurer, V. C. Emeakaroha, M. Risch, I. Brandic, and J. Altmann, "Cost and benefit of the SLA mapping approach for defining standardized goods in cloud computing markets," in International Conference on Utility and Cloud Computing (UCC 2010) in conjunction with the International Conference on Advanced Computing (ICoAC 2010), 2010.

[23] Rajesh Bose, Himadri Biswas, Debabrata Sarddar , Manas Kumar Sanyal, "Cloud Billing & Verification Of Consumed Resources and Storage Spaces by a Cloud User" in International Journal of Applied Engineering Research ISSN 0973-4562 Volume 11, Number 9 (2016) pp 6568-6576.

[24] Debabrata Sarddar, Himadri Biswas and Priyajit Sen, "Safety as a Service (SFaaS) Model - The New Invention in Cloud computing to establish a Secure Logical Communication Channel between Data Owner and the Cloud Service Provider before Storing, Retrieving or Accessing any Data in the Cloud" International Journal of Grid and Distributed Computing, ISSN 2005-4262 Vol. 10, No. 10 (2017), pp.1-20