International Journal of Current Engineering and Technology
E-ISSN 2277 – 4106, P-ISSN 2347 - 5161
©2014 INPRESSCO®, All Rights Reserved
Available at http://inpressco.com/category/ijcet

Research Article

Active Monitoring: ERT Based Load Balancing in Cloud

Priyanka Singh[A] and Shubham Pandey[B*]

[A]Department of CSE,[B]Department of Information Technology, SRM University, Kattankulathur-603023, INDIA

Accepted 11 April 2014, Available online 15 April 2014, **Vol.4, No.2 (April 2014)**

Abstract

In this paper we have proposed and implemented an active monitoring method for Cloud Infrastructure. Cloud computing environment is a model for delivering information technology services in which resources are retrieved from the internet through web-based tools and applications, rather than a direct connection to a server. Users can set up and boot the required resources and they have to pay only for the required resources. Thus, in the future providing a mechanism for efficient resource management and assignment will be an important objective of Cloud computing. In this project we propose an Expected Response time (ERT) based load balancing and dynamic scheduling approach that allocate resources based on the load of Virtual Machines (VMs) on Infrastructure as a service (IaaS). This method enables users to dynamically add and/or delete one or more instances on the basis of the load and the conditions specified by the user. Our objective is to develop an effective load balancing algorithm using Virtual Machine Monitoring to maximize or minimize different performance parameters(throughput for example) for the Clouds of different sizes (virtual topology de-pending on the application requirement).

Keywords: *Load Balancing, Virtualization, IaaS Cloud, Cloud Computing, ERT, Resource Allocation, Distributed Processing.*

1. Introduction

In this paper we have implemented a cloud application platform which enable the whole infrastructure having both virtual and physical components to appear as a single, highly extensible and reliable system, which has the ability to operate itself and adaptable to the failures and changing load (A. Haider *et al*,2009). It can also be provisioned either by an application or simply by a web request (Rochwerger B. *et al*, 2011).

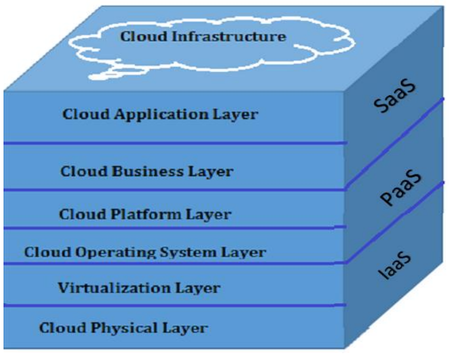


Fig 1. Cloud Infrastructure Model

Here we have proposed a cloud infrastructure model as shown in figure 1.

*Corresponding author **Shubham Pandey** and **Priyanka Singh** are PG scholars

Cloud computing (http://en.wikipedia.org/wiki/Cloud_computing) operates in three major layer- Cloud infrastructure (commonly known as Infrastructure as a service or IaaS), cloud application platform (commonly known as platform as a service or PaaS) and cloud application (commonly known as Software as a service or SaaS). We have proposed a cloud service model (Eric A. *et al*, 2010), here that is encapsulated using several layers as:

1.1 *Cloud Application Layer:* As a part of Software as service it provides various applications and policies to cloud consumers.
1.2 *Cloud Business Layer:* it provide the vision for various business objective, cloud enabled business application, software, policies for business deployment and various associated analysis framework.
1.3 *Cloud Platform Layer:* as a part of cloud application platform (PaaS) it abstract a middleware between cloud consumer and cloud platform for offering various technical solutions, application to consumers based on their demand.
1.4 *Cloud Operating System Layer:* as a part of cloud application platform it provide a model for application virtualization, cloud software and hardware provisioning, billing, task flow execution load balancing and monitoring for various event in cloud environment.
1.5 *Virtualization Layer:* as a part of Infrastructure as a service virtualization layer provide a core virtualize

infrastructure and potentially useful foundation for cloud computing to offer service on-demand.

1.6 *Cloud Physical layer:* as a part of Infrastructure as a service cloud physical layer will provide various physical computing component, storage and network resources and various measurement tools for the security that are virtualized and cloud enabled to support cloud environment.

By taking all these factors we have implemented an expected response time based approach for analysing and monitoring the behaviour of the virtual machines at various workloads. We are using this algorithm to distribute our workload across several virtual machine to avoid the performance bottleneck and to increase the throughput.

2. Proposed Model

2.1 Existing VM Load Balancer

Virtual machine enables the abstraction of an Operating System and Application running on it from the hardware (OpSource Cloud). The interior hardware infrastructure services interrelated to the Clouds are modeled in the simulator by a Datacentre element for handling service requests. These requests are application elements sandboxed within VMs, which need to be allocated a share of processing power on Datacentre's host components (Endo P. *et al*, 2011). Datacentres object manages the datacentre management activities such as VM creation and destruction and does the routing of user requests received from User Bases via the Internet to the VMs (Amazon Virtual Private Cloud).The Data Centre Controller, uses a VM Load Balancer to determine which VM should be assigned the next request for processing.

2.2 Proposed VM Load Balancing Algorithm

To provide the higher degree of satisfaction in the form of throughput and CPU Utilization we have proposed and implemented an approach for load balancing based upon the expected response time of VM. By using this approach we are distributing our workload across several VM for a successful task flow execution as shown in Figure 2.

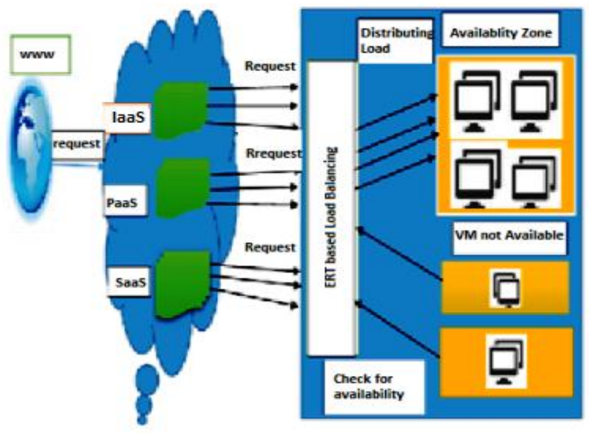


Fig 2. ERT Based Load Balancer Architecture

2.3 Expected Response Time (ERT) based Load Balancing Algorithm

The Proposed Expected Response Time (ERT) based Load balancing algorithm is divided into three parts.

The first phase is the initialization phase. In the first phase, the expected response time of each VM is to be find. In second Phase find the efficient VM, in Last Phase return the ID of efficient VM.

The working of algorithm is shown using the Flow Chart shown in figure 2.

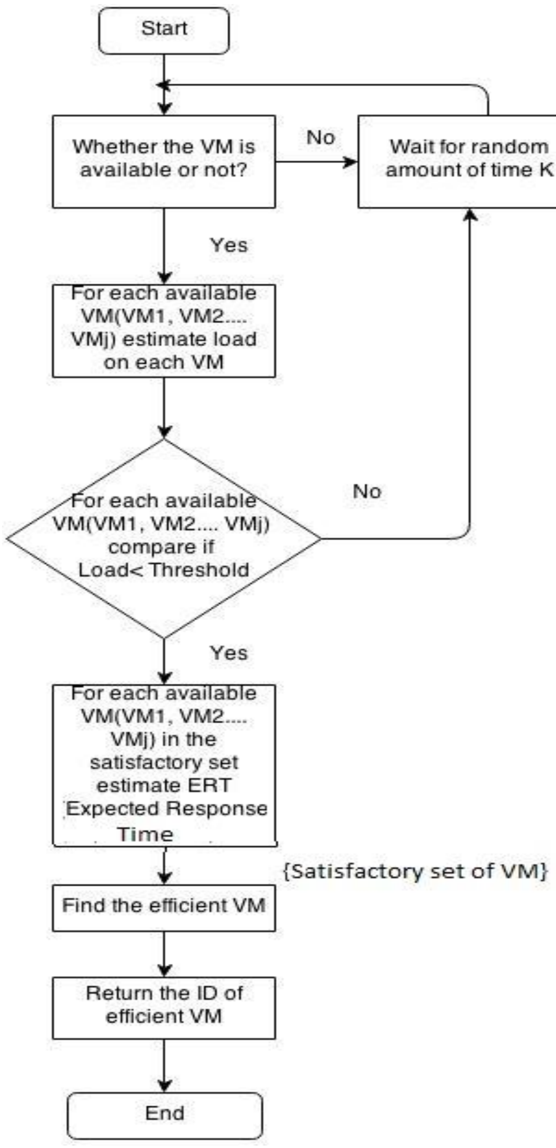

Start
Whether the VM is available or not?
No
Wait for random amount of time K
Yes
For each available VM(VM1, VM2.... VMj) estimate load on each VM
For each available VM(VM1, VM2.... VMj) compare if Load< Threshold
No
Yes
For each available VM(VM1, VM2.... VMj) in the satisfactory set estimate ERT Expected Response Time
{Satisfactory set of VM}
Find the efficient VM
Return the ID of efficient VM
End

Fig. 3. Flowchart for ERT based Load Balancing Algorithm.

To distribute the load across several nodes (VMs) we have presented the algorithm as shown in flow diagram represented in figure 3. We are assuming the threshold as a fraction from 0 to 100 where we are taking the default value as 10. Because we are assuming that none of the virtual machine will be idle after initialization. The steps for the computation to achieve the load balance are as follows. We are taking some assumption as:

VM_i= ith virtual machine

N= Total number of virtual machine

Available set of VM = {ϕ}

Satisfactory set of VM= {ϕ}

Algorithm 1: ERT Based Load Balancing

```
1.    In the initialization phase we compute whether the VM is available or not.
  1.1. If the machine is available then add it to the set of available VMs. And GOTO step 2.
  1.2. Else wait for random amount of time **k** and GOTO step1.1.
2.    For each VM in set of available VMs estimate the load on each machine.
  2.1. Now for each VM compare the load with the threshold.as we already have discussed that the default threshold is 10.
  2.2. If load ($VM_i$) < Threshold then add the $VM_i$ to the satisfactory set of VM.
  2.3. Else GOTO 1.2
3.    Now for each { $VM_1$,$VM_2$,….. $VM_P$ } in the satisfactory set of VM find the expected response time.
4.    Now find the efficient VM based upon the constraints given by the user (based upon user request) and the corresponding ERT.
5.    Return the ID of the VM.
```

2.4 Algorithm to calculate Expected Response Time

In this approach based upon the prior response time estimated from the previous requests we are analyzing the ERT for specific VM. Concept of supervised Learning is used in this estimation to incorporate more reliable outcomes.

Algorithm 2: Expected Response Time (ERT)

Let $\mathbf{T_r}$ be the actual response time for the previous request, N be the total number of VM and Wi represent load on the $\mathbf{V_i}$ (ith virtual machine) Then for nth number of request from the consumer the expected response time **μ (t)**can be expressed as

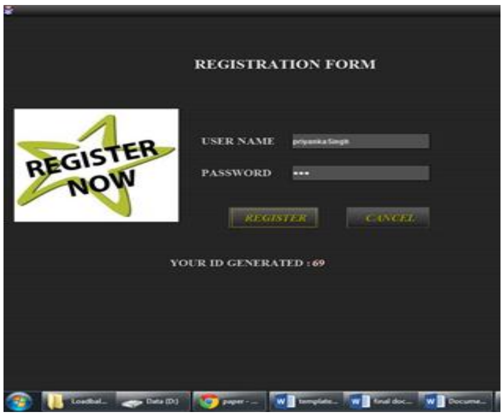
$$\mu(t) = \frac{N.Tr}{\sum_{i=1}^{n} W}$$

Where we are assuming that at any time on the ith node $\mathbf{W_i \neq 0}$.

Modules

The modules that we have implemented are:

3.1 Bucket Creation: Browser allows to easily create Buckets in all regions supported by commercial cloud. Once created a new bucket, consumer can create virtual folders to organize files, and upload and download files to and from commercial cloud database as shown in Figure 4.



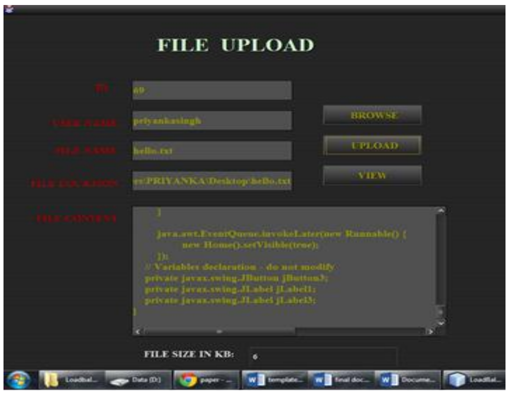
REGISTRATION FORM

USER NAME: priyanka Singh
PASSWORD: ***

YOUR ID GENERATED : 69

Fig. 4. Bucket Creation Module

3.2 Uploading Instances: VM import/export enables to easily import virtual machine image from existing environment to commercial cloud instances and export them back to on-premises environment as shown in figure 5 and 6. This offering allows to leverage an existing investment in the virtual machine that built to meet IT security. Configuration management and compliance requirements by seamlessly bringing those virtual machines into commercial cloud as ready to use instances. Consumer can easily export imported instances back to an on-premises virtualization infrastructure, allowing to deploy workload across IT infrastructure.



FILE UPLOAD

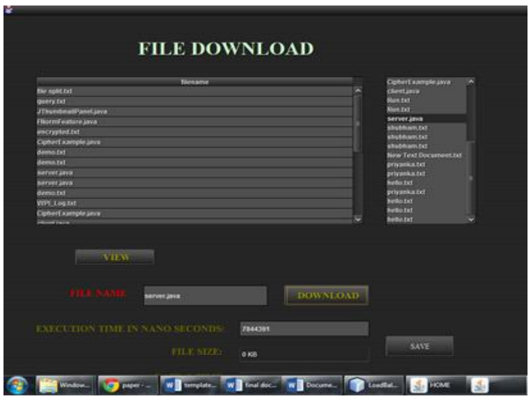
ID: 69
USER NAME: priyankasingh
FILE NAME: hello.txt
FILE LOCATION: e:\PRIYANKA\Desktop\hello.txt
FILE CONTENT:

```
    }

    java.awt.EventQueue.invokeLater(new Runnable() {
        new Home().setVisible(true);
    });
 // Variables declaration - do not modify
 private javax.swing.JButton jButton3;
 private javax.swing.JLabel jLabel1;
 private javax.swing.JLabel jLabel3;
}
```

FILE SIZE IN KB: 6

Fig 5. Instance Uploading Module



FILE DOWNLOAD

filename
file split.txt
query.txt
JThumbnailPanel.java
FNormFeature.java
encrypted.txt
CipherExample.java
demo.txt
demo.txt
server.java
server.java
demo.txt
WPI_Log.txt
CipherExample.java

CipherExample.java
client.java
[illegible].txt
[illegible].txt
server.java
shubham.txt
shubham.txt
shubham.txt
New Text Document.txt
priyanka.txt
priyanka.txt
hello.txt
priyanka.txt
hello.txt
hello.txt
hello.txt

FILE NAME: server.java
EXECUTION TIME IN NANO SECONDS: 7844391
FILE SIZE: 0 KB

Fig 6. Instance Downloading Module

3.3 Monitoring Instances: ERT based load balancer provides a reliable, scalable, and flexible monitoring solution that can start using within minutes. No longer need to set up, manage, or scale own monitoring systems and infrastructure. Using this approach admin can easily monitor the istances.it also lets admin to programmatically retrieve consumer instances. It has the features for monitoring data, view graphs, and set alarms to help you troubleshoot, spot trends, and take automated action based on the state of Cloud environment. The Monitoring Module is coded using Java language which is such as:

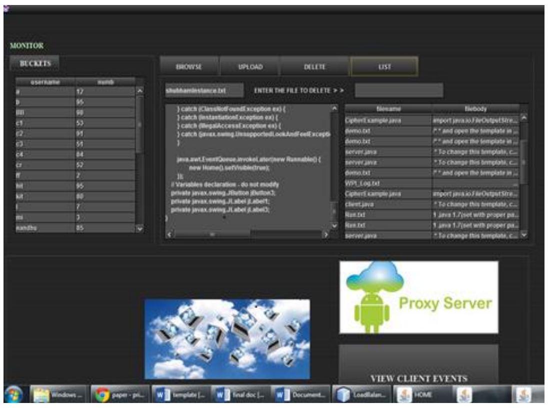
```
public class monitring extends javax.swing.JFrame
{   public monitring() {
      try {
            Class.forName(com.mysql.jdbc.Driver);
Connection con =
DriverManager.getConnection(jdbc:mysql://localhost:330
6/loadbalancing, root, root);
            st = con.createStatement();
        } catch (Exception e)
 {
           e.printStackTrace();
        }
        initComponents();
       }



 private void
jButton5ActionPerformed(java.awt.event.ActionEvent
evt)
 {
         try {
           FileInputStream fis = null;
                 String filebody=jTextArea2.getText();
                String filename=jTextField1.getText();
FileWriter fstream = new
FileWriter(proxyserver\\+filename);
  BufferedWriter out = new BufferedWriter(fstream);
  out.write(filebody);
               System.out.println(filebody);
                    File file = new File(jTextField1.getText());
                    fis = new
FileInputStream(jTextArea2.getText());
                    String str=jTextArea2.getText();
            st.executeUpdate(insert into upload
values('admin','admin','+filename+','+filebody+','fs','time'))
;
             FileOutputStream fos = new
FileOutputStream(F:\\+filename);
               fos.write(str);
                   fos.close();
            out.close();
            JOptionPane.showMessageDialog(this, File Has
Been Uploaded Successfully);
        }
 catch (Exception ex) {
Logger.getLogger(upload.class.getName()).log(Level.SEV
ERE, null, ex);
        }
    }
```

```
    private void
jButton6ActionPerformed(java.awt.event.ActionEvent
evt) {
        proxyserver pr=new proxyserver();
        pr.setVisible(true);
    }
    private void
jButton7ActionPerformed(java.awt.event.ActionEvent
evt) {
   clientevent ce = null;
        try {
            try {
                ce = new clientevent();
            }
 catch (Exception ex)
{
Logger.getLogger(monitring.class.getName()).log(Level.S
EVERE, null, ex);
            }
        }
```

In this way using this module Admin can monitor log event as well as characteristic of virtual machines on several instances as shown in Figure 7.



username	numb
a	17
b	95
bbb	98
c1	53
c2	91
c3	55
c4	84
cr	52
ff	2
hh	95
kk	80
l	7
mi	3
nandhu	85

shubhaminstance.txt

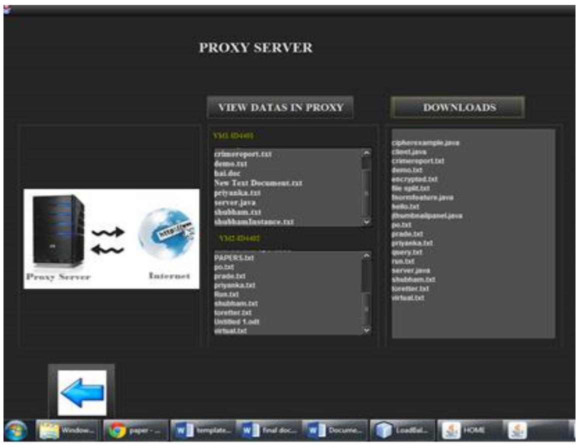
```
    } catch (ClassNotFoundException ex) {
    } catch (InstantiationException ex) {
    } catch (IllegalAccessException ex) {
    } catch (javax.swing.UnsupportedLookAndFeelExceptio
    }

    java.awt.EventQueue.invokeLater(new Runnable() {
        new Home().setVisible(true);
    });
 // Variables declaration - do not modify
 private javax.swing.JButton jButton3;
 private javax.swing.JLabel jLabel1;
 private javax.swing.JLabel jLabel3;
}
```

filename	filebody
CipherExample.java	import java.io.FileOutputStre...
demo.txt	/* * and open the template in ...
demo.txt	/* * and open the template in ...
server.java	* To change this template, c...
server.java	* To change this template, c...
demo.txt	/* * and open the template in ...
WPI_Log.txt	
CipherExample.java	import java.io.FileOutputStre...
client.java	* To change this template, c...
Run.txt	1 java 1.7(set with proper pa...
Run.txt	1 java 1.7(set with proper pa...
server.java	* To change this template, c...

Fig 7. Monitoring Instance Module

The administrator has the ability to check the availability of virtual machines using ERT based approach as shown in Figure 8.



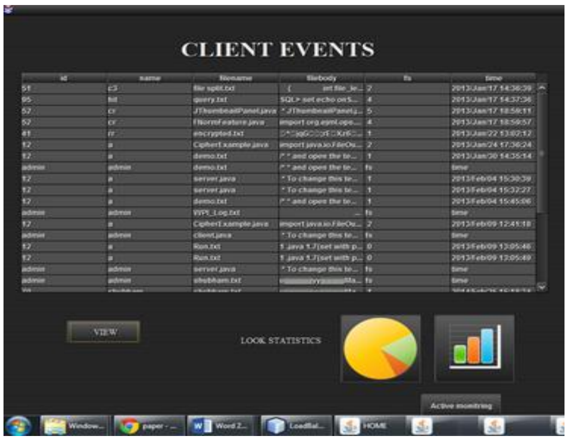
VM1-ID4401
crimereport.txt
demo.txt
hai.doc
New Text Document.txt
priyanka.txt
server.java
shubham.txt
shubhamInstance.txt

VM2-ID4402
PAPERS.txt
po.txt
prade.txt
priyanka.txt
Run.txt
shubham.txt
toretter.txt
Untitled 1.odt
virtual.txt

cipherexample.java
client.java
crimereport.txt
demo.txt
encrypted.txt
file split.txt
fbormfeature.java
hello.txt
jthumbnailpanel.java
po.txt
prade.txt
priyanka.txt
query.txt
run.txt
server.java
shubham.txt
toretter.txt
virtual.txt

Fig 8. Monitoring Availability of VM

Cloud administrator can also monitor the client log event. And the instances imported and exported on the cloud as shown in figure 9.


CLIENT EVENTS

id	name	filename	filebody	fs	time
51	c3	file split.txt	{ int file_le...	2	2013/Jan/17 14:36:39
95	hd	query.txt	SQL> set echo on5...	4	2013/Jan/17 14:37:36
57	cr	JThumbnailPanel.java	* JThumbnailPanel.j...	5	2013/Jan/17 18:58:11
52	cr	FNormFeature.java	import org.ejml.ope...	4	2013/Jan/17 18:59:57
41	rr	encrypted.txt	[illegible]	1	2013/Jan/22 13:02:12
12	a	CipherExample.java	import java.io.FileOu...	2	2013/Jan/24 17:36:24
12	a	demo.txt	/* * and open the te...	1	2013/Jan/30 14:35:14
admin	admin	demo.txt	/* * and open the te...	fs	time
12	a	server.java	* To change this te...	1	2013/Feb/04 15:30:39
12	a	server.java	* To change this te...	1	2013/Feb/04 15:32:27
12	a	demo.txt	/* * and open the te...	1	2013/Feb/04 15:45:06
admin	admin	WPI_Log.txt	...	fs	time
12	a	CipherExample.java	import java.io.FileOu...	2	2013/Feb/09 12:41:18
admin	admin	client.java	* To change this te...	fs	time
12	a	Run.txt	1 java 1.7(set with p...	0	2013/Feb/09 13:05:46
12	a	Run.txt	1 java 1.7(set with p...	0	2013/Feb/09 13:05:49
admin	admin	server.java	* To change this te...	fs	time
admin	admin	shubham.txt	[illegible]	fs	time

Fig 9. Monitoring Client Events

In this way cloud administrator has the ability to monitor and view load statistic as shown in Figure 10 and 11.

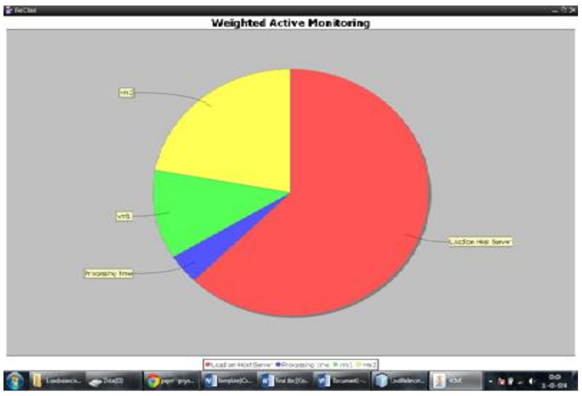


Fig 10. Weighted Active Monitoring

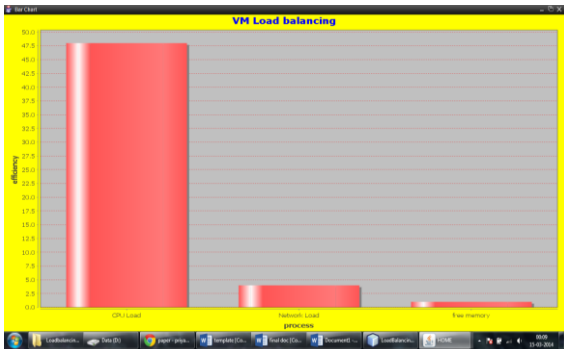


Fig 11. Monitoring the ERT and efficiency of each VM for request

3.4 Performance: The ERT based load balancing approach provide an efficient mechanism to balance the load across several machine. In our experiment we have monitor the performance of virtual machines based upon simultaneous requests as shown in Figure 12. Which leads to the conclusion that this approach provide an optimal solution for the load balancing.

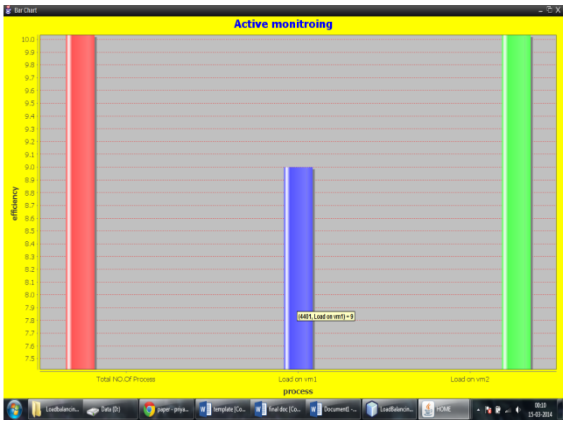


Fig 12. For each request performance of VM

Conclusion

In this paper we have proposed and implemented an ERT based load balancing concept through which the request from the cloud consumer can be handled efficiently and for the simultaneous requests load can be distributed so that to minimize the overhead and to avoid the performance bottleneck at any node. This approach also provide active monitoring for the clients events .admin can monitor all the instances imported and exported by the consumers and also can monitor event log for several consumers at each virtual machine. At last we are analyzing the performance of the virtual machines in terms of efficiency and throughput based upon different load at several time slots. Our future work is to implement ERT based algorithm in hybrid cloud environment. and to monitor the performance of ERT based algorithm in hybrid cloud.

References

Endo, P. T., Palhares, A. V. A., Pereira, N. N., Gonçalves, G. E., Sadok, D., Kelner, J., Melander, B., Mangs, J. E. (2011), Resource allocation for distributed cloud: concepts and research challenges, *IEEE Network Magazine*, vol. 25, pp. 42-46

Haider A., Potter, R., and Nakao, A. (2009), Challenges in Resource Allocation in Network Virtualization, *20th ITC Specialist Seminar*, pp. 18-20

Rochwerger B., Breitgand, D., Epstein, A., Hadas, D., Loy I., Nagin, K., Tordsson, J., Ragusa, C., Villari, M., Clayman, S., Levy, E., Maraschini, A., Massonet, P., Muñoz, H., and Tofetti, G. (2011) ,Reservoir - When One Cloud Is Not Enough, In Proceedings of *IEEE Computer*, Vol. 44, No. 3, pp. 44-51.

Eric A. Marks, Bob Lozano, (2010) *Executive's Guide to Cloud Computing* pp 155-180

Amazon, *Amazon Virtual Private Cloud*. OpSource, *OpSource Cloud*.

Cloud Computing *http://en.wikipedia.org/wiki/ Cloud_computing*