RESOURCE SHARING FOR HYPERVISOR BASED ARCHITECTURE IN VIRTUALIZATION ENVIRONMENT

S.Anthoniraj[1], S.Saraswathi[2]

[1]*Assistant Professor, Department of Information Technology, Paavai College of Engineering, Pachal -637018, Namakkal Dist, India.*

[2]*Associate Professor, Department of Information Technology, Pondicherry Engineering College, Pondicherry 605 014, India*

Abstract

In data administration resource sharing has a crucial liability. At the same time we should be conscious about the enslavement factors. In addition virtual platforms can be demoralized as part of gaining the information about a system. As a result the corruption of files can cause crash status of the system. Traditional systems are followers of this observable fact. It yield to the reality that present mechanisms are not proficient to cover all the dependency factors so that the demand of an innovative structure is important. As a result here we initiate a new system named resource sharing. It can be considered as a customized system that can enhance the needs of end users. Resource sharing can have link establishment between various internal and external systems. The medium of correlation can be an independent factor here. For an internal system it can make use of virtual platform, where as for the external it can make use both the virtual and physical connection establishments. The privacy of a user is considered with higher priority and bound them during the usage. Since the usage is in personalized manner, it would be capable of boosting overall performance of the system. For every system security is a major issue. Data security is ensured in such a way that supervision governance .The main advantage here is it can recover the data from distant location during the disaster of the local system. The whole mechanism is governed by the shell sheltered algorithm. Effective usage of the system can result better performance, security and simplified computation module so that terminologies becomes simpler.

Keywords: *Resource sharing, Virtual Machine (VM), Share Sheltered (SS), DNS (Domain Name System).*

----------***----------

1. INTRODUCTION

Virtualization, in computing, is the creation of a virtual version of something, such as a hardware platform, operating system, storage device, or network resources. With virtualization, several operating systems (OSs) can run in parallel on a single CPU. This parallelism tends to reduce overhead costs and differs from multitasking, which involves running several programs on the same OS. Different types of virtualizations include Hardware virtualization, Desktop virtualization, Software virtualization, Memory virtualization, Storage virtualization, Data virtualization and Network virtualization. In our work, we process with the Hardware virtualization or platform virtualization refers to the creation of a virtual machine that acts like a real computer with an operating system [1].

Operating systems were originally developed to provide a set of common system services, such as I/O, communication, and determined storage, to simplify application programming. With the advent of multiprogramming, this charter expanded to include abstracting shared resources so that they were as easy to use (and sometimes easier) as dedicated physical resources. The solutions that have been proposed to date are either based on hardware partitioning [2][3][4][5], or require developing new operating systems with improved scalability and fault containment characteristics [6][7][8][9]. The Resource sharing framework enables a new paradigm for Internet services. Instead of being fixed to a single location, services can dynamically push parts of their responsibilities out onto Internet computing resources, and even all the way to the client. (i) better end-to-end availability (service-specific extensions running in the client mask Internet or server failures), (ii)better cost-performance (by dynamically moving information closer to clients, network latency, congestion, and cost can all be reduced while maintaining server control), and (iii) better burst behavior (by dynamically recruiting resources to handle spikes in demand). In Section II gives an overview of the prior work. In Section III Explains Share Sheltered Algorithm Implementation in Virtualization Environment, In Section IV contains the Performance Evaluations and we conclude in section V.

2. RELATED WORKS

Propose an efficient consolidation technique for multithreaded workloads through adaptive resource sharing on virtual environments [10]. Efficient consolidation of multi-threaded

workloads requires a detailed understanding of various application characteristics such as performance scaling, inter-thread communications and memory access patterns. An experimental infrastructure enables accurate performance and power evaluation of consolidated multi-threaded workloads and then discusses the performance impact of co-scheduling on virtualized systems. Finally, it shows the performance of individual VMs secluded from each other virtualized system by controlling resource affinities and it present an application classification technique that is able to categorize benchmarks. A benchmark classification technique is presented to classify benchmarks according to their power efficiencies and utilize the classification technique to make design an adaptive resource sharing technique.

Resource virtualization [11] facilitates controlled and precise sharing of underlying physical resources such as processor cycles, memory units, and disk space. With virtualized resources, service instances are not directly deployed on top of specific physical resource instances. But instead are deployed on a collection of virtualized resources including virtualized processor, memory, file system, network, and so on. These virtualized resources are mapped to physical resources transparent to the services deployed in the virtual machine (VM).Realizing this control requires simultaneous handling of three problems: (i) determining the virtual resource configurations, (ii) the mapping of resulting virtual resources to physical resources, and (iii) the mapping of workloads to the virtual resources. The heuristic algorithm proposed here improves the execution time of the virtualization strategy computation even more, allowing the matching to be performed in an online mode.

3. SHARE SHELTERED ALGORITHM IMPLEMENTATION IN VIRTUALIZATION ENVIRONMENT

Now a day we make use of the individual systems for the computational purposes. But we are not aware of the thing that whether its performance and the scale are in the identical level. While developing a system the developer sees the anticipation of numeral users. But sometimes it will make discomfort to the individual user. Here the developer is forced to make applications even though the end user does not seek it. It would affect the overall performance of the system and the wastage of the developed applications. For making various networks, we are depending the same systems. We are attentive about the thing that computers are functionally capable one, so why can't we take their advantage. For creating a network we may have to consider number of individual systems and their individual applications shown in Fig.9, 10.

The resultant would be a network, but it will not be up to the mark that we can do. If we are customizing this existing system and shares the essential details, it would be an easy manageable and secure one. These thoughts lead to the introduction of a new environment later called resource sharing which simplifies the current scenario. Here the design consists of a centralized operating system which is to be referred as resource sharing. It can manage multiple virtual machines including the external systems through local area network. Now we make use of the customization through the application of front and back ends.

The verity can be as providing only the access rights and a further through emerging the right to the management category. We can even enlarge our management system by the use of other existing connection terminologies. While considering the function, the resource sharing provides variety of applications to the systems those having a connection establishment to the centralized system. On the basis of access rights we are able to access the data from the server and to save the changes shown in Fig.1. Like the default one we set the saving destination as server for further usage. If it need to be saved in the client system, the sever has to allot additional privileges on the specified system. If so, the destination can be both client and server out of which one can act as backup.

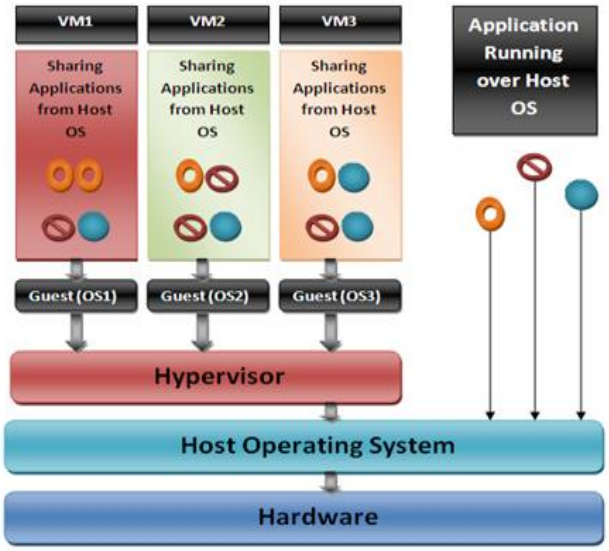

Fig.1. Virtual Application Sharing Architecture

The main advantage of this system is to provide maximum security to the information and to enable recovery during the accidental damage. The overall works are on the basis of resource sharing methodology. This can be made effective though the setting up of a new algorithm which we refer as SS (share sheltered) algorithm. Can be combined along with the maize routing mechanism and to make an optimal performing system which can overcome the current security issues and to solid the security walls

Living state of applications in the systems overheads speed exploitation of system. So utilization can be abridged by the forestalling of unwanted applications. Although the employment of virtual machines helps the system to make use of light weight processes. Here primarily server ensures the accessibility of physical and virtual machine. If so, then the authentication procedure comprises the virtual machines in to service list. During the eve of application request, the server appraises the service list and provides complimentary response if obtainable.

It is able to provide numeral applications exclusive of any interrupts. Here the platform is developed by the employing j-script. So it enables personalized utility supervision. After the tradition of an application, the virtual machine might have to save it. Then the current machine requests to the server for storage space. Followed by that server endorse it. If there arise any fault, then it alerts the overall system and knob the event. If the client desires local storage, then it has to request the server to get further privileges. Hence can provide local and server storage. The server storage can be supplementary used as a backup during the instant of crash. The consequential guarantees data security and best possible performance .Hence it yields to the advancement of a new scalable system.

Share Sheltered Algorithm

V_m=Virtual Machine, Conn=connection, P_m=Physical Machine, AU=Authentication, REQ=Request, A_p=Application Processed=list, S=Save, Retr= Retrieving, App=Application

Share Sheltered Connection

Begin
SSCN()
$\forall V_m \epsilon P_m$
$P_m \leftarrow V_m$ [i] Req
If ((Conn==true) && (P_m ==Active)do
Au ->V_m [i]
P_m IP_List= Add V_m [i] in P_m IP_List
Else if (Conn==false) &&(P_m==Inactive) do
Send Crash Msg "P_m Not Found"
Terminate Conn
End

Share Sheltered Conserving Application Process

Begin
SSCP()
$\forall V_m \epsilon P_m$
$P_m \leftarrow V_m$ [i] Req for App
If ((Conn==True)&&(P_m==Active)) do
If (V_m[i] in P_m IP_List) do
AuV_m[i]
Open App
$S=A_p$
If ((S==Success) &&(S==Complete)) do
$P_m=V_m$[i] =S
else
Send Crash Msg "Application Corrupted"
else
If (V_m[i]! in P_m IP_List) do
Send Crash Msg" IP Not Valid"
Terminate Conn
End
End
End

Share Sheltered Application Retrieving

Begin
SSAR()
$P_m \leftarrow V_m$ [i] Req for Retr (App)
Check V_m[i] in P_m IP_List
AuV_m[i]
If(App==active) do
Use App
else
If((Corrupt==true)&&(App==Inactive))do
Send Crash Msg to V_m[i] to use P_m
Open in P_m
End
End
End

4. PERFORMANCE ANALYSIS OF RESOURCE SHARING APPLICATIONS

Performance of a system explains its require. While considering this, we have to stare on three contingent factors as CPU, RAM, and bandwidth. RAM and bandwidth administrate the performance of CPU. Here the desires of applications can be many, but then also the new system is skilled to withstand in any circumstances according to Fig.2. CPU preserves resource sharing capability and provides optimal performance. Analysis shows the overture of a new time efficient and performable system. So the accompanying integrity on performance scales can be considered.

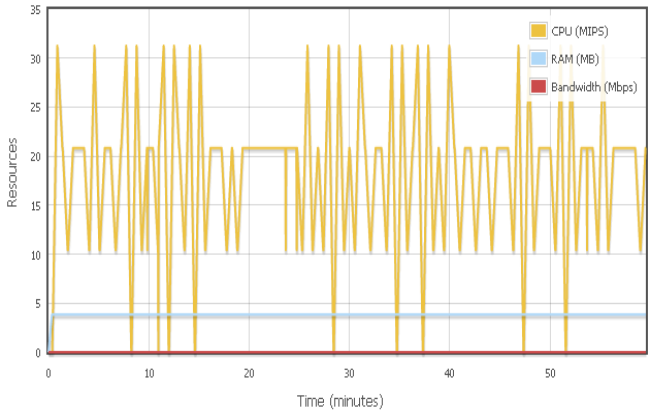

Fig.2. Physical Machine -Resource Utilization Report

A system can be deliberate as an influential one on the basis of execution appraisal. The execution can be further scaled on the basis of start and finish time. Here the measures illustrate that the finishing time is optimal compared to average finish time. So we are capable to resume minimal processing time for each application. Through this the bandwidth expenditure can be reduced and hence can avail more services. The resultant gives a professional system that is talented to supervise number of applications without any time lag.

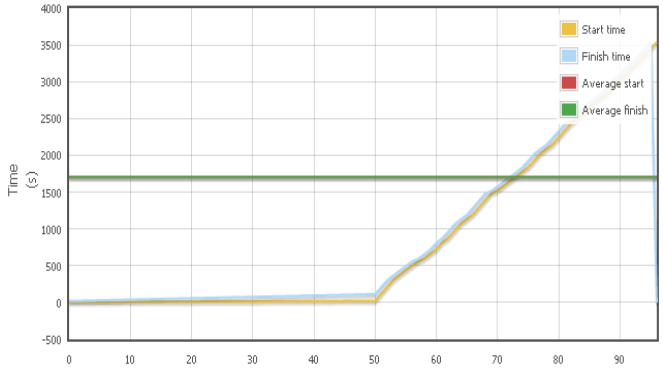

Fig.3. Execution Time for Overall Virtual Machine

Availability of resources facilitates the system to progress its performance. During the distribution process it should be conscious about the time extent. Here our new system is smart in these stages because it takes only requested applications as lively and proceeds further sharing on it. At this point it makes use of the threads in order to get estimated outcome. While examining fig.4, 5, and 6 it is sensible that the resource sharing services are efficient in each of the virtual machines while the time goes on. Hence it exhibits the capability of resource utilization management.

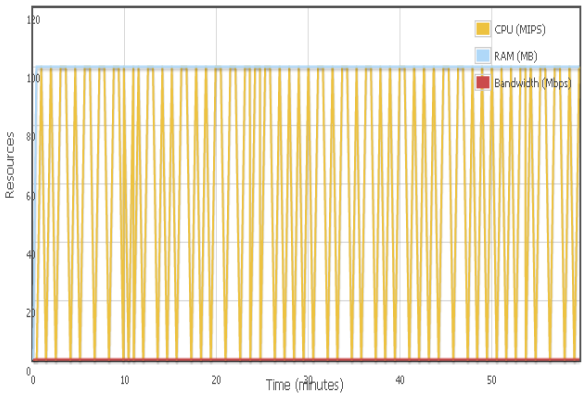

Fig.4. Resource Utilization Report in VM1

The process rate analysis helps the users to make a verdict on the novel system. On the basis of that additional transformations can be entailed on the process execution. Here the main motivation is towards achieving the successful processing with in minimal time rate. So that we are competent achieve optimal performance. According to Fig.7 the error rate can be considered as a trifling aspect for each virtual machine. At the same time the time analysis shows that there does not occurs any delays. On the basis of this we can formulate a self estimation on the excellence standards of our system. And it is clear that the new system is a scalable one.

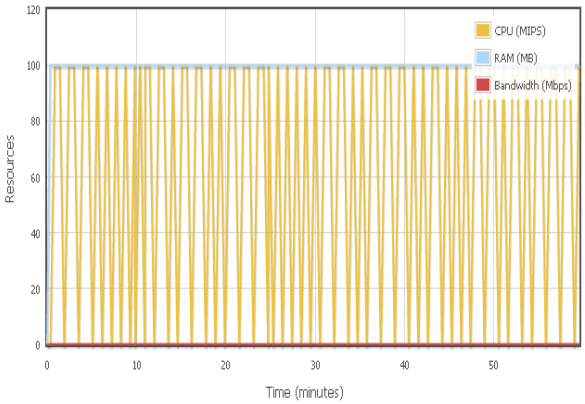

Fig.5. Resource Utilization Report in VM2

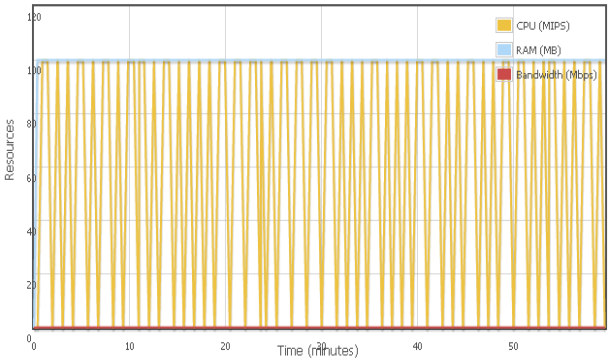

Fig.6. Resource Utilization Report in VM3

Cloudlet ID	STATUS	Resource ID	VM ID	Time	Start Time	Finish Time
0	Success	2	0	15.72	5.1	20.82
50	Success	2	0	92	22.82	114.82
51	Success	2	0	62	116.82	178.82
52	Success	2	0	62	180.82	242.82
53	Success	2	0	92	244.82	336.82
54	Success	2	0	62	338.82	400.82
55	Success	2	0	92	402.82	494.82
56	Success	2	0	60	496.82	556.82
57	Success	2	0	96	558.82	654.82
58	Success	2	0	62	656.82	718.82
59	Success	2	0	60	720.82	780.82
60	Success	2	0	92	782.82	874.82
61	Success	2	0	60	876.82	936.82
62	Success	2	0	62	938.82	1000.82
63	Success	2	0	62	1002.82	1064.82
64	Success	2	0	62	1066.82	1128.82

Fig.7. Result Analysis for each Virtual Machine

```
20.82: [Host #0] Total allocated MIPS for VM #0 (Host #0) is 0.00, was requested 0.00 out of total 1000.00 (0.00%)
20.82: [Host #0] MIPS for VM #0 by PEs (4 * 2400.0).
20.82: [Host #0] Total allocated MIPS for VM #0 (Host #0) is 0.00, was requested 0.00 out of total 1000.00 (0.00%)
20.82: [Host #0] MIPS for VM #0 by PEs (4 * 2400.0).
20.82: [Host #0] Total allocated MIPS for VM #0 (Host #0) is 0.00, was requested 0.00 out of total 1000.00 (0.00%)
20.82: [Host #0] MIPS for VM #0 by PEs (4 * 2400.0).
21.817999999999998: VMClient1: Cloudlet 0 received
21.817999999999998: VMClient1: Sending cloudlet 50 to VM #0
21.817999999999998: VMClient2: Cloudlet 0 received
21.817999999999998: VMClient2: Sending cloudlet 50 to VM #0
21.817999999999998: VMClient3: Cloudlet 0 received
21.817999999999998: VMClient3: Sending cloudlet 50 to VM #0
```

Fig.8. Final Report -Result Analysis

A system with acknowledgement awards fulfillment to end users. Due to lack of error factors the new system is clever to convey uninterrupted services to the client machines. Fig.8 shows the report of the overall progression carried out in the system. From this we can analyze the process overview and the data enslavement factors. So that each individual factors of the system can be practiced in isolation. Finally it produces the carbon copy of the trustful, secure and proficient system.

Resource Sharing Screen Shots:

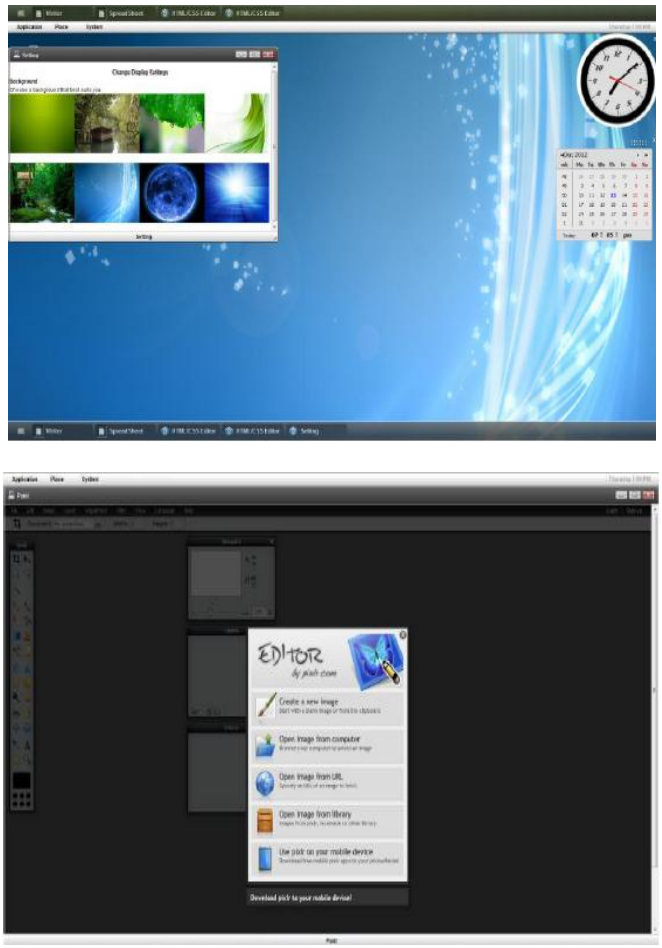

Fig 9 User Friendly Applications in Virtual Machine1

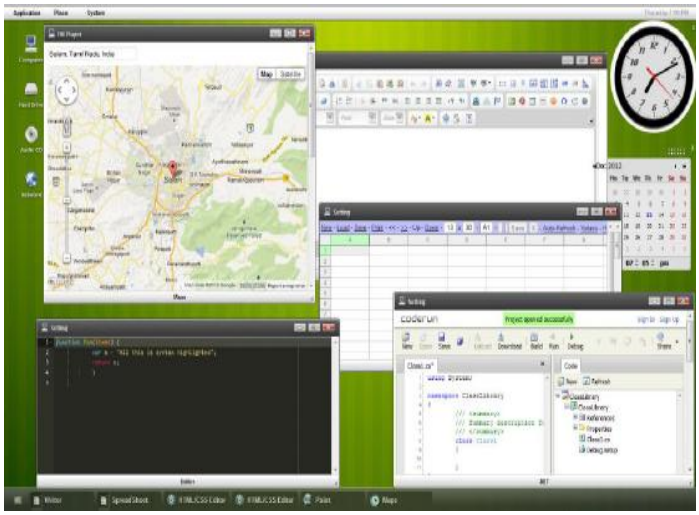

Fig 10 a) User Friendly Application in VM2 b) Various User Application Multithreaded in VM2

5. CONCLUSIONS

In virtual environment, traditional systems are transparent gateways. It intends the users to make use of customized environment. Resource sharing can't be measured as a negotiable strategy. The compromises on it cause dreadful effects on our system. We have to provide with maximum flexibility at users end and strict security on the other side. Management of all these aspects concurrently is a complex assignment. We proposed a new virtualized system which supports the resource sharing technique. Fundamentally availability and security act as the crucial factors. At this stage we have demonstrated the synergy available from exporting traditional operating system functionality to wide area applications. Our prototype implementation, Resource sharing, describes one possible organization of these system services. We show that extending server functionality onto client machines allows for more flexible implementation of name resolution, load balancing, and fault tolerance. We present a methodology for coherently caching program results through the file system, speeding the performance of applications which must repeatedly execute programs with common inputs. More efficiently utilizes system resources for Web server access. Result of evaluations verifies that the new approach is very effective and useful. Currently we look forward on working with virtual environments to make explore on the flexibility and integrity as future enrichment.

REFERENCES:

[1]. S.Anthoniraj, S.Saraswathi and M.Anandraj, Disaster Recovery Planning using fault Tolerant Mechanism, ARTCom 2012, Springer-Verlag Berlin Heidelberg 2012, pp. 215–224, 2012.

[2]. Compaq Computer Corporation. OpenVMS Galaxy. http://www.openvms.digital.com/availability/galaxy.html. Accessed October 1999.
[3]. Sequent Computer Systems, Inc. Sequent's Application Region Manager. http://www.sequent.com/dcsolutions/ agile_wp1.html. Accessed October 1999.
[4]. Sun Microsystems, Inc. Sun Enterprise 10000 Server: Dynamic System Domains.http://www.sun.com/servers/Highend/10000/Tour/domains.html. Accessed October1999.
[5]. Unisys Corporation. Cellular Multiprocessing: Breakthrough Architecture for an Open Mainframe. http://www.marketplace.unisys.com/ent/cmp.html. Accessed October 1999.
[6]. John Chapin, Mendel Rosenblum, Scott Devine, Tirthankar Lahiri, Dan Teodosiu, and Anoop Gupta.Hive: Fault containment for shared-memory Multiprocessors. In Proceedings of the 15th Symposium on Operating Systems Principles (SOSP), pp. 12-25.December 1995.
[7]. Ben Gamsa, Orran Krieger, Jonathan Appavoo, and Michael Stumm. Tornado: Maximizing Locality and Concurrency in a Shared Memory Multiprocessor Operating System. In Proceedings of the 3rd Symposium on Operating Systems Design and Implementation(OSDI), pp. 87-100. February 1999.
[8]. IBM Corporation. The K42 Project. http://www.research.ibm.com/K42/index.html. Accessed October 1999.
[9]. SGI Inc. IRIX 6.5. http://www.sgi.com/software/irix6.5.Accessed October 1999.
[10]. Paweł Garbacki, Vijay K. Naik, "Efficient Resource Virtualization and Sharing Strategies for Heterogeneous Grid Environments", IBM, 2012.
[11]. Can Hankendi Ayse K. Coskun, "Adaptive Energy-Efficient Resource Sharing for Multi-threaded Workloads in Virtualized Systems", 2010.

Web Sites:

[13]. http://-now.cs.berkeley.edu/WebOS.
[14].http://www.vmware.com/pdf/vsphere4/r41/vsp_41_resource_mgmt.pdf