ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 6, June 2015

Software Rejuvenation and Workload Distribution in Virtualized System

Payal Kulkarni

PG Student, Department of Computer Engineering, RMD Sinhgad School of Engineering, University of Pune, Pune, India

ABSTRACT: Cloud computing is a promising paradigm where applications, data, memory, bandwidth and IT services are provided over the Internet. Cloud computing is a technique based on pay per usage model .It allows use of hardware resources by a technique called as virtualization. Virtualization is a technique having one or more VM's which is monitored by virtual machine monitor. In this work we propose a technique to enhance the performance of VM under variable workload conditions. We also propose a fixed timer policy and a methodology for rejuvenating high available virtualized system also for Detection and Estimation of Software Aging,"

KEYWORDS: Time-based rejuvenation, cloud computing, dynamic availability, Virtualization, performance enhancement

I. INTRODUCTION

Cloud computing is an emerging infrastructure paradigm that allows efficient maintenance of cloud with efficient uses of servers. Cloud is a package of services that offers infrastructure, platform, software and data as services. So many researches are being made for improving these flavours of services. Resource availability can be increase due to elastic behavior of cloud. Cloud computing offers their customer to pay only what they use. We can buy any software or service for required period of time on the cloud rather than to purchase a machine for that purpose.

Software rejuvenation [6] deals with software faults. It is called as fault management technique as it prevent occurrence of one or more failure. Software rejuvenation is the concept of gracefully terminating an application and immediately restarting it with a refreshed internal state. Software rejuvenation solves software aging by minimizing failures through periodic, preemptive rollback of applications and server reboot. The most straightforward way is to manually reboot the server periodically, and users often do this. Manual reboot minimizes the above-mentioned problems or prevents them from occurring by not allowing the allocation of internal resources to reach critical levels Virtualization is a technique which allows to instantiate multiple virtual machines (VMs) on top of a physical machine managed by the virtual machine monitor (VMM). Attention has been put in the literature to the possibility of applying rejuvenation to mitigate the effect of software aging in the VMM as software degradation mainly affects long term running software and services. The resource utilization can be maximized by using live VM migration for shifting of VMs when VMMs are rejuvenated.

Contribution of present work is fixed timer policy in which timer is set at the system start up and does not change with respect to workload condition. Another contribution of present work is shifting of application on number of virtual machine to avoid workload.

II. RELATED WORK

Jie Li, and Dengyi Zhang, [7] focus on software rejuvenation ,a proactive fault management technique aimed at cleaning up the system internal state to prevent the occurrence of more severe crash failures .It involves occasionally terminating an application or a system, cleaning its internal state and restarting it.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 6, June 2015

DomenicoCotroneo, and Roberto Natella [6] focus on Software Aging and Rejuvenation in a Soap-Based Server software aging is degradation of software performance or sudden crash or hang of system it can be called as increase of failure rate or decrease of software performance.

Kenichi Kourai, and Shigeru Chiba [11] focus on A Measurement-Based Model which is used for Estimation of Resource Exhaustion in Operational Software Systems basic idea is to periodically monitor and collect data on the attributes responsible for determining the system health.

F. Machida & D. Kim proposed a technique to reduce the system downtime during the rejuvenation. VM and VMM rejuvenations in cloud environments are investigated in [12] through the use of analytical techniques. In [12], the authors highlight the need to rejuvenate the VMs as well as the VMM, measuring the advantages obtained in terms of system availability. Three different rejuvenation schemes are proposed starting from the concept that a VMM rejuvenation affects the running VMs. According to these schemes, each time the VMM is rejuvenated the running VMs can be suspended, rebooted, or migrated.

Fumio Machida, Dong Seong Kim [10] focus on Analysis of Service Availability for Time-Triggered Rejuvenation Policies which consist of two policies .The main goal of time-based rejuvenation models is to find an optimal rejuvenation timer that allows minimizing some objective functions. Usually, the timer is set at system start-up and it does not change with respect to the system dynamics (e.g., system workload variations).

We refer to such kind of approach as fixed timer policy. Another contribution of the present work is the specification of a time-based policy adapting the rejuvenation timer to the VMM conditions, taking into account its workload and age (variable timer policy). The effectiveness of the proposed modelling technique is demonstrated through a numerical example based on a case study taken from the literature. It shows how the proposed variable timer policy outperforms the fixed one in terms of improved system availability also varying the way failure rates are affected by the workload.

M. Neuts focus on Probability Distributions of Phase Type a phase-type distribution is a probability distribution constructed by a convolution of exponential distributionsit results from a system of one or more inter-related Poisson processes occurring in sequence or phases. The sequence in which each of the phases occurs may itself be a stochastic process. The distribution can be represented by a random variable describing the time until absorption of a Markov process with one absorbing state. Each of the states of the Markov process represents one of the phases. It has a discrete time equivalent the phase type distribution. The set of phase-type distributions is dense in the field of all positive-valued distributions, that is, it can be used to approximate any positive-valued distribution .Consider a continuous-time Markov process with $m+1$ states, where $m \geq 1$, such that the states 1,...,m are transient states and state 0 is an absorbing state.

AutranMacêdo, Taís B [5] have proposed work on memory related aging effects. The authors have explained how memory management works inside application process, focusing on two memory problems that cause software aging: fragmentation and leakage. Here they have explained the procedure of memory-related software aging focusing on a real and widely adopted memory allocator and presented an experimental study that illustrates how memory fragmentation and leakage occur and how they accumulate over time in order to cause system aging-related failures.
Fumio Machida et al. have presented the issues of performability management in a virtualized data center (VDC) that hosts multiple services using virtualization. Performability is a concept of a mixed metric of performance and availability. The users of a VDC generally request a certain level of application performance in a service level agreement (SLA). VDC providers need to decide an optimal server configuration and management operations for guaranteeing application performance and maximizing the availability. They have focused on placement algorithm of VMs and rejuvenation schedules for VMs and VMM in a VDC.

III. PROBLEM STATEMENT

Aim of this work is to propose a new innovative approach to model software aging in cloud system and also in LAN network and we propose a technique to model and to evaluate the VMM aging process and to investigate the optimal rejuvenation policy that maximizes the VMM availability under variable workload conditions.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 6, June 2015

IV. PROPOSED SYSTEM

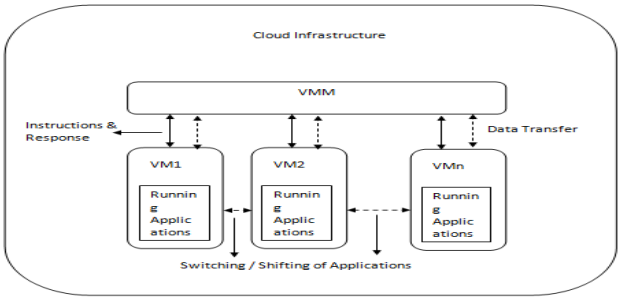

Figure: 1 System Design

A cloud infrastructure is composed of network-connected cloud nodes coordinated by a cloud management system. A cloud node can be a workstation or a multicore system, a cluster, or also a data center, implementing a stand-alone administrative domain with its management system. To improve node availability, software rejuvenation
Policies can be adopted.
Fig 1 consists of mainly two modules/parts which are again divided into sub modules.
These main two main modules are,

1. VMM
2. Node

VMM is a virtual machine monitor, having higher level of physical machine. But in actual implementation, VMM is a virtual machine residing on a physical machine. Hence our application can consist of a VMM application located on physical machine, for which, here after, we are called as VMM manager or only as manager. The node is nothing but a machine which is under monitoring by VMM manager it must be workstation such that a physical machine connected in network. Hence, nodes are physical machines, might be slaves or clients connected in network, either internet or intranet (lan), and manager becomes their server. As workload occurs in the VM 1 instruction is given to VMM for monitoring application VMM then gives response to the vm1 for shifting application form vm1 to VM2.

A .Technique used

- Fixed timer policy

The timer is set at system start-up and it does not change with respect to the system dynamics (e.g., system workload variations).

- Measurement based

The basic idea is to periodically monitor and collect data on the attributes responsible for determining the system health

- Model based

  The model-based approach focuses on analytical models representing the system behaviour to investigate the costs/ benefits associated with rejuvenation.

B.Proposed algorithm

Proposed algorithm steps are as below:

1. While implementing proposed algorithm, it would be assumed that all data was stored at central server.
2. Get details of virtual machine a (vm[a])
3. Get total running applications on vm[a]
4. Count total memory occupied by all running applications on vm[a] and its status
5. If status is running then jump to step 5 and if status is not responding jump to step 10
6. Consider/assume threshold memory
7. If memory occupied on VM [a] is greater than threshold memory, then find out the application occupying max memory

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 6, June 2015

8. Initiate look method to find any other VM , called as VM [b] in same network having (total memory occupied + memory of considered application on VM [a]) less than its threshold memory
9. If found, send 'application close' command to VM[a] and 'application start' command to VM [b] (shift application from VM[a] to VM [b])
10. If not found, send instruction 'application close' on VM [a]
11. If application was not responding, send instruction 'application close' on VM [a] to forcefully close the application
12. Repeat steps 1 to 8 after pre- defined period

C.Algorithm_VM check

1. Enter vm check state at *tp (periodic time)*
2. If check=true, while(i<n) { vmm(app[i]) = vm(app[n])} (where n=total number of running applications and app[i]=i^{th} running application)
3. While(i<n) { mem[tot]=mem[tot] + mem[i] }
4. Memth = (vm_config/100)*per) (at initial value of per should be considered as 5%)
5. If (mem[tot]<memth)
    a. { if (look(vm(otthr) =true)
    b. {while(i<n) { if (((mem[tot]) - (mem[app[i]))>memth)
    c. { m(othr){ (app[i]} = vm{ (app[i] }}}
6. .if ((mem[tot]<memth) and (vm(target)==0))
    a. { while(i<n) { if (((mem[tot]) - (mem[app[i]))>memth)
    b. {close(app[i] }}}

D.Algorithm-performance monitoring

1. Read task manager of machine, for understanding, number of applications running at given time, memory used by them, and status of applications such that whether they are running or get hanged.
2. Check memory space available on each drive, to check whether that drive is running out of memory.
3. Try to distinguish between network applications, such that the applications which are used by network users and local applications such that application which are accessed only on local machine.
4. Send all above collected information to vmm manager for monitoring.
5. Check for any instructions from vmm manager and follow them.
6. All these tasks are performed at regular intervals set by vmm.

E. Mathematical model

- Identify the cloud infrastructure
  c= {c1, c2, c3….}
- Identify the virtual machine monitors (vmm)
  v= {vmm1, vmm2, vmm3….}
  v € c (v is a subset of c, such that cloud contains various virtual machine monitors)
- Identify the virtual monitors
  m = {vm1, vm2, vm3 …}
- Identify the running applications
  a = {a1, a2, a3….}
- Identify the instructions

i= {i1, i2, i3…}

- Evaluate the algorithm

algo= {al1, al2, al3}

A1) analysis virtual machine:

Input: a set of running applications on vm,
Output: status of vm in terms of memory and applications state

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 6, June 2015

$m_{tot} = m(a_1) + m(a_2) + m(a_3) + \ldots + m(a_n)$

A2) load balancing

Input: output of al1
Output:instruction to vm& load balancing
Let,
Identify the heavy application of vm
a_h = {set of heavy applications} = {a_{h1}, a_{h2}, a_{h3} . . . }
where, a_h € a
If (application was consuming more memory) then instruct shift

A3) rejuvenation

Input: an output of algorithm al1
Output:either application was terminated or restarted
let,
Identify the failure applications
A_f = {set of failure application)
If (application was not responding) then instruct terminate

[4.6] practical work

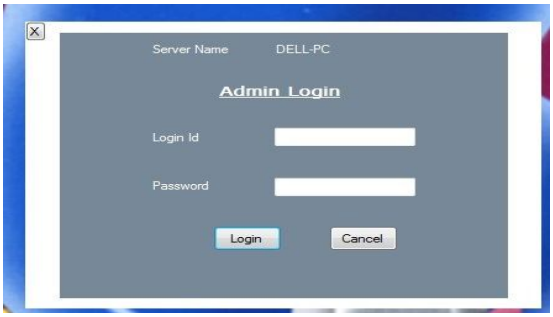

Figure: 2 admin login

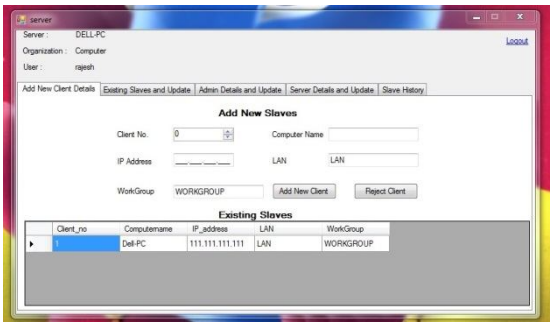

Figure: 3 server view1

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 6, June 2015

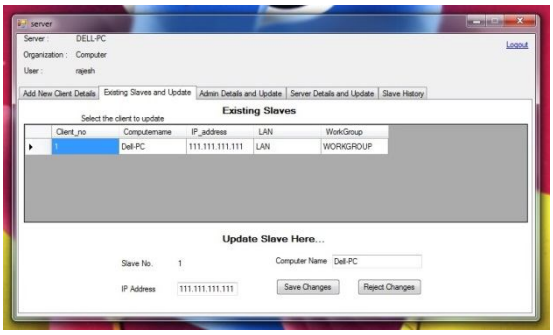

Figure: 4 server view 2

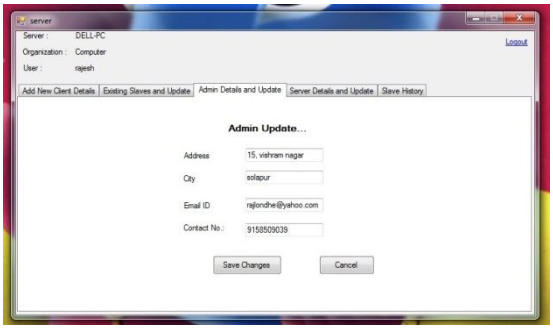

Figure: 5 server view 3

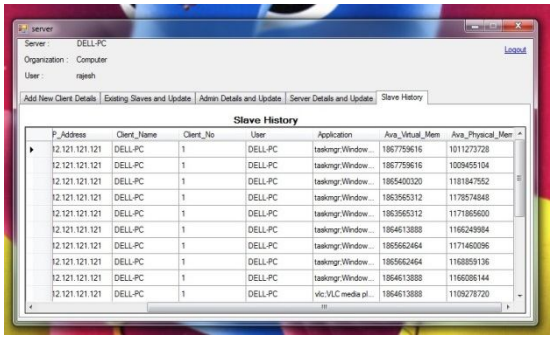

Figure: 6 server view 4

[5] Result

Fig 7 shows graph for system availability at initial time period system is highly available .Fig 8 shows graph for memory Occupation which shows that as soon as memory used become more than threshold value one of the application get closed and used memory get back to under threshold value.

`

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 6, June 2015

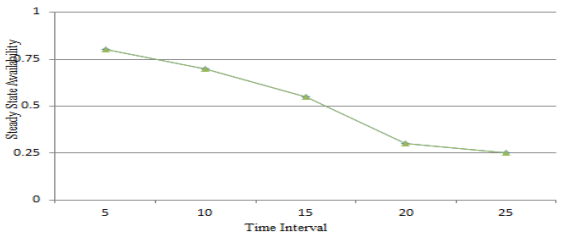

Figure: 7 graph for system availability

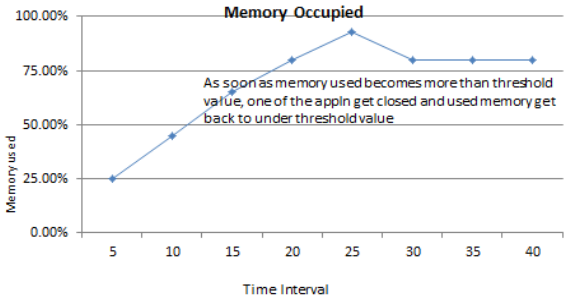

Figure: 8 graph for memory occupation for system

IV. CONCLUSION AND FUTURE SCOPE

In this paper we have discussed how performance of the system can be increased in case of heavy workload using vmm .we also deal with software rejuvenation, a specific form of environment diversity that is gaining importance as an effective preventive maintenance technique. The main contribution of our work is a measurement-based model that integrates the effect of system workload on operating system resources and an approach to investigate its effect on software aging. Since many studies have suggested strong correlations between workload and system reliability/ availability, this model is an improvement over the purely time-based model. We have also distinguished relation between the system workload and resource exhaustion.

We are trying to extend proposed technique in 1) along with wired and wireless LAN i.e. In intranet, in future; we are going to try to implement this application in internet environment. 2) Implementation of application in internet i.e. in global network which does not restrict by any boundaries and hence it was becomes possible to monitor and take care of any machine from anywhere.3) also, along with solution to software failures, hardware failures detection and their solution are to be included in future

REFERENCES

[1]workload-based software rejuvenation in cloud systems “IEEE transaction on computer 2013”
[2] Michael Grottke, RivalinoMatias Jr., and Kishor S. Trivedi, “The Fundamentals of Software Aging," IEEE, 2008
[3] DomenicoCotroneo, Roberto Natella, , Roberto Pietrantuono, and Stefano Russo, " A Survey of Software Aging and Rejuvenation Studies," ACM, Vol 5
[4] "Software Aging and Rejuvenation," Wiley & Sons, 2008
[5] AutranMacêdo, Taís B. Ferreira, and RivalinoMatiasJr, "The Mechanics of Memory-Related Software Aging," IEEE, 2011
[6] DomenicoCotroneo, and Roberto Natella, “Monitoring of Aging Software Systems affected by Integer Overflows," IEEE, 2012
[7] Kehua Su, Hongbo Fu, Jie Li, and Dengyi Zhang, "Software Rejuvenation in Virtualization Environment," IEEE, 2011
[8] JyotiprakashSahoo, SubasishMohapatra and, Radha Lath, "Virtualization: A Survey On Concepts, Taxonomy and Associated Security Issues," IEEE, 2010

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 6, June 2015

[9] Kenichi Kourai, and Shigeru Chiba, "Fast Software Rejuvenation of Virtual Machine Monitors," IEEE, Vol 8, No 6, 2011
[10] Fumio Machida, Dong Seong Kim, Jong Sou Park, and Kishor S. Trivedi, "Toward Optimal Virtual Machine Placement and Rejuvenation Scheduling in a Virtualized Data Center," IEEE, 2008
[11] Kenichi Kourai, and Shigeru Chiba, "A Fast Rejuvenation Technique for Server Consolidation with Virtual Machines, " IEEE, 2007
[12] Fumio Machida, Jianwen Xiang, Kumiko Tadano, and Yoshiharu Maeno, "Combined Server Rejuvenation in a Virtualized Data Center"

BIOGRAPHY

Payal Kulkarni Research Scholar RMD Sinhgad School of Engineering Warje, Pune, University of Pune. Received B.E. in Information technology from Information Technology Department of MastyodariShikshanSanstha's College of Engineering, Jalna fromDr.BAMU. University, Aurangabad.Currently persuing M.E. in computer engineering from RMD Sinhgad School Of Engineering Warje, Pune, India.