Memory Management by Inter-positioning the Balloon Mechanism in Virtualization

Tehami Mustaasam, Shahid Naseem, Muhammad Waqas, Mushtaq Niazi, Ayesha Iqbal, M.Salman Akram

Abstract----Memory over commitment is a well-known problem in operating system virtualization. A virtual machine utilizes its maximum memory when it is not over committed due to the memory pressure of its high workload. To control memory pressure in virtualization a Ballooning mechanism is used, however it is not helpful to deal with memory over commitment. Therefore it is required to modify this mechanism to deal with memory over commitment. This paper introduces interposition phenomenon to extend the ballooning mechanism that deals with the issue of memory over commitment in virtualization. The proposed extended ballooning mechanism helps the virtualized operating system to deal with the memory over commitment.

Introduction

In the past decade, virtual machines have become popular as a technology for multiplexing operating systems. However some challenges are faced in CPU virtualization but now there are both software and hardware CPU virtualization as mentioned in (Micah Dowty, 2008). Operating system virtualization provides us many benefits like transparent migration of applications and enhancement of system security. Virtual machines are good abstractions of server workloads as to encapsulate the running state of system both on user end that are applications and on kernel end that are operating system services.

Many useful resources in servers are often not properly utilized and virtualization can be used to consolidate different physical servers so that resources are more utilized. Memory management in virtualization is a major task as a virtual machine is provided its maximum space for running by the system when memory is not facing over commitment.

If a virtual machine creates the illusion that there are more instances or bandwidth of a resource than actually available, that resource is considered to be over committed or oversubscribed. For example, if three uniprocessor servers are virtualized as guests and consolidated onto one dual-processor server, the CPU resource is considered over committed as mentioned in (Magenheimer, 2008).

- *Tahami Mustaasam is with Computer Sciences Department as research fellow at NCBA&E, Lahore, Pakistan. His area of interest is artificial intelligence. He is working as Software Quality Assurance Engineer at Pheedra Solutions, Lahore, Pakistan (raj.sahir091@gmail.com).*
- *Shahid Naseem is Assistant Professor (IT) with University College of Engineering, Sciences & Technology, Lahore Leads University, Lahore, Pakistan (shahid.naseem@gmail.com).*
- *Muhammad Salman Akram is with Computer Sciences Department as research fellow at NCBA&E, Lahore, Pakistan. His area of interest is artificial intelligence. He is working as Computer Expert at NAB, Lahore, Pakistan (msam128@gmail.com).*
- *Ayesha Iqbal is with Computer Sciences Department as research fellow at NCBA&E, Lahore, Pakistan. Her area of interest is cloud computing. She is working as Assistant Manager (HRM) at TEVTA Secretariat, Lahore, Pakistan (isha19_2005@hotmail.com).*
- *Mohammad Waqas is with Computer Sciences Department as research fellow at NCBA&E, Lahore, Pakistan. His area of interest is cloud computing. He is working as Software Engineer at Bilytic (Pvt.) Lahore, Pakistan (waq_as@hotmail.com).*
- *Mushtaq Niazi is with Computer Sciences Department as research fellow at NCBA&E, Lahore, Pakistan. His area of interest is database (th3khan@gmail.com).*

Ballooning mechanism in (Waldspurger, 2010) is used for memory management by virtualization EXS Server software that allocates and de allocates the memory according to the condition of the memory pressure, it uses a balloon module that inflate if it has space for further processing and it deflates when the memory pressure is high.

Interposition allows virtualization to link between a virtual machine and physical hardware, it usually works as a execution check point in operating system virtualization as in (Micah Dowty, 2008). It check out the objects and the

similarity and differences between them and upon the similarities it allows the objects to proceed further and blocks the distinct objects for a while so the the similar objects will complete there processes as described in (Oren Laadan, 2010).

Literature Review

Waldspurger in (Waldspurger, 2010) describes the ballooning mechanism that software ESX Server uses to manage the memory resource in virtualization by a balloon module. A balloon module is loaded into the guest operating system as a kernel service. It has no external interface within the guest and communicates with the server through private channel. Whenever the server reclaim memory the balloon starts inflate by allocating pinned physical pages within the virtual machine using appropriate native interfaces and when server de allocate previous allocated memory the balloon starts deflate by instructing it to de allocate previously allocated pages, but it is not helpful to deal with the memory over commitment and not efficient in such a manner to absorb the huge amount of memory pressure and thus memory over commitment occurs.

Xen in (Magenheimer, 2008) provides a functioning balloon driver and some limited tools for interacting with memory management and its challenges. A balloon driver when managed, complements well with the linux kernel page replacement mechanism. When the balloon is inflated by one page the kernel surrenders the page that it believes is least likely to be used again. If the balloon can be inflated until just before the guest hurts for memory, the guest's idle memory will be minimized and Xen can use that otherwise idle memory for another guest but still no helpful management is done with this mechanism. If memory pressure is increased and balloon is inflated more than it deflates itself that is self-ballooning that is described by the Xen in (Magenheimer, 2008).

Interposition in (Oren Laadan, 2010) is used as a key mechanism that can provide the redirection that is needed for virtualization of namespaces. Interposition captures events or processes at the interface between applications and operating system and performs some useful processing on those events or processes before passing them down to the operating system or up to the applications. The interposition that is used for implementing operating system is preprocessed so that the native kernel functionality is executed. Similarly, some post processing is also done after the native kernel functionality execution.

Proposed Model

Ballooning mechanism uses the balloon module that inflates and deflates according to the current condition of the memory pressure but it was not handling the memory over commitment. This mechanism is extended and a module of Intermediate is added to this mechanism. This Intermediate module is placed just after the balloon module that is directly linked with the condition of the balloon.

When the balloon module is in its normal condition, the balloon module request the intermediate module to claim more workload or task and balloon utilizes its memory. If the balloon is fully utilized and memory pressure is high it also tells the intermediate module and then this intermediate module starts checking the objects by using interposition technique and those objects that are under process in the balloon, those objects that are similar and dependent on them are not interrupted or blocked and all those objects that are distinct are blocked for a while so that the memory pressure will decrease.

When the memory pressure is decreased, the intermediate module unblocks the objects and the resume to be executed normally as they are before blocking.

The below model in (figure 1) is the proposed model with the addition of a new module named as the Intermediate that will help the balloon to inflate and deflate by using the interposition technique as its decision making mechanism.

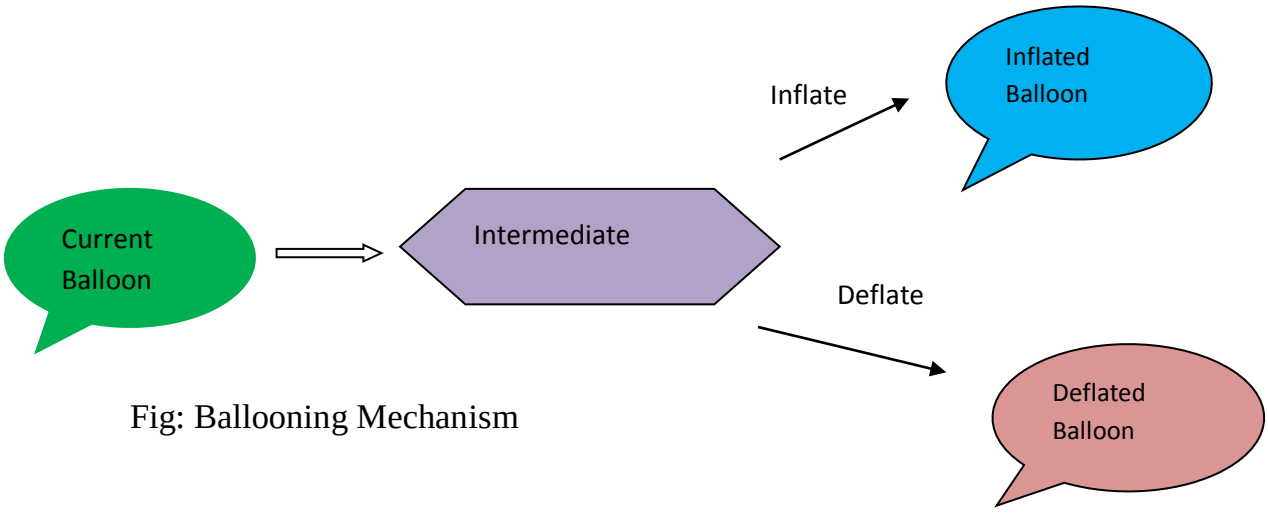

Fig: Ballooning Mechanism

Algorithm

Function is_full(baloon:full)

If baloon != baloon+1

Return FULL

Else if baloon → baloon +1

Return NOT FULL

Function intermediate (baloon:fully)

If baloon → FULL

Return Deflate

Else if baloon != FULL

Return Inflate

Function de_in(baloon:deflate_inflate)

If baloon → FULL

For(baloon →FULL;baloon--)

{ obj1 != obj2

obj3 != obj2

obj1 == obj3

.

Return obj2 → Block

.

.

}

Return obj2

.

.

Else if baloon != FULL

For(baloon != FULL; baloon++)

{ baloon ← obj1 + obj2 + obj3 ……..

Return obj1, obj2, obj3

.

.

}

Return Unblock

Conclusion

We have presented the mechanism used to manage memory resources to overcome the memory over committed problem. Our contribution includes novel technique and algorithm for allocating memory across virtual

machines to deal with memory over commitment problem.

Our extended ballooning technique uses the intermediate module to manage the memory via balloon module by using interposition as a decision making technique to allocate and de allocate the memory and avoid memory over commitment and we believe that this is very simple solution delivers the help for the vast value of memory over commitment.

References

1. Magenheimer, D. (2008). Memory Overcommit… without the commitment. Extended Abstract for Xen Summit.

2. Micah Dowty, J. S. (2008). GPU Virtualization on VMware's Hosted I/O Architecture. USENIX Workshop.

3. Oren Laadan, J. N. (2010). Operating System Virtualization: Practice and Experience. SYSTOR, Haifa, Israel.

4. Potter, S. (2010). Virtualization Mechanisms for Mobility, Security. COLUMBIA UNIVERSITY.

5. Waldspurger, C. A. (2010). Memory Resource Management in VMware Server. palo Alto, CA 94304 USA .