AUTONOMIC COMPUTING IN VIRTUALIZED ENVIRONMENT

Anala M R[1] and Dr.Shobha G[2]

[1]Department of CSE, RVCE, Bangalore

anala_m_r@yahoo.co.in

[2]Dean PG Studies (CSE & ISE), RVCE, Bangalore

shobhatilak@rediffmail.com

ABSTRACT

The infrastructure of IT-sector is heterogeneous and complex. It is difficult for a system administrator to manage these complex systems. This led to complex, unmanageable and insecure systems. A new approach to solve this problem is the IT infrastructure automation. Autonomic computing is the solution to manage complex infrastructure which provides "must have" solutions for the problems above. The increased resource utilization in IT made the enterprises to use the costly physical servers. These physical servers were under-utilized. Virtualization is a technology with efficient and better utilization of resources. Virtualization and autonomic computing are of major importance for the IT-sector and could be successfully combined to reach the ideals of the autonomic computing. This paper presents a framework for automating live migration of virtual machines in virtualized environment using autonomic computing. This paper also addresses the types of attacks encountered during live migration of virtual machines.

KEYWORDS

Virtualization, autonomic computing, Self-management and live migration.

1. INTRODUCTION

1.1 Autonomic computing

The heterogeneous infrastructures such as hardware and software applications, services and networks, directed towards the complex, unmanageable and insecure systems. The system administrator will not be able to manage alone such a complex computing system. Therefore these complex computing systems are managed by autonomic computing. Autonomic computing is the technique of managing the computing environment by computing systems themselves based on the policies decided by the system administrator. Self-management is the core of autonomic computing [1] and represents the process by which computing systems manage themselves. There are four aspects of self-management:

• **Self-configuration**: The property in which the computing systems will automatically configure itself by sparing the system administrator with respect to high-level policies. Any new components added to the system are incorporated seamlessly. It can dynamically adapt to changing environments. Such changes could include the deployment of new components or the removal of existing ones, or dramatic changes in the system characteristics

• **Self-optimization**: The computing system will continuously try to improve its performance by dynamically changing its parameter at run-time. It can monitor, allocate and tune resources

DOI : 10.5121/ijitcs.2012.2108

automatically. The tuning of resources means reallocation of resources in response to dynamically changing workloads in order to improve overall utilization. Without self-optimizing functions it is difficult to effectively use the computing resources when an application does not completely use its allocated resources.

• **Self-healing**: the property of a computing system to detect, diagnose and repair local problems caused either by software of by hardware failures. It can discover, diagnose and react to disruptions. Self-healing components can detect system malfunctions and initiate policy-based corrective action without disrupting the IT environment.

• **Self-protecting**: the ability of a computing system to secure itself from a malicious attack. The self-protected computing system is capable of predicting possible problems based on logs and reports. It can anticipate, detect, identify and protect against threats from anywhere.

1.2Virtualization

Virtualization [2] is a technique of separating the resources of computer into multiple execution environments. Virtualization techniques create multiple partitions which are isolated with each other called virtual machines. A virtual machine is a software implementation of a physical machine where an isolated operating system is installed within the operating system of a physical machine. The desire to run multiple operating systems was the original motivation for virtual machines. Virtualization is a technology with efficient and better utilization of resources.

Techniques of virtualization:

There are three different approaches to achieve server virtualization: the virtual machine model, the paravirtual machine model, and virtualization at the operating system (OS) layer.

Architecture of Virtual machine model:

The first technique, virtual machine model uses the host/guest paradigm as shown in figure 1. Each guest runs on a virtual imitation of the hardware layer. This approach does not require any alteration to the guest operating system to run. The administrator can create guests that use different operating systems. The guest has no knowledge of the host's operating system because it is not aware that it's not running on real hardware. The guests require real computing resources from the host therefore it uses a hypervisor [3] to coordinate instructions to the CPU.

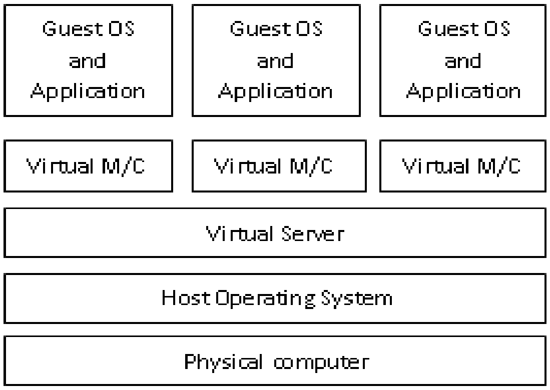

Figure 1: Architecture of virtual machine model

The hypervisor is called a virtual machine monitor (VMM) [4]. It validates all the guest-issued CPU instructions and manages any executed code that requires addition privileges. VMware, Virtualbox and Microsoft Virtual Server both use the virtual machine model. In the bottom of the stack the Host OS manages the physical computer. The VMM manages all VMs running on this physical computer. The Guest OS and the application run on these virtual machines.

Architecture of para-virtual machine model:

As shown in figure 2 the paravirtual machine (PVM) model is also based on the host/guest paradigm. This technique also uses a virtual machine monitor. Unlike virtual machine model the para-virtual machine model the VMM actually modifies the guest operating system's code.

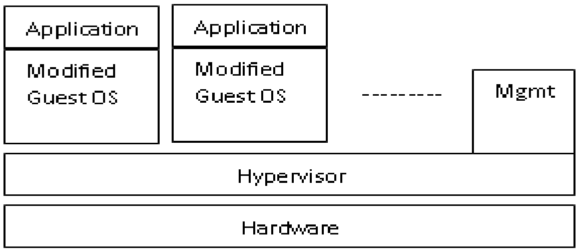

Figure 2: Architecture of para-virtual machines

As guest OS is to be modified, this is possible for only open source OS such as Linux and BSD and wnkdows cannot. The Intel(Intel VT) and AMD (AMD-V) provide functionality that enables unmodified OS to be hosted by a para-virtualized hypervisor. Like virtual machines, para-virtual machines are capable of running multiple operating systems. Xen[2,5] and UML both use the para-virtual machine model.

Architecture of Virtualization at the OS level:

This technique is different from virtual machine models and paravirtual machines. It isn't based on the host/guest paradigm. In the OS level model, the host runs a single OS kernel as its core and exports operating system functionality to each of the guests. Guests must use the same operating system as the host. Each virtual environment has its own file system, process table, networking configuration and system libraries as shown in figure 3.

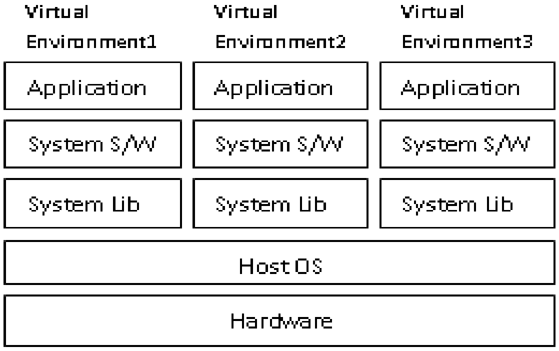

Figure 3: Virtualization at OS level

OS virtualization provides software emulation. This distributed architecture [6] eliminates system calls between layers, which reduces CPU usage overhead. Virtuozzo and Solaris Zones both use OS-level virtualization. The major limitation of this technique is the selection of OS since each container must be of same type, version and patch level.

Comparison:

The three techniques available for virtualization differ in implementation complexity, the extent of OS support, performance in comparison with standalone server, the level of access to common resources. The virtual machine models has wider scope of usage, but exhibits poor performance. Para-VMs have better performance, but they support fewer OSs because they modify OS. Therefore para-virtual machine models preferably use open source OS like linux.

Migrating operating system instances across distinct physical hosts is a useful tool for administrators of data centers and clusters. It allows a clean separation between hardware and software, and facilitates fault management, load balancing, and low-level system maintenance. By carrying out the majority of migration while OSes continue to run, we achieve impressive performance with minimal service downtimes. Migration can be conducted offline (where the guest is suspended and then moved) or live (where a guest is moved without suspending).

An offline migration suspends the guest then moves an image of the guest's memory to the destination host. The guest is resumed on the destination host and the memory the guest used on the source host is freed. The time an offline migration depends on the network bandwidth and latency. A guest with 2GB of memory should take several seconds on a 1G bit Ethernet link. A live migration keeps the guest running on the source host and begins moving the memory without stopping the guest. Migration is useful for

- **Load balancing** – Guests running in source host can be moved to some other host when guest doesn't get sufficient resources at source host.
- **Hardware failover** – When source machine's hardware fails, for the continued services guests can be safely moved to new host.
- **Energy saving** – During lower usage of a host system in order to save power guests can be reallocated to other hosts and host systems are powered off.
- **Geographic migration** – To minimize the latency guests can be moved to another location in serious circumstances.

During live migration of the virtual machine instance to another virtual machine introduces additional security vulnerabilities [7].

2. VIRTUALIZATION AND AUTONOMIC COMPUTING- A LITERATURE

The heterogeneous infrastructure used in IT-sector is difficult to be managed by a system administrator. Similarly in a virtualized environment the difficult task is management of running virtual machines, balancing of load in virtual machines and availability of virtual machines. It is a complex task to manage all the virtual machines by a system administrator. Therefore it is necessary to design a fully autonomic virtual machine-based environment. In this the system needs to be able to manage itself, with no or little interference from the system administrator. Unfortunately, virtual machine-based environments nowadays are manually managed by system administrators using GUI-based management tools that do not provide any automation capabilities. Yet virtualization itself can be used to split a complex system in multiple homogenous virtual machines which can be easily managed and physically moved around the system, thus simplifying the autonomic task.

The lack of such solutions shows that the self-management of virtual machine-based environments is a field of research where there is still a lot of work is to be done. This subsection presents a survey of preliminary academic work related to the self-management of virtual-machine based environment [8]. There are different approaches used to manage resources in virtualized environment- policy based, Control theory based and it can be based on utility functions. This section details the study of these techniques.

1.1Policy-based approach:

This approach is a popular way of managing resources by a computing system in a virtualized environment. Using this approach in [9], they developed a system called VIOLIN. The VIOLIN is a virtual computational environment deployed on nanoHUB infrastructure, designed and developed using Xen virtual machines. VIOLIN is capable of connecting multiple virtual machines by a virtual network. The autonomic adaptation of virtual environment in VIOLIN is driven by two factors, the availability of resources and the resource needs of each application running inside the VIOLIN. The VIOLIN system supports live migration of virtual machines on unavailability of resources in source physical machine.

Another policy-based approach has been used by Grit et al. [10]. Here, the authors used Shirako, a Java-based toolkit for resource leasing services for what they refer to as an autonomic orchestration. Shirako system leases networked resources on-demand basis. The Shirako system uses Java toolkit for leasing the resources on the basis of SHARP framework. Shirako focuses on orchestrating hosting of Xen VMs as a basis for secure adaptive, on-demand resource sharing in federated cluster. The mapping of application services onto a shared server network is known as orchestration. The resource leasing of Shirako is dynamic and renewable. The leasing and the contract mechanisms are common for all the resources but the space of contract attributes and values is resource specific. The specific attribute sets selected for each contract is a policy choice.

1.1 Control theory

This is another approach used for the dynamic allocation of resources in virtual environments. In [11], To manage infrastructure pool used by multiple application, it is required to have a control system that can dynamically allocate resources to applications in real time. Initially the static allocation technique was used. Due to the varying workload of running virtual machine it is necessary to use the dynamic resource allocation techniques. This paper addresses the overhead of dynamic allocation over static allocation.

In paper [12] a new concept called friendly virtual machine was introduced where VM themselves are adaptable. In this paper [13] they present a grey prediction control model used for dynamic allocation of resources in virtual machines. The dynamic adjustment of resources to an application depends on the local resource demands of a node that hosts the application. There are many techniques to allocate resources dynamically but this paper is based on control theory. Predictive control technique uses the historical data to allocate resources. The grey prediction control is accurate under the environment when the usage of resource in each VM is random.

1.1 Utility functions

The paper [14] demonstrates the effectiveness of utility functions for web based transactional workloads running on Linux. An autonomic computing system optimizes the high-level guidance from humans. This high-level guidance is converted to low-level actions to achieve the objective of desired optimizations using utility functions. This utility function is designed by the administrators. The utility functions allow the on-the fly identification of best feasible state.

In paper [15] the autonomic computing was used to allocate CPU resources dynamically for VMs by optimizing utility functions. The first utility function used to allocate resource is based on CPU share and priority based models explained in[15]. This paper addressed the technique used to dynamically allocate resources to various virtual machines with respect to varying workloads. It is based on the dynamic CPU priority allocation by allocating CPU shares to virtual machines. The self – organised and self-tuned technique is designed which is based on the combined use of combinatorial search techniques and analytical queuing network models. The global utility function is optimized under varying workload levels. The results are obtained through simulation.

3. PROPOSED SYSTEM DESIGN

The proposed work is to design and implement an autonomic system to manage virtual environment automatically without the involvement of system administrator. The designed system automatically identifies the failure of running virtual machine. The failure may be because of insufficient resource allocation for running virtual machine and hardware failure. An appropriate destination physical machine with sufficient resource is identified. The source physical machine indicates the failure information to destination machine. The destination physical machine after identifying the failure will configure a new virtual machine and installs the same. The running source virtual machine will start migrating to destination physical machine. Once the complete memory image of source VM is copied to destination, the source VM is stopped and the VM at destination takes over.

There are different phases in designing an autonomic system in a virtualized environment as shown in figure 4. The source host runs two Self-management aspects the Self-Optimization and the Self-healing. Similarly the destination host has two different Self-management aspects the Self-Configuration and Self-Protection.

Initially the source host VMs will be efficiently executing the applications by Self-optimizing the resources. The resources are optimized by frequently monitoring the available resources, status of the hardware and power consumption. The self-optimizing system will tune the resources allocated to all the virtual machines. The optimizer is also responsible to allocate resources dynamically for a newly created virtual machine.

Due to unavailability of resources or higher power consumption or hardware failure, VM in source host may fail to continue. This point of failure is identified by the Self- healing system and it will initiate the migration by indicating the failure information to destination host. The failover is decided based on set of parameters like performance of each application running on VM, resource conflicts and system performance etc. the resource requirement of each VM, VM size etc.

The destination host after receiving failure information will self-configure a new VM but with the same IP address as the source VM and installs. The memory image of the source host VM is copied to the destination host VM which is installed and this copy is initiated by Self-healing system. The process of moving a virtualized guest from one host to another is called as live migration. During live migration of the virtual machine instance to another virtual machine introduces additional security vulnerabilities. This is because the memory is copied in the form of plain text to the destination VM. The Self-Protection system is responsible to protect live migration from the above mentioned security attacks. Once the live migration is done successfully the VM installed at the destination resumes. The VM running at the source host is stopped completely. The types of attack on live migration are classified as

- ➢ *Attack due to lack of proper access control:* may allow an attacker to arbitrarily initiate virtual machine migrations. There are different security issues to be considered during live virtual machine migration
    - ▪ *False Migration:* An attacker may live migrate guest virtual machine to attacker's machine and gain access to guest virtual machine OS.
    - ▪ *Denial Of Service:* An attacker may migrate a large number of guest virtual machines to a victim virtual machine. This leads to overloading and causing disruptions in victim virtual machine.
    - ▪ *False resource sharing:* An attacker may falsely advertise available resources.
- ➢ *Attack due to insecure and unprotected transmission channel:* If the transmission channels are not protected the migrated data may be corrupted or accessed by an unauthorized person.
    - ▪ *Man-in the-middle attack:* it allows an intruder to gain access to transmission channel. An intruder may collect sensitive information and leak. An intruder can manipulate the memory copy of migrating VM intern compromises a guest VM.

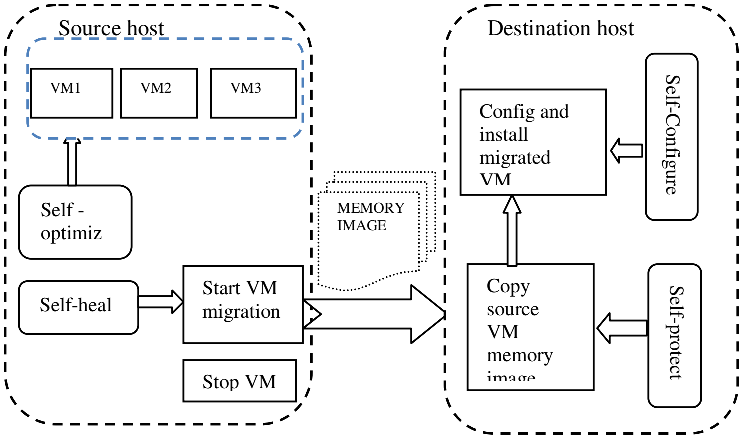

Fig 4: Autonomic computing in virtualized environment

3. CONCLUSIONS

The system administrator finds it difficult to manage the entire complex system alone since the system requirements will be changing dynamically. Similarly in a virtualized environment it is difficult to manage the resource requirement of each virtual machine by an administrator. Therefore the resources for these virtual machines are needed to be managed automatically without the intervention of system administrator. This is achieved by applying the techniques of autonomic computing in virtualized environment. This paper proposes a design of an autonomic virtualized environment. Here the resource allocations for all the virtual machines are managed dynamically and automatically. The proposed technique will reduce the load of system administrator and it fastens the entire process. Since vulnerabilities are introduced during live migration, they are overcome by designing an efficient self - protecting system.

REFERENCES

[1] Kephart, J. O. ; Chess, D. M.: The Vision of Autonomic Computing. In: Computer 36 (2003), S. 41–50

[2] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho,R. Neugebauer, I. Pratt, and A. Warfield, “Xen and the art of virtualization,” Symposium of Operating Systems Principles, 2003.

[3] Mitchem T Lu, R.O'Brien, R. ”Using kernel hypervisors to secure applications” Secure Comput. Corp., Roseville, MN; This paper appears in: Computer Security Applications Conference, 1997. Proceedings., 13thAnnual Publication Date: 8-12 Dec 1997 On page(s): 175-181

[4] "High Performance VMM-Bypass I/O in Virtual Machines" Jiuxing Liu, Wei Huang, Bulent Abali, and Dhabaleswar K. Panda USENIX Annual Technical Conference (2006)

[5] “XenSource.” [Online]. Available: http://www.xensource.com/.

[6] Crago,S.P. Schott,B. Parker,R. “SLAAC: a distributed architecture for adaptive computing” Inf. Sci. Inst., Univ. of Southern California, Arlington, VA;This paper appears in: FPGAs for Custom Computing Machines, 1998. Proceedings. IEEE Symposium on Publication Date:15-17 Apr 1998 On page(s): 286-287

[7] Jon Oberheide, Evan Cooke, Farnam jahanian. ” Empirical Exploitation of Live Virtual Machine Migration”, University of Michigan.

[8] Dan Marinescu, Reinhold Kr¨oger: “State of the art in autonomic computing and virtualization”, September 2007

[9] Ruth, P. ; Rhee, Junghwan ; Xu, Dongyan ; Kennell, R. ; Goasguen, S.: Autonomic Live Adaptation of Virtual Computational Environments in a Multi-Domain Infrastructure. In: Autonomic Computing, 2006. ICAC ’06. IEEE International Conference on, 5–14

[10] Grit, Laura ; Irwin, David ; Yumerefendi, Aydan ; Chase, Jeff: Virtual Machine Hosting for Networked Clusters: Building the Foundations for Autonomic Orchestration

[11] Wang, Zhikui ; Zhu, Xiaoyun ; Padala, Pradeep ; Singhal, Sharad: Capacity and Performance Overhead in Dynamic Resource Allocation to Virtual Containers. (2007)

[12] Zhang, Yuting ; Bestavros, Azer ; Guirguis, Mina ; Matta, Ibrahim ;West, Richard: Friendly virtual machines: leveraging a feedback-control model for application adaptation. In: VEE ’05: Proceedings of the 1st ACM/USENIX international conference on Virtual execution environments. ACM Press. – ISBN 1595930477.

[13] Xianghua Xu, Yanna Yan, Jian Wan : Grey Prediction Control of Adaptive Resources Allocation in Virtualized Computing System , 2009 Eighth IEEE International Conference on Dependable, Autonomic and Secure Computing.

[14] Walsh, W. E. ; Tesauro, G. ; Kephart, J. O. ; Das, R.: Utility functions in autonomic systems. In: Autonomic Computing, 2004. Proceedings. International Conference on, 70–77

[15] Menasce, Daniel A. ; Bennani, Mohamed N.: Autonomic Virtualized Environments. In: ICAS ’06: Proceedings of the International Conference on Autonomic and Autonomous Systems. Washington, DC, USA : IEEE Computer Society, 2006. – ISBN 0–7695–2653–5, S. 28.

Anala M R., Asst.Professor in the department of Computer Science and Engineering, RVCE. She has completed her M.Tech from Visveswaraya technological University in the year 2007 in the field of Computer Network and Engineering. She is pursuing her Ph.D in Visveswaraya technological University .She has 6 Years of academic experience in R.V.C.E. She is a member of CSI and ISTE. Her area of research includes Networking , virtualization and computer architecture.

Dr. Shobha G., Professor in the Department of Computer Science & Engg, RVCE. She has been awarded Ph.D for her thesis titled “Knowledge Discovery in Transactional Database Systems” from Mangalore University, Mangalore. She obtained her M.S. degree in Software Systems. From BITS, Pilani and BE in Computer Science from Gulbarga University. Her research interests are Data Mining, DBMS, Operating Systems & Networking. She has guided more than 35 undergraduate and 15 post graduate projects. Currently she is teaching courses on DBMS, Data Mining, Networks & Operating System. She has presented and published papers at national and International journals / conference.