Distributed Virtualization Manager for KVM Based Cluster

Mr. Uchit Gandhi*, Mr. Mitul Modi, Ms. Mitali Raval, Mr. Paavan Maniard, Dr. Narendra Patel, Prof Kirti Sharma

Birla Vishvakarma Mahavidyalaya, Gujarat Technological University, India

Abstract

Recent times have witnessed a growing use of Virtualization and Cluster environment in order to optimally handle and use resources. KVM is a hypervisor, which is used to achieve full virtualization. Many of the cluster environments make use of KVM in order to achieve a cluster of virtual machines. In order to manage this type of cluster and the virtual machines therein, managing systems have been developed. Major problem with existing management systems is that they are developed for native systems, which poses serious problems when need for changing underlying environment arises. Also, these systems need high capacity dedicated systems that serve only as manager.

The aim of this system is to address above issues by developing a distributed, web-based manager for KVM based cluster. As this system is a web-based one, it becomes independent of underlying environment. Also, this results in lesser resource consumption. As the system is a distributed system, fault tolerance can be achieved.

Keywords: Virtualization; KVM; Virtual Machine Manager; Distributed System

1. Introduction

Hardware Virtualization may be defined as partitioning the computer’s memory into separate and isolated "virtual machines" that simulate multiple machines within one physical computer. Hardware Virtualization enables multiple copies of the same or different operating systems to run in the computer and also prevents applications from interfering with each other. For example, a computer running Microsoft Windows 7 can make a virtual machine that can run Fedora 21 or Ubuntu 15.04. KVM is a bare-metal hypervisor that enables full virtualization by making use of virtualization instructions of the microprocessor.
The system intends to meet following objectives. These objectives can be divided in two parts:

- **Part 1:** Providing a web-based interface to create, pause, shut down and destroy virtual machines on any of the machines (node) connected in the cluster from any other machine. Providing resource usage information of virtual machines running on any node. Enabling Live Migration of virtual machines from any node to any other node

- **Part 2**
  - Load Balancing Policy - Carry out load balancing across nodes based on information regarding resource consumption by virtual machines.
  - Fail Over Policy - In case of failure of any of the physical machines, migrating virtual machines to another node and resuming them. Keeping physical machines synchronized with information regarding virtual machines existing across all the nodes.

2. **Components of the system**

In this section, the structure of the system and components involved are discussed:

2.1. **Interacting with hypervisor - libvirt**

For communication with hypervisor KVM, use of libvirt is done in this system. Libvirt API provides support for hypervisors like KVM, Xen, VirtualBox, VMWare Workstation etc. It also has a support for LVM and iSCSI filesystems, which are used in the system, as described in section Storage.

2.2 **Storage**

System makes use of shared storage in order to store the disks of virtual machines. As the disks are centrally stored, live migration can be carried out without having to actually migrate the disks across the network, resulting in controlled network traffic, which ultimately results in less overhead on network infrastructure. The use of iSCSI file system is made to store the volumes on the shared storage. As iSCSI enables block-level access to remote hard disks, the physical location of storage on shared storage is abstracted. Also in case of failure of physical machine, there is no loss of virtual machine data as the virtual machine disk is located on shared storage. Use of LVM volumes is made, as LVM facilitates dynamic volume resizing and provides mechanism of Snapshots.

2.3. **Web-based Manager**

Use of JSP and Servlet technologies is made in order to develop the web-based system. Also Libvirt API provides bindings for Java, facilitating development of the system. MVC architecture is loosely followed with JSP enabling the view and Servlets acting as Controllers.
2.4. Resource Information

Collection of information regarding resource utilization is achieved with the help of collectd. Collectd is a system, which collects information about resource usage automatically at predefined time interval. Collectd provides plug-in for virtualization, which may be used to obtain information from KVM hypervisor. Collectd periodically generates Round Robin Database (RRD) files, which carry resource information at different intervals of time. This information can subsequently be rendered in the form of graphs on web-pages using JavaScript and can then be served via the JSP-Servlet architecture.

2.5. Viewing graphical output of virtual machine in web browser

To enable viewing the standard graphical output of virtual machine in web browser, use of noVNC package was made. noVNC package emulates the popular Virtual Network Computing technology in web browser.

2.6. Maintain Synchronization with rsync and XML Files

The information about all the virtual machines running on all the physical nodes in the cluster needs to be there with all the machines in order to achieve a distributed system. For this, use of files is made. One of the files carries information regarding the IP addresses and user names of different systems, which is needed in order to remotely communicate via SSH. Another file carries information like IP address of physical machine and its role along with number and names of virtual machines running on the node. This information is copied on all the nodes using rsync utility, which is an open source utility. Rsync provides ways of copying files to local and remote machines and also supports differential backups, resulting in reduced network traffic.

The XML description of all the virtual machines running on all the nodes is also maintained at all the physical machines along with Shared Storage. This XML description of virtual machines is used along with the former XML file carrying list of virtual machines, to resume the virtual machines in case a node fails, on any other node.

2.7. Maintaining Communication among the nodes

The machines exchange control information like Startup and Shutdown broadcasts by messages, achieved using Sockets. Libvirt enables interaction with remote KVM hypervisor with the help of SSH. Hence it is essential to set up SSH and mark all the machines as authorized prior to usage of system. Rsync which is used to transport configuration files across the network also makes use of SSH.

2.8. Coordinating the system and achieving fault tolerance

In order to co-ordinate the system, all of the active nodes under consideration are dynamically assigned one of the three roles:

- coordinator
- backup-coordinator
- member

As a node starts up, it broadcasts a message about its startup to all the nodes of cluster. If a node has already been marked as coordinator, it informs the newly active node about its own status as coordinator and hence newly active node becomes member node. If the newly active node does not receive any message about coordinator node, it makes itself coordinator. Subsequently, if any other nodes become active, it informs them about its status as coordinator. Node designated as coordinator carries out following tasks:

- It periodically pings the machines to check whether all nodes are up-and-running or not.
- If any of the node is detected as down, it invokes the Failover Manager to handle the virtual machines on the failed node and resume their operation on one of the remaining active nodes.
- It also periodically queries the hypervisor on remote machines for resource utilization information of the virtual machines, obtained using collectd. Using this information, it calculates load on the machines and invokes Load Balancer if the load is not in the predefined limits.
- Periodically transfers configuration files to all the nodes in the cluster.

Backup coordinators are the nodes that work to make the system fault tolerant. If any of the member nodes are down, coordinator node becomes aware about it. But if the coordinator node goes down, the system loses its
coherence. To avoid this situation, backup coordinators are introduced. If there are $n$ nodes in a cluster, $((\log n)-1)$ nodes may be designated as backup nodes. These nodes, classified as backup nodes, ping each other and also the coordinator. Also, the configuration files, which are present on coordinator, are also available with backup coordinators. If any of the backup coordinator goes down, coordinator designates another node as backup coordinator. In the scenario when coordinator goes down, one of the backup coordinators becomes coordinator and a member node is assigned backup coordinator status.

2.9. Policies used in the system

- **Fail Over:**
  
  According to this policy, we check if all nodes are working properly. If any node goes down, its Virtual Machines running on that node are migrated to another node. Heartbeat mechanism can be used to check status of all nodes at predefined intervals. *Please note that VMM stands for Virtual Machine Manager – the system we are designing.*

![Fail Over Policy](image)

- **Load Balancing:**
  
  According to this policy, resource usage information by each virtual machine on each node is fetched. Using this information and predefined algorithm, load on each physical machine / node is calculated. If the load on any node increases beyond predefined limit, one or more virtual machine(s) are migrated from heavily loaded node to node with lower load.

  Also, in the scenario where any of the nodes are under-loaded or are having load less than predefined limit, the virtual machines on the under-loaded node may be migrated to another node and the under-loaded node can be shut down to conserve energy.

![Load Balancing Policy](image)

3. Result Analysis
System was deployed and tested with Apache Tomcat 8 and Mozilla Firefox on Centos 6.5. The UI of manager seen in screenshots is developed with the help of Bootstrap CSS framework.

1. Fig - 4 The Dashboard of the system – the first screen where all of the connected nodes of the system are listed. From here, virtual machines on any of the node can be controlled or new virtual machines can be created.

2. Fig - 5 Dashboard-2 - Listing of Virtual Machines within nodes and web based controls to control individual virtual machines.

3. Fig - 6 Dashboard-3 – Example: Delete Virtual Machines using web controls.
4. Fig – 7: New Virtual machines can be created from the distributed manager dashboard. Options about how operating system should be installed can be provided, as shown above.

5. Fig – 8: For creating virtual machines, location of source can be supplied.

6. Fig – 9: RAM and No of CPUs that may be used by Virtual machines can be supplied
Information about Resource Utilization by Virtual Machine

1. Disk I/O

![Disk I/O graph](image1)

The figure shows the Resource Information of a particular virtual machine. The resources seen are Disk Input Output and Network activity. Disk I/O graph on X axis has time and on Y axis has the number of Kilobytes read and written by the virtual machine at that instant of time.

2. Network I/O

![Network I/O graph](image2)

The figure shows the Resource Information of a particular virtual machine. The resources seen are Disk Input Output and Network activity. Disk I/O graph on X axis has time and on Y axis has the number of Kilobytes read and written by the virtual machine at that instant of time.

3. CPU Utilization

![CPU Utilization graph](image3)

Fig – 11: Result Analysis – 1- I/O consumption analysis

Fig – 12: Result Analysis – 2 – CPU utilization
The figure shows the Resource Information of a particular virtual machine. The resources seen are CPU Utilization and Disk Input Output. CPU Utilization graph on X axis has time and on Y axis has the number of nanoseconds for which CPU was used by the virtual machine at that instant of time. Disk I/O graph on X axis has time and on Y axis has the number of Kilobytes read and written by the virtual machine at that instant of time.

4. Conclusion

Thus, this system works as a web-based manager for cluster of KVM virtual machines. Also, this system consumes significantly lesser resources compared to existing alternatives and achieves fault-tolerance. We are planning to work in future to enable live migration for virtual machine with local storage. Also we are planning to add different user roles and access rights for better administration of environment.

Acknowledgements

We would like to sincerely thank Mr. Nilesh Vaghela and Mr. Sham Arsiwala of Electromech Corporation, Ahmedabad, Gujarat, India for providing the idea of this system and helping us from time to time during the development process. We would also like to thank Dr. Darshak Thakore, Head of Computer Engineering Department, Birla Vishvakarma Mahavidyalaya for his guidance and support. We also thank the faculties of Computer Engineering Department for all the help they have provided.

References

[12] noVNC: https://kanaka.github.io/noVNC