Management of High Quality Available Services Using Virtualization

Ergest Alite[1], Olimpjon Shurdi[2] and Igli Tafa[3]

**[1] Faculty of Information Technology, Polytechnic University of Tirana,
Tirane, 1001, Albania**

**[2] Faculty of Information Technology, Polytechnic University of Tirana,
Tirane, 1001, Albania**

**[3] Faculty of Information Technology, Polytechnic University of Tirana,
Tirane, 1001, Albania**

Abstract

This paper addresses the use of virtualization technology in the management of high availability services using open source tools. Services are set up on virtual machines and can be migrated between physical nodes of a "cluster" system automatically with the help of a corresponding program. The work of this thesis is based on an open source program called Heartbeat. We will then propose a solution and create the corresponding code that will allow the Heartbeat program to migrate virtual machines from one physical cluster node to another. This migration is required to be performed at the time the first node is lowered from us to perform foreseen maintenance. The solution we propose will be tested in an experimental system set up by us where we will measure the impact that migration has on system network performance. On the practical side we will use Xen virtualization, which we will install on a Linux system Centos version 7. On Linux we will configure Xen and on the latter will be set up two other Centos 7. On two Virtual Centos 7 servers mounted on Xen will install heartbeat to simulate failover of a simple web page to be built on Apache installed on two Centos 7 servers. With this thesis we aim to achieve the following objectives:

1. Provide a detailed overview of previous work on the use of virtualization in critical service clusters.
2. Propose and implement a solution that enables the Heartbeat program to migrate virtual machines from one physical cluster node to another.
3. Evaluate system performance when migrating virtual machines from one node to another.

Keywords: *Virtualization; High Availability (HA); Heartbeat migration; System Performance; Cluster Topology.*

1. Introduction

The question raised randomly this days among is related with visualization and more precisely what is virtualization? Virtualization refers to technologies designed to provide an abstract layer between a computer's hardware systems and the software that runs on them. Providing a logical rather than physical appearance. Virtualization allows you to run multiple action systems at the same time on a single physical machine. Exactly, virtualization is the simulation of a hardware platform, an operating system, a memory device or a simulation of network resources. Virtualization provides the ability to execute applications, operating systems, or service systems in a particular logical environment independent of the physical computer system. Virtualization is used in a variety of ways including: in-app virtualization, desktop, user, memory and hardware as per Fig.1.

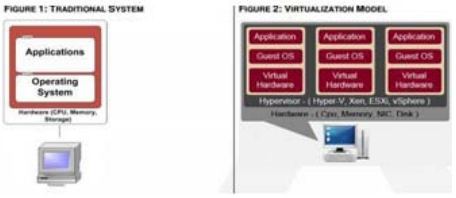


FIGURE 1: TRADITIONAL SYSTEM
Applications
Operating System
Hardware (CPU, Memory, Storage)
FIGURE 2: VIRTUALIZATION MODEL
Application
Guest OS
Virtual Hardware
Application
Guest OS
Virtual Hardware
Application
Guest OS
Virtual Hardware
Hypervisor - (Hyper-V, Xen, ESXi, vSphere)
Hardware - (Cpu, Memory, NIC, Disk)

Fig. 1 Virtualization Model.

1.1 Server Virtualization

This kind of virtualization is the one that the virtualization work is focused on today; and that virtualization that all companies are implementing. The key aspect of server virtualization or machine visualization is that different virtual machines do not share the same kernel and can therefore run different operating systems. This differs from system-level virtualization, where virtual servers share the same kernel and provide a number of unique infrastructures, clients and business opportunities. Server virtualization goes beyond the "one application, one server" principle and facilitates the consolidation of several servers into a single one. Server virtualization allows one server to do the work of several servers by sharing the resources of a single server in multiple environments. Virtualization software empowers the company to host several operating systems and many applications locally and remotely, breaking physical and geographical limits for users. Some methods for server virtualization are: Complete-virtualization; Para-virtualization; Virtualization at the operating system level.

2. HA (High Availability) Systems and Virtualization

A high availability system (HA system) is, by its name, a highly available system. Business critical services are often categorized as high availability services (HA services). Systems intended to provide critical services should be designed and implemented with the aim of minimizing the planned or unplanned downtime of services. There are many other examples of this all in common: the use of "hardware" in duplicate or multiply to avoid dependency on a single component. To ensure high availability we need two or more servers for the same service as per Fig.2.

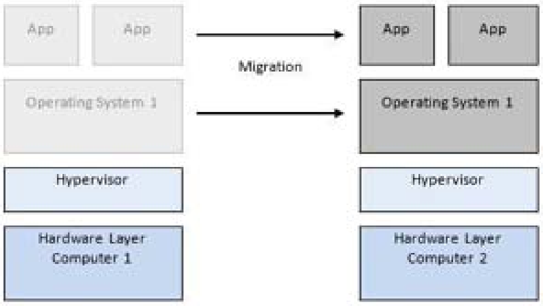


Fig. 2 Migration of a host from computer 1 to computer 2.

2.1 Clusters

High availability clusters are systems composed of several interconnected physical nodes that collaborate with one another through specialized software [1]. If one node falls then its services are transferred to the other node in order to continue offering. There are several ways to accomplish this scenario:

- *The LVS project ("Linux Virtual Server")* that provides a centralized topology where the "arbitrator" plays a central device or "gateway".
- *Heartbeat* another open source solution for creating HA cluster created by the High-Availability Linux Project. Heartbeat nodes do their "broadcast" periodically, just like a heartbeat, to show not only whether they are up or not but also to understand what services they are offering. In case one of the nodes fails, the other node cooperate to establish the services that the failed node was carrying.

Other open source projects include Ultra-monkey or Red Hat Cluster Manager.

2.2 Using Virtualization to Create an HA Cluster

Traditionally a server's "software" is dependent on the "hardware" where it is executed. This brings very little flexibility when it comes to moving a service or application from one hardware to another. Moving services from one server to another may be necessary in the case of a heterogeneous "cluster" where not all nodes have the same performance.

The case of virtualization makes it possible to completely migrate one operating system from one hardware to another due to the abstraction that the hypervisor brings. Migrating a virtual machine is relatively simple because the virtual machine and its applications are treated as a single object.

This kind of virtualization is the one that the virtualization work is focused on today; and that virtualization that all companies are implementing. The key aspect of server virtualization or machine visualization is that different virtual machines do not share the same kernel and can therefore run different operating systems. This differs from system-level virtualization, where virtual servers share the same kernel and provide a number of unique infrastructures, clients and business opportunities.

2.3 Availability Assessment

Availability is an important factor in HA systems and therefore the need for a numerical value for its formalization arises. We can then refer to this value to determine how available a system should be. Based on the history of the system we can calculate how available this system was. System availability depends on the rate of equipment failures.

2.4 Heartbeat and Xen

Heartbeat, as described in the previous section, is an open source solution for creating HA clusters.

Today's version of Heartbeat, is significantly improved than the previous versions. The most important change is the ability to support cluster n-nodes, out of only 2 nodes that version 1 could support. To increase communication security and reduce load on the existing network, it is recommended to use a dedicated Heartbeat network thing.

Xen, is a "software" to implement virtualization in x86 architectures [5] [3]. Theoretically Xen can support up to

100 virtual machines on a single physical server but practically this limit is much lower. Experimental evidence has shown that the maximum number of competing machines that Xen can support without losing performance is about 16. At Xen every virtual machine has a dedicated memory that is not shared with other machines [5].

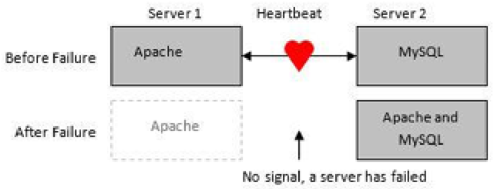


Fig. 3 Migration of services in case of an active-active configuration with two servers. Initially both servers are serviced while after the first one is serviced only the second but with both services.

In our project we want to execute at least 2 virtual machines running on the physical computer, which can be achieved without significant performance decline from Xen.

Virtual machine migration means moving virtual machines from one computer to another to maintain availability. The main objective is to make virtual machine migration unnoticed by service users by setting up the virtual machine on the new node while copying memory. The two main conditions to be able to perform a live migration are:

- Disk images ("disk images") that use virtual machines have to be accessible to all physical nodes, computers (cluster nodes) [3].

Physical computers must be homogeneous: their processors must have the same characteristics.

Finally Xen offers us an excellent virtualization platform. It is durable, easy to use, has high performance and some virtual machines can run on the same physical computer at the same time. The advantages of Xen are best used in a "cluster" managed by Heartbeat software. The sharing of services is one of the advantages of having a critical service for a virtual computer. If one service is corrupted, the other service remains intact.

The only privileged operating system is the system considered host operating system. Heartbeat has the ability to monitor and control resources but currently cannot migrate virtual machines live. It can only downgrade and upgrade virtual machines because it lacks the functionality to migrate them live.

2.5 Implementing the Cluster

The basic element of the system is the layer created by physical computers for executing virtual machines, which enables live migration of virtual machines from one node to another. Nodes collaborate to keep virtual machines set up. A prerequisite for live migration of virtual machines is that the nodes can be able to access the shared storage where the virtual machines have their images, which makes it possible to mount these virtual machines from any cluster-based node. it".

In order for each of the virtual machines to have access to the shared disk, this disk must be mounted on each of the physical nodes. Only a basic configuration is needed to install a new node.

Virtual machines are separate from physical computers and therefore when one of the computers fails the other computers can cooperate to keep all virtual machines running. From the Heartbeat perspective we have two categories of failures:

- *The Uncontrolled failures*. These include typical hardware failures such as power outages, damage to the power source, or any other hardware failure that causes the entire system to malfunction.
- *Controlled failures*. In this category scenarios are not exactly failures because they can be controlled. An example is the case where the system administrator lowers and raises the system after installing an application.

By automating the migration process we can provide a more affordable, scalable, cost effective, and easy to manage system than the classic systems used today.

3. Realizing "Live" Migration

We are familiar with the solution we propose for Heartbeat software, a solution that will allow it to perform live virtual machine migration from one physical node to another. We chose Heartbeat software because it is "open source" and highly tested in ensuring high availability in Linux environments.

3.1 Implementing the Cluster

Heartbeat bases its operation on certain processes ("daemons") running on "cluster" and communicating with each other at all times. Information on the operation of these processes is stored in the form of logs. If over 50% of the nodes are available then the nodes themselves choose one of them to be the "master cluster". Master cluster is otherwise called DC.

In our experimental system every node is equivalent so we have not defined any preferred nodes for setting up new virtual machines. If a client pays for additional hardware resources then the client will be given priority in using the hardware. Heartbeat can be configured to backup hardware for certain virtual machines but distribute this hardware when failures occur without interruptions.

3.2 Implementing the Cluster

In its simplest configuration, Heartbeat has access to control the node services using init.d scripts and the "start", "stop" and "status" arguments. As the names themselves indicate, these commands serve respectively to start, stop services, and check whether services are in a state of execution or not. For Xen, the "software" that will manage the virtualization, the start and stop commands are as follows:

```
xm create x0
xm shutdown x0
```

Where x0 is the virtual machine upon which these commands are implemented.

3.3 Heartbeat Internal Messages

Since the necessary documentation is missing it is difficult to predict Heartbeat behavior at the time of failure of a node.

```
xm migrate x0 node1 – live
```

The script using this command must know the destination where the virtual machines will migrate otherwise this command does not work. This command, if successfully executed, will migrate the virtual machine x0 live from the existing node to node1.

3.4 Integrity of “Shared Storage”

Virtual machine images are stored in "shared storage" enabling virtual machines to be mounted on any of the physical nodes. One danger here is the possibility of mounting the same virtual machine on more than one physical node.

Heartbeat offers two techniques to avoid this problem: "node fencing" and "STONITH”.

- "Node fencing" prevents access to "shared storage" from several nodes at once.
- "STONITH (Shot the Other Node in the Head)" looks brutal and such is it. When a node does not respond so it does not report its status, Heartbeat ensures that the “node is dead by "shooting it in the head”.

In our experiment we have not implemented any of these techniques because virtual machines do not have as critical data as they would in the case of a real production system.

3.5 Migration Intensity and Heartbeat Configuration

The behavior of the Heartbeat is defined in the cib.xml and ha.cf configuration files. The Heartbeat configuration has a significant impact on the behavior of the cluster because all critical services (virtual machines) are controlled by it.

When one node fails, the migration process begins to move its virtual machines to other nodes. As a result we have at least two migrations for each node that fails. After some failures and many migrations the "cluster" will reach a stable state, so we will have a balanced system.

3.6 Integrating Xen with Heartbeat

As mentioned, the uncontrolled failures are resolved by lowering and upgrading the affected virtual machines from scratch, which is easily accomplished in Heartbeat. So let's take as an example a scenario where the system consists of two nodes 0 and 1 and where node0 holds two virtual machines x0 and x1. If node0 fails uncontrollably then the virtual machines carrying x0 and x1 will be shut down and restarted at node1. Virtual machines are not available in the interval of time between the start of shutdown at node0 until the start of shutdown at node1.

Today's production environments are characterized by a high dynamics where nodes are maintained all the time, added to and removed from the "cluster" so a better solution is needed to provide a lower interruption of services at the time of migration of virtual machines.

4. Measurements and Results

A solution for Heartbeat was developed that enables automatic control of Xen instances including live migration. There are two main questions that arise and deserve special attention:

1- Are critical services available during migration?
2- To what extent are services provided to users affected by physical node closure for maintenance reasons?

The experiments will be based on TCP and UDP protocols, widely used today across the Internet as per Figure.4.

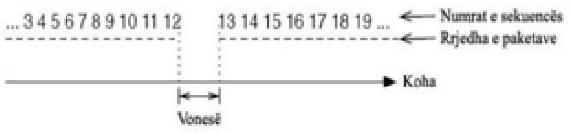


Fig. 4 TCP protocol behavior.

All applications for which performance matters more than reliability are best suited to use this mode of communication as per Figure.5

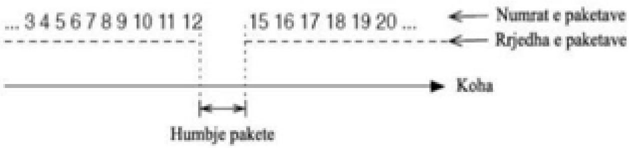


Fig. 5 Shows a UDP packet flow as an outage occurs. The main purpose of the measurements we perform is to estimate the cost of controlled failures in different scenarios.

- Measuring CPU Performance on Xen Virtual Machines. We appreciate Xen's ability to provide us with high-performance virtualization suited to today's servers.
- Measuring network performance during a failure. We will measure the impact on network performance of a controlled failure coupled with live migration of virtual machines.
- Impact of controlled failures on file transfers. Measure the time needed for copying a file when a migration occurs and find out how file transfer is affected by controlled failures. The worst case is verified when the transfer is interrupted or the file is corrupted.
- Impact of controlled failures on web services. How is a "web server" affected when a controlled failure occurs? Testing will be performed on both Apache2 and MySQL which illustrate the cost of migrating to a web server against the cost of migrating to a database server.

4.1 CPU performance on Xen Virtual Machines

We begin the series of measurements precisely by measuring the performance of the CPU on Xen virtual machines and comparing it with physical machines. To check if Xen is capable of delivering high performance in virtualization.

➢ *Methodology*

For the experiment we will use an Intel Core2Quad computer with 2GB of RAM. On this computer we have installed the Centos 7 Server operating system and through Xen we have created a virtual machine with openSuse operating system 11. When running the physical computer we can set up the operating system (Centos 7) without Xen support or Xen support. To perform the tests we will measure the total time the processor will need to calculate the value 123456123456. We will perform this measurement 10 times and calculate the average value.

1. *The steps to follow in the experiment are as follows*

- Run the system (Centos 7) without Xen support and measure CPU performance directly on the host operating system using the method described above. Save the result.
- Reboot the system but this time with Xen support (Centos 7-Xen) and measure the CPU performance on the host operating system again. Save the result.
- Without rebooting the system (Centos 7-Xen) we build the virtual machine and measure CPU performance over it. Save the result.

➢ *Results*

The test results are shown in Figure.6 where it appears that the virtual machine from the processing point of view is almost as fast as the physical computer on which it is running. The average difference is about 0.2 seconds. Interestingly, the Centos 7-Xen is 0.6 seconds faster on average than the Centos.

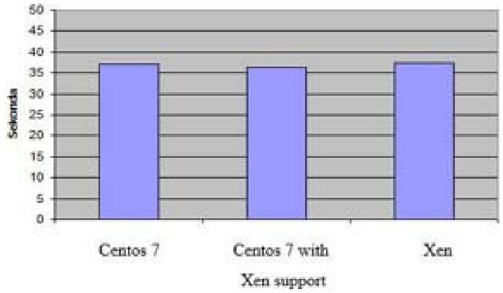


Fig. 6 Comparison of CPU performance between three different platforms: Centos 7, Centos 7 with Xen support and Xen virtual machine.

4.2 Network performance during a failure

The main purpose of Xen as it migrates live a virtual machine is to minimize disruption. This process can be separate into three phases [2]:

- The "PUSH" phase that copies the memory pages as the virtual machine continues to run.
- The “STOP-AND-COPY” phase where the virtual machine stops at the old node to continue executing at the new node. In stop and copy phase the virtual machine does not shut down but only becomes "pause - resume".
- Phase “PULL” to copy and those memory pages that may have been upgraded by the virtual machine between the first and second phases.

But it is precisely during these stages that we may experience a decrease in network performance due to the occupation of "bandwidth" by copying the memory contents. In the second phase, the virtual machine stops executing at the first node and resumes execution at the second node.

▪

In this short interval virtual machine is not available [2].

➢ *Methodology*

To measure network performance during a live migration we will use the topology of Figure.7. The devices used in this experiment are: "shared storage", two physical nodes, and a virtual machine on one of the nodes.

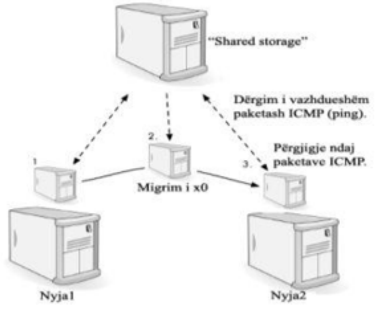

"Shared storage"
Dërgim i vazhdueshëm paketash ICMP (ping).
Përgjigje ndaj paketave ICMP.
1
2
3
Migrim i x0
Nyja1
Nyja2

Fig. 7 Live migration topology.

In order to measure network response time before, during and after migration, we will send ICMP packets to the virtual machine from the "shared storage" computer. As it appears from the command, the total number of ICMP packets will be 2000, sent over 20 seconds (100 packets per second). During the 20 seconds of the experiment we will simulate the controlled failure of the virtual machine x0, a failure that will lead to migration it's from node1 to node2.

➢ *Results*

The graph in Figure.8 shows how the ICMP packet turn-around time is affected by the virtual machine migration process.

If we analyze the graph further we can say that the time during which the virtual machine was unavailable (due to migration) was quite short otherwise we would have a significant break in the graph. The number of packages lost as a result of migration is about 20.

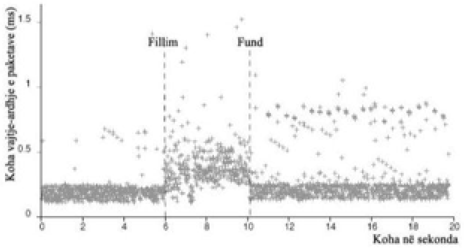


Fig. 8 Impact of a controlled failure on network performance.

4.3 Impact of failures on web services

1) Performance Testing of Apache2 and MySQL.

The experiment in this paragraph measures the response time and number of transactions performed while Apache2 and MySQL are executed on different virtual machines under the access of a certain number of users.

➢ *Methodology*

To test the performance of the "web server" we will be using an application called Siege whose purpose is to generate http traffic to this server, during which the "web server" will be under the constant traffic generated by Siege.

The web server and database run:

- On different physical nodes and there are no failures during the measurements. We expected that this scenario will have the highest performance compared to other scenarios.
- On the same physical node and there doesn't occur any failure during the measurements.
- On different physical nodes and during each measurement the failure of the physical node where the web server is located occurs. The web server is forced to migrate to another node. This scenario will help us compare the performance of a "web server" in migration with its performance when no failure occurs.

Node failures will be simulated by executing the already known /etc/init.d/heartbeat stop command on the physical node where the virtual machine is born at the time of failure.

➢ *Results*

In the Figure.9 is shown the average response time that is assumed to be lower in the two non-virtual machine migration scenarios. A lower response time means higher performance. This response time is also lower when the web server and that database are on different physical nodes compared to when they are on the same node.

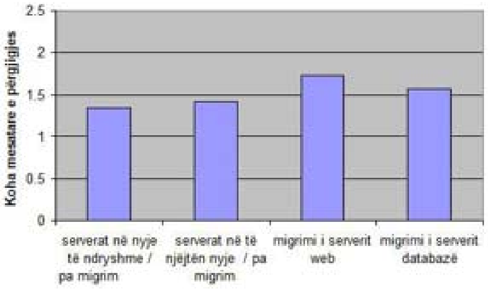


Fig. 9 Average response time in milliseconds.

5. Realization

To accomplish the practical part of this work, unable to have real equipment all was realized in a workstation in which is installed in the software "VMware Workstation" on which all the practical part was realized. We will use Xen virtualization, which we will install on a Linux system

Centos 7 version on which we will configure Xen, two more Centos 7 systems will be installed on Xen.
The two Virtos Centos 7 servers built on Xen will have a heartbeat installed to simulate the failover of a simple web page that will be mounted on Apache installed on the two Centos 7 servers. After installing VMware Workstation on the computer we will start with setting up a Centos 7 machine on this VMware Workstation (Figure.10).

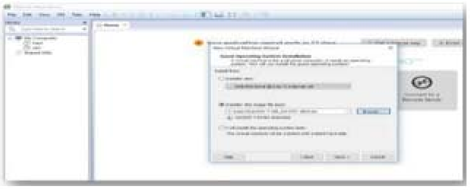
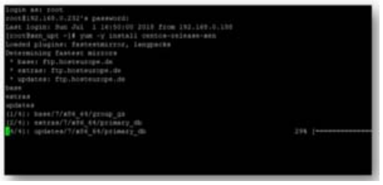


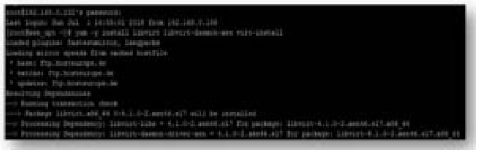
Fig. 10 Installation of Virtual Servers.

After successfully installing Cento we will access it with Putty to start installing Xen. Meanwhile I have also updated the Cento system to get the latest packages from the repository (Figure.11 and Figure.12).



```
login as: root
root@192.168.0.232's password:
Last login: Sun Jul  1 14:50:00 2018 from 192.168.0.100
[root@xen_upt ~]# yum -y install centos-release-xen
Loaded plugins: fastestmirror, langpacks
Determining fastest mirrors
 * base: ftp.hosteurope.de
 * extras: ftp.hosteurope.de
 * updates: ftp.hosteurope.de
base
extras
updates
(1/4): base/7/x86_64/group_gz
(2/4): extras/7/x86_64/primary_db
(3/4): updates/7/x86_64/primary_db                              29% [=============
```

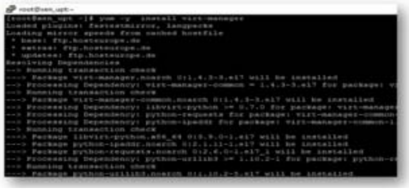
Fig. 11 Xen installation commands.



```
root@192.168.0.232's password:
Last login: Sun Jul  1 14:53:01 2018 from 192.168.0.100
[root@xen_upt ~]# yum -y install libvirt libvirt-daemon-xen virt-install
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
 * base: ftp.hosteurope.de
 * extras: ftp.hosteurope.de
 * updates: ftp.hosteurope.de
Resolving Dependencies
--> Running transaction check
---> Package libvirt.x86_64 0:4.1.0-2.xen46.el7 will be installed
--> Processing Dependency: libvirt-libs = 4.1.0-2.xen46.el7 for package: libvirt-4.1.0-2.xen46.el7.x86_64
--> Processing Dependency: libvirt-daemon-driver-xen = 4.1.0-2.xen46.el7 for package: libvirt-4.1.0-2.xen46.el7.x86_64
```

Fig. 12 Library installation result.

To work with xen there are two ways, with commands accessing through putty and installing the other virtual machines we will need for further simulation that we will do. The second way is by installing virt-manager on the server Centos in order to work with xen graphic. After installing Xen on the Centos server we start working with virt-manager to set up two more machines already on Xen (Figure.13 and Figure.14).



```
root@xen_upt:~
[root@xen_upt ~]# yum -y  install virt-manager
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
 * base: ftp.hosteurope.de
 * extras: ftp.hosteurope.de
 * updates: ftp.hosteurope.de
Resolving Dependencies
--> Running transaction check
---> Package virt-manager.noarch 0:1.4.3-3.el7 will be installed
--> Processing Dependency: virt-manager-common = 1.4.3-3.el7 for package: vi
--> Running transaction check
---> Package virt-manager-common.noarch 0:1.4.3-3.el7 will be installed
--> Processing Dependency: libvirt-python >= 0.7.0 for package: virt-manage
--> Processing Dependency: python-requests for package: virt-manager-common-
--> Processing Dependency: python-ipaddr for package: virt-manager-common-1.
--> Running transaction check
---> Package libvirt-python.x86_64 0:3.9.0-1.el7 will be installed
---> Package python-ipaddr.noarch 0:2.1.11-1.el7 will be installed
---> Package python-requests.noarch 0:2.6.0-1.el7_1 will be installed
--> Processing Dependency: python-urllib3 >= 1.10.2-1 for package: python-re
--> Running transaction check
---> Package python-urllib3.noarch 0:1.10.2-5.el7 will be installed
```

Fig. 13 Installation packages.

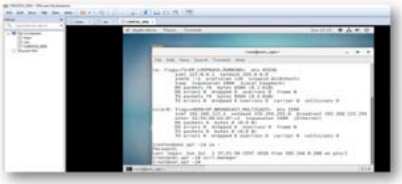
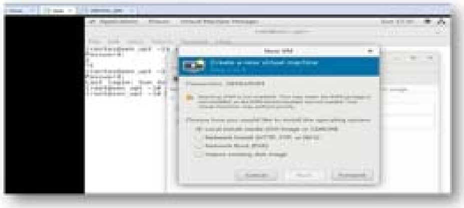


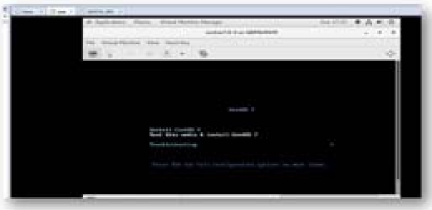
Fig. 14 Network configuration.

After opening Virt-manager and downloading a Centos 7 version of the machine we are working on, we will install another machine, via Virt-manager on Xen. The following Figure.15 and Figure.16 shows the start of server installation with Centos 7 operating system on Xen through Virt-manager.



New VM
Create a new virtual machine
○ Local install media (ISO image or CDROM)
○ Network Install (HTTP, FTP, or NFS)
○ Network Boot (PXE)
○ Import existing disk image

Fig. 15 Centos Installation.



```
CentOS 7

Install CentOS 7
Test this media & install CentOS 7

Troubleshooting

Press Tab for full configuration options on menu items.
```

Fig. 16 Centos Installation progress.

So far we have shown that we have simulated the installation of xen on a linux system and on this xen we have installed another virtual machine, which will work just like any other machine.

We need to install another machine with linux system needed for heartbeat configuration.
To show another advantage of xen, we will not install the second machine according to the process we have already followed, but will clone the first virtual machine we have installed on xen as per Figure.17.

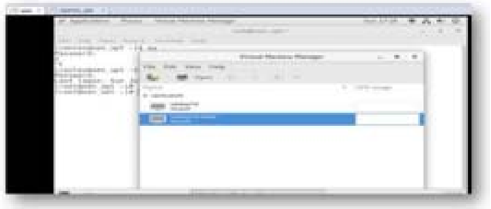


Fig. 17 Clone of second virtual machine.

We have already mounted two virtual machines above the xen which is itself a form of virtualization. In each of the two centos machines, by accessing the putty we install heartbeat and apache as shown in the figures below and make their configurations work as failover. Then we make the migration.

6. Improvements for the Future

6.1 Impact of Failures on Web Services

This project was based on the realization of a system with only one layer of availability and monitoring the “health” of virtual machines. This solution has disadvantages:

- Because of the strong division between the physical machine and the virtual machine, Heartbeat since the physical machine can only control whether or not the virtual machines are running.
- A single availability layer is less tolerant of uncontrolled failures such as power outage because the only way to repair the system is to reboot virtual machines from scratch. It is important to reduce the average repair time (MTTR) in the case of uncontrolled failures.

These two problems can be solved by adding another availability layer (HA). Addition of a second HA layer is not part of this study and has not been tested in test environments. I think its implementation would not be difficult to accomplish and could serve as future work in this regard.

6.2 Voluntary Cooperation

Using the "PULL" method and implementing the migration request in the "start" section of the script we can create the possibility of voluntary collaboration between nodes. The new node where the virtual machine will be sent may choose to receive this machine either by live migration or by restarting it.

One of the most important conditions is that there should be no major architectural differences between the failed node processor and the node where the migration will take place. Using voluntary interaction each node can be provided with a list of nodes suitable for migration to or from them. If the node that failed is not on th list, the new node may choose to reboot the virtual machine instead of risking unstable migration.

Voluntary interaction is beyond the scope of this study but may be considered for further developments and improvements in the case of heterogeneous systems.

7. Conclusions

The standard method of repairing failures consists of shutting down the virtual machine and starting it on another node. This method causes a significant disruption to the services that the virtual machine provided. Another problem with the standard method is the loss of connectivity between users and the virtual machine. For this reason it is necessary for the user to connect to the virtual machine first after the latter has been set up.
In this study we developed an algorithm and implemented it so that Heartbeat can manage virtual machines in high availability clusters.
The solution developed consists of a change made to the inti.d / x0 script and the creation of a "daemon" whose task is to execute at each of the nodes and wait for request to migrate. The init.d / x0 script is used when the virtual machine needs to be set up but, if the virtual machine is running somewhere in the "cluster", this script will send a live migration request instead of rebooting the virtual machine. Preferred to repair system failures is the live migration of virtual machines between the physical cluster nodes. This method minimizes interruptions caused by controlled failures (ie failures caused by shutdown or reboot of the node for maintenance reasons) so that the user does not feel a lack of service.
With regard to uncontrolled failures, immediate failures due to hardware failures, we proposed the addition of a second availability layer (see paragraph 5) as a solution.
The live migration we propose to repair controlled failures ensures a much smaller service outage, small enough to be overlooked by users.
The consequences of network-controlled failures (TCP protocol) were experimentally measured using ICMP

https://doi.org/10.5281/zenodo.4431064

protocol and real services (file transfers, web server and database server). The results show that the outages are extremely short and the client-server connection maintained during migration.

References

[1] Dejan S. Miloji, Fred Douglis, Yves Paindaveine, Richard Wheeler, and Songnian Zhou. Process migration.

[2] Bryan Clark, Todd Deshane, Eli Dow, Stephen Evanchik, Matthew Finlayson, Jason Herne, and Jeanna Neefe Matthews. Xen and the art of repeated research.

[3] http://www.vmware.com/files/pdf/analysts/Future-of-virtualization-IDC.pdf

[4] http://www.cio.com/article/2439494/virtualization/virtualization-definition-and-solutions.html

[5] http://www.losangelescomputerhelp.com/04/the-different-types-of-virtualization/

Ergest Alite graduated from the Faculty of Electrical Engineering, Telecommunication Branch in 2002. He holds an MPA diploma in Economics from 2013 and he has worked as adjunct lecturer at the Polytechnic University of Tirana from 2010. He has an extensive experience in the Information Technology field since 2004, starting from the very bottom, as a specialist, to management positions. Currently he holds the position of Department Manager of Information Technology, NOC & Service Desk Department. He is PHD Candidate at the Polytechnic University of Tirana, Faculty of Information Technology. His work focuses on the analysis of security in the Cloud Computing technology.

Olimpjon Shurdi is a Lecturer in the Department of Electronic and Telecommunications, Faculty of Information Technology, Polytechnic University of Tirana. He received the BE, Ph.D. and Associate Professor from Polytechnic University of Tirana Albania, in 1998, 2009 and 2013 respectively. His research interests include wireless networks, mobile ad hoc networks, wireless sensor networks and multimedia communication in next generation network.

Igli Tafa is a Lecturer in the Department of Computer Engineering, Faculty of Information Technology, Polytechnic University of Tirana. He received the BE, Ph.D. and Associate Professor from Polytechnic University of Tirana Albania. His research interests include virtualization platforms, security over data networks, operating systems and Data Warehouse.