Report Using XEN technology for Green.IT experimentation

Arthur van Kleef *arthur.vankleef@os3.nl*,
System and Network Engineering, University of Amsterdam

February 2, 2009

Abstract

Recent developments in climate concern have lead to changes in technology to reduce energy consumption in order to decrease green house gas emissions, as well as cost reduction.

One of the methods (re)introduced is virtualization. Virtualization makes it possible to consolidate multiple operating systems on one hardware platform, making utilization of hardware resources more efficient. This research project continues these developments by investigating if live migrations of virtual machines can help to offer a way to make use of green energy.

This report discusses choices that have to be considered when designing an infrastructure that allows for live migrations to take place, an example of a possible implementation is evaluated, and it discusses ways to determine where to migrate to for green energy.

Introduction

The University of Amsterdam (UvA) is researching the benefits of using migration technology and virtualization to reduce Green House Gas (GHG) emissions. The intended approach for this research project is to use operating systems running on a virtual machine (VM) that can be migrated based on green energy availability on different locations. To realize this a network of Mac Mini's running XEN/Ubuntu[1][2] will be used to conduct a series of tests. The results from these tests will be used to evaluate the advantages and pitfalls that come from using this technology. The UvA also participates in the Green Next Generation Internet (GreenNGI) effort[1]. Goal of this effort is to have a next generation network using the latest technologies in virtualization and utility infrastructure to provide dynamic networks capable of following the sun or wind, moving Internet traffic where power sources are available. This will be used to evaluate environmental impacts of Green IT strategies. This research project will take place in the context of the course RP1, which is part of the master's program System and Network Engineering[2] at the UvA.

Approach

To familiarize with Xen and live migrations, papers [3][4][5] that focus on this topics will be studied. To design an infrastructure capable of housing live migrating virtual systems using Xen, it will be necessary to investigate the virtualization techniques Xen offers. When a virtual machine migrates to another host, it connects to another point-of-entry in the network, techniques to preserve existing network connections will be addressed. Any operating system

[1] GreenNGI, http://www.greenngi.com
[2] SNE/OS3 Homepage, https://www.os3.nl/

or application works with data stored on disk, because the virtual machine migrates from one machine to another a mechanism needs to be in place that ensures storage is available at all locations. This report will be shaped by the following research questions and their answers.

Research Questions

How do VM migrations take place?

- Will the migration take place on a LAN or WAN network (IP subnets) and how will this affect the design of the experiment setup?
- Will (live) migrations across different locations introduce any subtle problems?
- Are long haul migrations for the sake of using green energy justifiable?

How does one make the decision to start migrating?

- How can green energy yields be measured?
- Is it possible to automate this process?

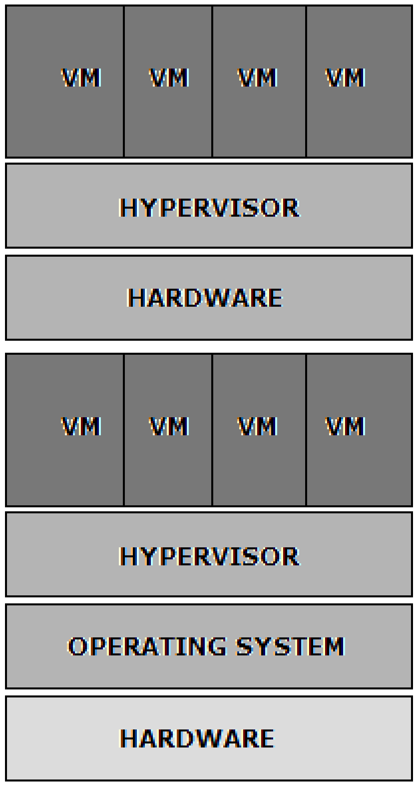
Due to the limited time available, the main focus of this RP will be on performing the migrations using XEN/Ubuntu.

Xen

Xen is a virtual machine monitor (VMM or Hypervisor). A hypervisor is specialized software that simulates a hardware environment in order to run multiple operating systems on a fysical computer (host). Hypervisors can be classified in to two types[7]

- **Type 1** hypervisors run directly on top of the hardware of the fysical computer.
- **Type 2** hypervisors are installed on top of a host operating system that provides virtualization services.

Figure 1: Type 1 vs Type 2Hypervisor



In practice type 1 hypervisors are preferred above type 2 hypervisors. Being able to access hardware directly allows a type 1 hypervisor to achieve higher virtualization efficiency, resulting in higher performance, availability, and security.

Xen is a type 1 hypervisor, not dependent on a host operating system. A Xen system is built in different layers, each representing a certain amount of privilege. The lowest and most privileged layer is the Xen kernel itself. At the layer above Xen hosts guest operating systems by simulating virtual machines (domains). To virtualize the underlying hardware to the virtual machine Xen makes use of the following techniques.

Paravirtualization

Most operating systems are developed to run directly on top of the hardware the system provides. Because some CPU instructions are hard to virtualize, Xen makes use of paravirtualization. It offers the virtual machine a software interface that is similar but not identical to the underlying hardware. So instead of communicating with the hardware directly, the virtualized operating system needs to communicate through Xen. Therefore kernels of operating systems need to be modified so they can communicate with the Xen hypervisor.

Hardware-assisted virtualization

Since Xen 3.0 it is possible to make use of hardware-assisted virtualization using Intel Virtualization Technology (Intel VT[6]), AMD also offers this by implementing AMD-V extensions. This allows Xen to run proprietary operating systems like Microsoft Windows in virtual machines since it is no longer required to make modifications to the operating system's kernel, as is the case in paravirtualization. Hardware-assisted virtualization allows Xen to instruct the CPU to execute code in a virtual machine.

Migration

Xen offers support the migration of virtual machines between physical hosts. This can be useful for administrators to perform load balancing, system maintenance, fault management, and for this project energy savings. The migration is performed by copying the memory pages from the source to the destination virtual machine. Clark et al [4] generalize memory transfer in the following phases.

- ***Push phase*** *The source VM continues running while certain pages are pushed across the network to the new destination. To ensure consistency, pages modified must be re-sent.*
- ***Stop-and-copy phase*** *The source VM is stopped, pages are copied across to the destination VM, then the new VM is started.*
- ***Pull phase*** *The new VM executes and, if it accesses a page that has not yet been copied, this page is faulted in ("pulled") across the network from the source VM.*

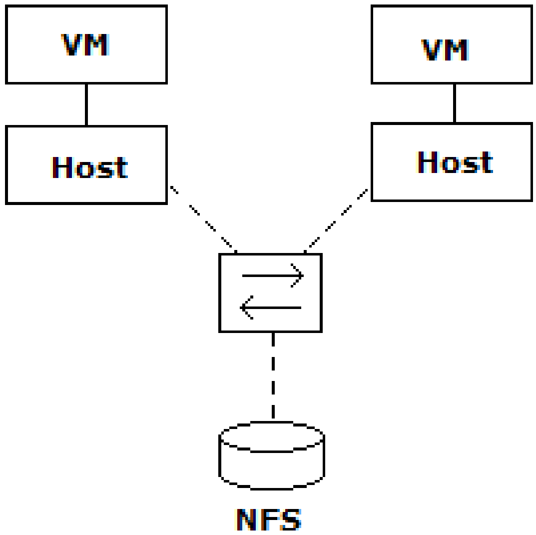
In Xen these are implemented in two migration methods using one or a combination of the memory transfer phases.

- **Regular** migrations are performed by pausing the virtual machine. This involves a *stop-and-copy* approach where the virtual machine is stopped, pages in memory are copied to the destination virtual machine, and the destination virtual machine is started. Since the virtual machine is stopped and only restarted after the memory pages are copied this approach will cause downtime to happen. The amount of downtime depends on the total amount of memory allocated to the virtual machine and the capacity of the network connection between the two fysical machines between which the virtual machine is being migrated.
- **Live** migrations solve the issue of downtime involved with regular migrations by using a *pre-copy* approach. In this approach a combination of the memory transfer phases described. By combining an iterative *push* and a short *stop-and-copy* the downtime that occurs is almost unnoticeable for end users. An iterative push means that this phase is executed more than once. At the first execution all pages in memory are copied to the destination virtual machine. But since during this action pages in memory can change, a second iteration is required to copy the changed files. Until a small set of very frequently updated pages is left over this phase continues, then a short *stop-and-copy* is performed to copy the last small set of pages. After that the destination virtual machine is started and the guest operating system continues to operate.

Setup

This section describes the challenges in the design of the environment and how these are ad-

Figure 2: LAN Setup



dressed.

Network

The experiment will take place in two scenarios. The first will be held on a local area network (LAN) network. The second scenario will take place over on a wide area network (WAN). The details for each will be outlined in the following sections.

LAN

Two Mac Mini's will be connected via a Dell PowerConnect 2716 switch that offers 1 Gbps connectivity over Ethernet. Because the Mac Mini's are shipped with only one Ethernet port, Internet connectivity and LAN connectivity will not be separated. This will affect performance a bit, but because there will be 1 Gbps of bandwidth available not much performance degradation is expected.

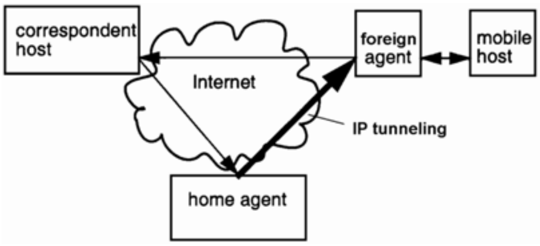
WAN

To investigate the effect of migrating virtual machines across a WAN a connection between the local lab setup will be connected to an experimentation machine at the OS3 lab. Because Xen currently does not implement any features to migrate machines across different subnets it is required to carefully design the connection between the two sites. Possible solutions are implementing techniques like Mobile IP, and IP Tunneling.

- **Mobile IP** allows nodes to roam across the Internet while preserving the IP address through which it can be contacted. Mobile IP makes use of two addresses; the *home address*, an IP address at which the node is permanently available, and a *care-of address*, that is the address assigned to the node on the network it is visiting. A *home agent* running in the network that the home address belongs to keeps track of the real point of presence of the roaming node by binding the home address to the care-of address that is currently assigned to it. Communication to the node is done by sending data to the home address of the roaming node, the home agent receives this data and encapsulates so it can be sent over an IP tunnel to the roaming node. A *foreign agent* decapsulates the packets received from the home agent and forwards them to the node. Communication coming from the roaming node is sent directly through the foreign agent. It will use its home address as source IP address. Caution has to be taken when the network the roaming client is connected to involves routers that perform ingress filtering, in that case *reverse tunneling* can be performed meaning that communication will take place via the home agent.

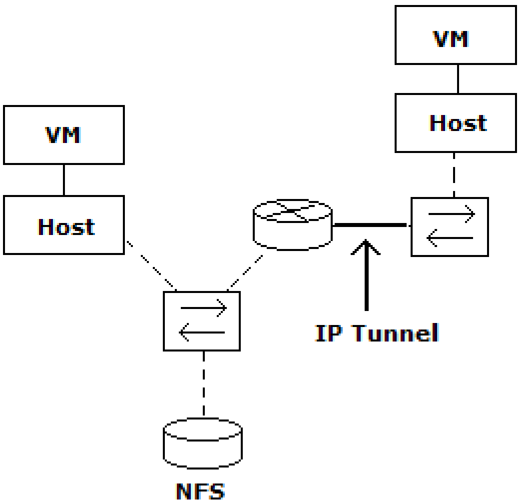
- **IP Tunneling** can be used to transport IP packets that cannot or may not be routed across the underlying infrastructure. Cases where IP tunneling may be used are two networks that do not have a native routing path to each other. A widely deployed protocol is Generic Routing Encapsulation (GRE) which creates a

Figure 3: Mobile IP



virtual point-to-point link between remote networks.

Figure 4: IP Tunneling



Storage

Xen uses a Virtual Block Device (VDB) that holds the filesystem of the virtual machine. Since this project focuses on migrating virtual machines, a mechanism needs to be in place to keep the VDB available for the virtual machine when it migrates to another machine. Several techniques are available to accomplish this goal.

- **Internet SCSI** (iSCSI)[8] is a protocol that allows clients (initiators) to send SCSI commands to SCSI storage devices (targets). It uses TCP/IP networks as the underlying infrastructure. This makes it possible to make storage devices available across routed networks. iSCSI Enterprise Target (IET)[9] is an implementation of a iSCSI target.

- **Network File System** (NFS)[10] is a network filesystem protocol that provides file access across a network. Unlike iSCSI it is not sending commands to a storage device, instead it uses a Remote Procedure Call (RPC) system to access files on the network.

- **Distributed Block Device** (DRDB)[11] works on top of block devices like hard disk partitions, logical volumes and iSCSI devices. It mirrors data that is stored on a block device to another block device hosted by another machine. It is primarily used to create high availability clusters, where one node has the active role and another the passive role that takes over operation when the previously active node fails.

Configuration

This subsection describes the configurations implemented for conducting the experiments. It also motivates the choices for particular techniques used. The lab setup will come in two forms, a setup that will take place on a LAN (UvA lab), and another setup on a WAN (between the OS3 lab and the UvA lab).

The hypervisors will be installed on Apple Mac Mini's consisting of Core 2 Duo processors running at 2.0 GHz, 1 GB RAM and one 1 gigabit Ethernet port. For connecting these a Dell PowerConnect 2716 will be used. The Xen hypervisors will be installed on Ubuntu 8.04.1 LTS. The original plan was to use Ubuntu 8.10, but because this version does not support the Xen hypervisor any longer we decided to fall back to 8.04.

For the LAN setup two Mac Mini's will be configured as Xen hypervisors. A third Mac Mini will be used to provide storage for the virtual machines that will be running on the hypervisors. When investigating the possible solutions it rendered to be difficult to find relevant information about virtual machines booting from an iSCSI target. For this reason we decided to use NFS for hosting the root filesystem of the virtual machine and compile a Xen kernel that supports virtual machines booting from NFS. The network will be configured with static IP addresses and all nodes will be connected to each other at 1 Gbps via the Dell PowerConnect 2716 switch.

Experiments

With the lab setup in place a couple of tests were performed to verify the proces of migration. The virtual machine, running Ubuntu 8.10, functioned as webserver using Apache serving a PHP file that returned configuration data of PHP. A desktop client was used to test the responsiveness of the virtual machine.

Connectivity

The first test conducted was performed by sending ICMP echo requests to the idle virtual machine at the moments before, during, and after the migration took place. This simple test shows how the virtual machine will behave with regards to connectivity:

```
icmp_seq=14 ttl=64 time=0.302 ms
icmp_seq=15 ttl=64 time=0.302 ms
icmp_seq=16 ttl=64 time=0.300 ms
icmp_seq=17 ttl=64 time=5.96  ms **
icmp_seq=18 ttl=64 time=5.12  ms **
icmp_seq=19 ttl=64 time=0.317 ms
icmp_seq=20 ttl=64 time=0.304 ms
icmp_seq=21 ttl=64 time=0.301 ms
```

During idle operation the virtual machine the round trip time (rtt) is stable at approximately 0.300 ms, whereas during the migration the round trip time (marked **) rises to almost 6 ms. After migration was completed active SSH connections were preserved.

Network transfer

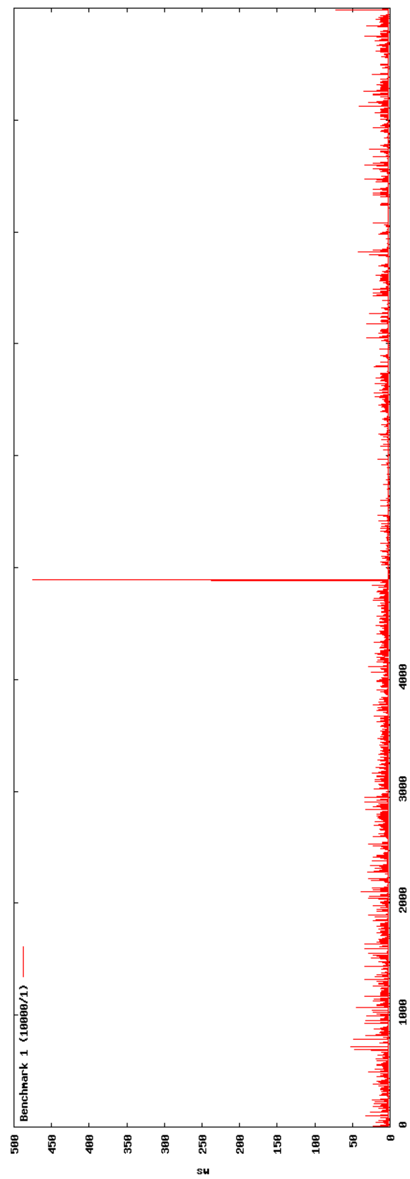
The virtual machine was connected to a remote FTP server on the Internet, downloading a video file of approximately 1 GB. The file was saved to the file system of the virtual machine hosted on the NFS server. The connection to the remote FTP was preserved, and file transfer continued normal. In the FTP log file only a very small drop in average download speed can be detected.

Load

To stress the virtual machine a PHP script (bench.php which is included in source distributions) was used to generate a high CPU load. When migrating the virtual machines there was no notable difference in round trip times when compared to a migration while the virtual machine was idle.

Apache Benchmark was used to benchmark the Apache web server during a live migration, simulating one concurrent connections making 10.000 requests. The output is show in Figure 5. The graph shows that the first 4.000 requests get server at about 50ms, while the virtual machine is migrated the graph show a peak that rises to just under 500ms. After migration is completed the rest of the requests get served within average response times. In a real world scenario these sorts of short latencies are not noticed by end users when it comes to visiting websites.

Figure 5: Responsiveness Apache Web Server



Green energy

In order to reduce greenhouse gas emissions, responsible for global warming, different techniques were developed to yield energy in a greener way to preserve the environment. Energy sources like the sun, the wind and water current can be exploited to create energy that can replace fossil based fuels.

As an effect of the 'Information Age' we currently live in, and the ongoing globalization that takes place, people start to communicate with more and more people around the globe using methods like cell phones, e-mail, instant messaging, read news from all over the world on the World Wide Web, and mobile workers connecting to the office to use their applications.

Information and Communication Technology (ICT) plays a very important role in facilitating these demands. Using specialized application devices to meet the needs of customers all over the world. The Internet also plays a key role in the information flow and is one of the fastest growing form of media.

As a result of using more of these technologies, more energy is required to power all devices involved. In order to preserve the environment and perhaps to reduce costs, many providers are using virtualization so they can use their hardware more efficient. Making use of green energy can be complicated; it is not always available. At night there will be no sunlight, there are days the wind does not seem to blow, and the flow of water also depends on several factors.

So in order to make effective use of green energy resources it is useful to have a system in place that can go where green energy is available. For this Xen already enables virtual machines to be migrated to different locations, but how does one decide when to migrate to a different location?

Forecasting

From the complications described, weather forecasting seems a very logical and easy to implement mechanism to determine the most suitable location for virtual machines to run.

Using metrics like expected hours of sunshine, millimeters of rain, and wind speed, it is be possible to determine the amount of green energy that can be yielded and compare the results of different location across the region, country or even world to seek out the best location for the virtual machine to run. Because of the minimal service interruption that most instances will encounter.

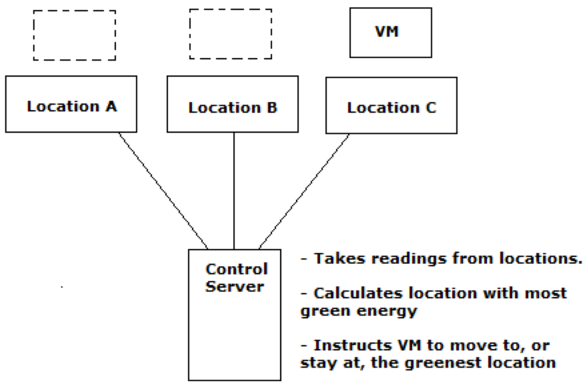
Measuring

Sensors in green energy power plants often provide an interface that can be used to read how much energy is being produced. These readings can be taken by an application that uses the data to calculate the greenest location for the virtual machine to run.

Green Energy & Xen

With the methods that describe how one can determine where green energy resources will be available, it is possible to implement a system that monitors weather conditions and/or monitors the amount of green energy resources available at locations. Such a system could look like is shown in Figure 6

Figure 6: Green Energy Control



Conclusions

The experiments have shown that Xen is excellent in maintaining service continuity during migrations of virtual machines. All applications that were tested remained functioning during and after the migrations took place with no noticeable side effects. Of course in a lab environment it is impossible to test all applications as in real world scenarios. The benchmark performed on Apache shows that when under load, latency can increase to about 500ms. For some applications, like heavy used database servers, this might introduce problems due to increased latency. Applications like video streaming and Voice over IP are examples that might suffer.

When designing an infrastructure that it is intended to perform live migrations, it is important to carefully plan the way storage is offered to the virtual machine. An application running on a virtual machine that is performing a lot of I/O on its file system can drop performance because of higher latencies introduced by migration to a remote location. To prevent this one can choose to make sure a reliable and fast network connection is in place to minimize latency. Another option is to replicate file systems and making data always available close to the virtual machines actual point of presence.

Although not investigated in practice during this research, it is very possible to make use of Xen's migration technology to make efficient use of green energy sources. The possible methods to determine where the most green energy is, or will be available, can be implemented in a system that automatically migrates virtual machines to the 'greenest' location. Of course one has to consider the way storage is handled in long haul migrations, where because of performance requirements a second data store has to be in place. But since most locations already have some kind of Network Attached Storage (NAS) in place, this should not drive energy consumptions much higher.

References

[1] Xen Hypervisor, http://www.xen.org

[2] Ubuntu Home Page, http://www.ubuntu.com

[3] F. Travostino, P. Daspit, L. Gommans, C. Jog, C. de Laat, J. Mambretti, I. Monga, B. van Oudenaarde, S. Raghunath, P. Y. Wang, Seamless live migration of virtual machines over the MAN/WAN, in Future Generation Computer Systems 22 (901-907), 2006

[4] C. Clark, K. Fraser, S. Hand, J. G. Hansen, E. Jul, C. Limpach, I. Pratt, and A.Warfield, Live Migration of Virtual Machines, in: 2nd Symposium on Networked Systems Design and Implementation (NSDI), May 2005

[5] E. Silvera, G. Sharaby, D. Lorenz, I. Shapira, IP Mobility to Support Live Migration of Virtual Machines Across Subnets, IBM Research Report H0264(H0811-004), November 19, 2008

[6] Extending Xen with Intel Virtualization Technology, http://www.intel.com/technology/itj/2006/v10i3/3-xen/1-abstract.htm, *Retrieved on January 14, 2009*

[7] IBM Systems Virtualization, IBM Corporation, Version 2 Release 1 (2005), http://publib.boulder.ibm.com/infocenter/eserver/v1r2/topic/eicay/eicay.pdf *Retrieved on January 15, 2009*

[8] http://tools.ietf.org/html/rfc3720

[9] iSCSI Enterprise Target Home Page, http://iscsitarget.sourceforge.net/

[10] http://tools.ietf.org/html/rfc3530

[11] DRDB Home Page, http://www.drdb.org