Xen을 이용한 가상화

2007-06-04 박원호

1. Xen 설치

- **Xen Domain 0(Dom0) 및 Domain U(DomU)의 개념**

  Xen의 경우, CPU 가상화 기술에 의한 반가상화(Para-virtualization)을 지원하기 때문에 기존의 가상화(VMWare) 등과는 다른 개념이 존재한다.

  Xen은 CPU의 가상화에 따라 Domain 0와 Domain U라는 개념을 제공하는데, 간단하게 말하면, Domain 0(Dom0)는 가상화를 위한 Host가 동작하는 환경을 의미하며, Domain U(DomU)는 게스트가 동작하는 환경을 의미한다. Xen의 리눅스 커널의 경우, Dom0와 DomU를 별도의 커널로 배포하고 있으므로, 주의해서 설치하여야 한다.

  Xen의 성능 향상을 위해서는 Dom0 커널도 실질적으로 Dom0도 기존 리눅스 환경과 동일하지만, 성능 향상을 위해서 가장 가벼운 상태에서 오직 가상화와 I/O, Network 등을 위한 서비스를 포함하는 것이 가장 좋으며, 그 이외의 서비스는 모두 정지시키는 것이 좋다. (Xen 서버 가상화를 위한 솔루션이므로, Xen이 설치된 서버는 반드시 가상화를 위해 사용해야한다.)

  이를 위해서 Xen에서는 XenEnterprize(XenExpress)라는 별도의 커널 및 간단한 리눅스 배포판을 만들어서 제품화 시켜서 판매하고 있다. 이 제품은 가장 기본적인 리눅스 명령어만 포함되어 있으며, sendmail, mysql 등의 프로그램은 포함되어 있지 않다. (심지어, gcc도 포함되어 있지 않다.) 하지만, 이 제품의 경우 원격에서 Xen 가상 서버를 관리하기 위한 Xen Agent와 같은 공개 소스에서 볼 수 없는 별도의 소프트웨어도 함께 제공된다.

- **Xen Dom0의 설치 : 호스트 시스템 마련**

  Xen Dom0의 경우, 가상화가 이루어지면 나머지 모든 소프트웨어 환경은 모두 게스트 OS에 의해 제어되므로, Dom0의 리눅스 커널 버전 및 배포판 종류는 중요하지 않다.

  테스트를 위해서 osbmt 장비의 CentOS 4.4에서 Xen Dom0를 설치하고 테스트하기로한다.

  Xen Download을 위해서 다음과 같은 명령으로 Xen Dom0 커널 및 관리 도구를 다운로드 받는다. 관리도구는 Python으로 작성되어 있다.

```
# wget http://bits.xensource.com/oss-xen/release/3.1.0/kernel-3.1-
rhel4x/RPMS/i386/xen-3.1.0-1.i386.rpm
```

```
# wget http://bits.xensource.com/oss-xen/release/3.1.0/kernel-3.1-
rhel4x/RPMS/i386/kernel-xen-2.6.18-3.1.0.i386.rpm
# -- 네트워크 가상화를 위한 Network bridge 유틸리티 설치
# yum install -y bridge-utils SDL
# rpm -ivh xen-3.1.0-1.i386.rpm
# rpm -ivh kernel-xen-2.6.18-3.1.0.i386.rpm
# -- Xen 서버들이 서버 기동시 자동으로 기동 되도록 서비스 설치(Optional)
# chkconfig --add xendomains
```

Dom0가 배포되는 커널의 버전은 2.6.18이고, CentOS 4.4의 커널 버전은 2.6.9이므로, Dom0 설치시 커널 버전 차이로 인한 오류가 발생한다. (오류의 내용은 iksscsctp-tools 버전의 차이로 인한 것이다.) 이 오류를 해결하기 위해서 iksctp-tools를 제거한다. 설치 제거가 완료되면 커널을 다시 설치한다.

```
# rpm -e lksctp-tools-devel
# rpm -e lksctp-tools
```

http://www.karan.org/blog/index.php/2005/12/06/xen3_on_centos4

커널 설치가 완료되면, /boot/grub/grub.conf를 수정하여, Xen Dom0 커널로 부팅할 수 있도록 한다.

```
default=0
title Xen Dom0 (2.6.18-xen_3.1.0)
        root (hd0,0)
        kernel /xen-3.1.0.gz
        module /vmlinuz-2.6.18-xen_3.1.0 ro root=/dev/VolGroup00/LogVol00
        module /initrd-2.6.18-xen_3.1.0.img
```

서버 부팅시 자동으로 xend와 xendomains(Xen Server 자동 실행 스크립트)으로 실행될 수 있도록 다음과 같은 명령을 실행한다.

```
# chkconfig --add xend
# chkconfig --add xendomains
```

네트워크 카드가 2개 이상 설치되어 있는 경우, 2개의 네트워크 카드를 모두 Bridge로 동작시켜야하므로, 다음과 같이 설정을 추가/변경한다. (네트웍 카드가 1개일 경우는 수정할 필요가 없다.)

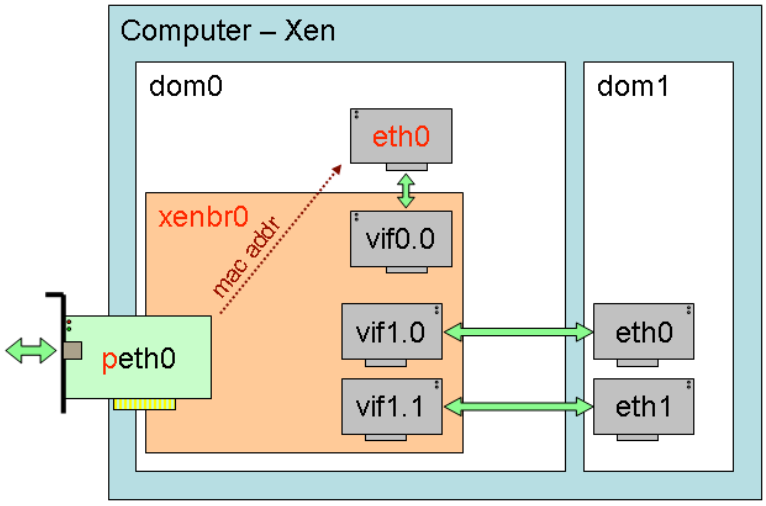
```
# vi /etc/xen/xen-config.sxp
-- 삭제
(network-script network-bridge)
-- 추가
(network-script 'network-bridge-multi')

# vi /etc/xen/scripts/network-bridge-multi
#!/bin/sh
dir=$(dirname "$0")
"$dir/network-bridge" "$@" vifnum=0
"$dir/network-bridge" "$@" vifnum=1
```

위와 같이 설정을 하면, 서버를 리부팅하고, ifconfig, network-bridge 등의 명령을

수행하면 다음과 같은 결과를 볼 수 있다. Xen에서 실제 물리적 네트워크 카드는 가상 네트워크 카드로 Bridge가 생성되고, 가상 머신은 Bridge된 네트워크 카드를 통해서 네트웍에 연결된다. (NAT 등의 구성도 가능하지만, 서버 서비스 환경에서는 큰 의미는 없다.) peth*, vif*, xebr* 등의 네트워크 인터페이스는 network-bridge 스크립트에서 동적으로 생성하는 것이므로, 이 스크립트가 실행되지 않으면, 하드웨어의 eth*만 ifconfig에서 나타난다. Network-bridge 스크립트는 xend가 시작될 때, xen-config.sxp 파일을 읽어서 실행된다.

```
# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:04:23:DC:72:32
...

eth1      Link encap:Ethernet  HWaddr 00:04:23:DC:72:33
...

lo        Link encap:Local Loopback
...

peth0     Link encap:Ethernet  HWaddr FE:FF:FF:FF:FF:FF
...

peth1     Link encap:Ethernet  HWaddr FE:FF:FF:FF:FF:FF
...

vif0.0    Link encap:Ethernet  HWaddr FE:FF:FF:FF:FF:FF
...

vif0.1    Link encap:Ethernet  HWaddr FE:FF:FF:FF:FF:FF
...

vif2.0    Link encap:Ethernet  HWaddr FE:FF:FF:FF:FF:FF
...

vif2.1    Link encap:Ethernet  HWaddr FE:FF:FF:FF:FF:FF
...

xenbr0    Link encap:Ethernet  HWaddr FE:FF:FF:FF:FF:FF
...

xenbr1    Link encap:Ethernet  HWaddr FE:FF:FF:FF:FF:FF
...

# /etc/xen/scripts/network-bridge status
============================================================
3: eth0: <BROADCAST,MULTICAST,UP,10000> mtu 1500 qdisc noqueue
...
13: xenbr0: <BROADCAST,MULTICAST,UP,10000> mtu 1500 qdisc noqueue
    link/ether fe:ff:ff:ff:ff:ff brd ff:ff:ff:ff:ff:ff

bridge name     bridge id               STP enabled     interfaces
xenbr0          8000.feffffffffff       no              peth0
                                                        vif0.0
xenbr1          8000.feffffffffff       no              peth1
                                                        vif0.1
```

```
**.**.**.*/24 dev eth1  proto kernel  scope link  src **.**.**.*
**.**.**.*/24 dev eth0  proto kernel  scope link  src **.**.**.*
169.254.0.0/16 dev eth1  scope link
default via **.**.**.* dev eth0

Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
**.**.**.*      0.0.0.0         255.255.255.0   U     0      0        0 eth1
**.**.**.*      0.0.0.0         255.255.255.0   U     0      0        0 eth0
**.**.**.*      0.0.0.0         255.255.0.0     U     0      0        0 eth1
0.0.0.0         **.**.**.* 0.0.0.0         UG    0      0        0 eth0
```



http://wiki.xensource.com/xenwiki/XenNetworking

위의 설정이 마무리되면, 서버를 재기동 시키면 된다. 서버가 재기동 된 다음, 커널의 버전을 확인하면 아래와 같다.

```
# uname -a
Linux osbmt 2.6.18-xen_3.1.0 #1 SMP Thu May 17 05:35:24 EDT 2007 i686 i686 i386
GNU/Linux
```

** 주의 :

Xen의 경우, 하드웨어 가상화 기능 사용하여 가상화를 수행하므로, CPU가 가상화를 지원해야한다. 사용 가능한 하드웨어는 Intel VT 기술이 적용된 Xeon 또는 AMD-V 기술이 적용된 Opteron 등이다.

CPU 가상화 기술을 활용하기 위해서는 서버 BIOS에서 하드웨어 가상화(Hardware Virtualization) 기능을 활성화 시켜주어야 한다.(기본적으로는 비활성화 되어 있다.)

- **Xen DomU의 템플릿 이미지 작성 – 디스크 이미지 생성 및 새로운 템플릿 작성**

Domain U(DomU)는 게스트 시스템을 의미하며, 실제 가상화 기능을 활용할 시스템을 의미한다. 가상화가 이루어지면, 모든 애플리케이션은 DomU에서 동작하는 것을 기본으로 한다.

Xen의 DomU는 VMWare와 같은 전가상화(Full Virtualization)과 상당히 다른 형태를 보인다.
먼저 전가상화의 경우, 시스템 전체를 완전 가상화하여 시스템의 BIOS부터 CPU, Memory, I/O 등을 완전히 에뮬레이션하여 가상화는 하는 방법을 사용한다. 그래서 가상화가 이루어지면, 호스트 시스템과는 전혀 관계없는 독립된 하드웨어로 인식된다.
하지만, Xen의 반가상화(Para-virtualization)의 경우, 하드웨어 전체를 가상화하는 대신에 가상화가 적용된 리눅스 커널을 시스템 부팅시 적용하므로써 모든 장치를 가상화 하는 것은 아니다. CPU, 메모리 등은 하드웨어의 가상화 기능을 이용하며, 디스크 장치는 리눅스의 마운트 기술을 그대로 이용하므로, 마운트 대상 미디어는 물리 디스크, 논리 파티션, 디스크 이미지 등 리눅스에서 사용 가능한 모든 장치이다.
이러한 가상화 방식으로 인하여, DomU를 기동시키기 위한 설정 파일에는 리눅스 부팅 커널, 디스크 이미지 위치 등을 지정하는 옵션들이 존재한다.

가상화 디스크는 물리적인 장치(/dev/hda 등) 또는 논리적인 파티션(저널링 파일 시스템 /dev/mapper/VolGroup00-LogVol00 등)을 사용할 수 있다. 이러한 경우, 가상화 커널에서 하드웨어에 바로 접근하여 I/O가 이루어지므로, 높은 성능을 발휘할 수 있지만, 디스크 관리 등의 별도의 작업이 필요하다.

이번 테스트에서는 하드웨어 등의 제약으로 인하여, 디스크 이미지를 생성하고 이 디스크 이미지 가상 디스크로 사용하는 방법으로 테스트를 진행하였다.

이미지 디스크 생성을 위해서 리눅스의 dd와 mkfs 명령을 이용해서 이미지 생성 및 파일 시스템 생성을 수행한다.

```
# -- 1G짜리 디스크 이미지 생성
# dd if=/dev/zero of=/vmimages/testvm.img bs=1k seek=1024k count=1
# -- 생성된 디스크 이미지를 ext2로 포멧
# mkfs -t ext2 /vmimages/testvm.img
```

디스크 이미지 생성 완료 후, 설정 파일 및 각종 응용프로그램을 생성된 디스크로 복사한다. 이때, 복사되는 프로그램 및 설정 파일은 모드 호스트 시스템의 내용이므로, 만일 호스트 시스템이 이미 다른 용도로 사용되었던 시스템일 경우, 좋은 방법은 아니다. 이 경우, 새로운 디스크에 리눅스를 설치하고, 설치된 디스크를 마운트 시킨 후, 복사하는 것이 효율적이다.

```
# -- 디스크 이미지 마운트
# mount -o loop /vmimages/testvm.img /mnt
# -- 필수 디렉토리 복사
# cp -ax /{dev,var,etc,usr,bin,sbin,lib} /mnt
# -- 몇몇 디렉토리를 생성
# mkdir /mnt/{root,proc,sys,home,tmp}
```

위의 작업이 끝나면 /etc/fstab 등의 부팅시 설정 파일을 수정한다. 디스크 이미지는 /dev/sda1으로 마운트된다.

```
# cat > /mnt/etc/fstab
/dev/sda1               /                ext2     defaults       1 1
```

위의 작업이 모두 끝나면, 이미지를 마운트에서 제거한다.

```
# umount /mnt
```

http://tx.downloads.xensource.com/downloads/docs/user/#SECTION03320000000000000000

디스크 이미지 생성이 완료되면, /etc/xen/testvm.hvm 파일을 작성한다. 설정 파일에는 게스트 시스템에서 사용될 디스크 이미지와 kernel 정보가 표시되어 있다.

```
# cat > /etc/xen/testvm.hvm
disk     = [ 'file:/vmimages/testvm.img,sda1,w']
memory = 512
root='/dev/sda1'
kernel = "/boot/vmlinuz-2.6.9-42.0.3.EL.xs0.4.0.263xenU"
```

디스크 이미지 생성 및 설정 파일이 모두 작성되었으면, Xen 게스트 시스템을 실제로 생성하면 된다.

```
# cd /etc/xen
# xm create -c testvm.hvm
```

- **Xen DomU의 템플릿 이미지 작성 - 기존 디스크 이미지를 이용**

  DomU 디스크 이미지를 기존 시스템을 기준으로 작성할 수 있지만, 하드웨어 종속적인 설정 파일 및 필요 없는 패키지들도 함께 설치되어 있으므로, 효율이 반드시 좋다고 할 수 없다.

  이러한 상태에서 인터넷에서는 각각의 리눅스 배포판을 Xen에 설치한 후에, 디스크 이미지만을 배포하는 사이트도 존재한다. 이번 테스트에 사용된 사이트는 http://jailtime.org/으로 CentOS 및 Fedora Core 등을 최소 패키지로 설치한 디스크 이미지를 배포하고 있다.

  디스크 이미지를 설치하기 위해서는 http://jailtime.org/download:centos:v4.4에서 디스크 이미지를 다운로드하고, 압축 해제 후, 이미지 파일과 swap 파일 파일을 /vmimages 디렉토리로 복사한다. 함께 포함된 VM 설정 파일은 /etc/xen 디렉토리로 복사한다.

```
tar xvfj centos.4-4.20061223.img.tar.bz2
mv centos.4-4.img /vmimages/centos4.4.img
mv centos.swap /vmimages/centos4.4.swap
mv centos.4-4.xen3.cfg /etc/xen/centos.4.4.hvm
```

다운로드한 이미지는 최소한의 패키지만 설치되어 있으므로, gcc 등의 패키지를 추가로 설치하기 위해서는 디스크 이미지의 크기를 2.5G 정도로 늘려준다.

```
dd if=/dev/zero of=/vmimages/centos4.4.img bs=1M conv=notrunc count=1 seek=2500
losetup /dev/loop0 /vmimages/centos4.4.img
e2fsck -f /dev/loop0
resize2fs /dev/loop0
e2fsck -f /dev/loop0
losetup -d /dev/loop0
```

이미지를 사용하기 위한 설정 파일을 다음과 같이 수정한다.

```
# cat > /etc/xen/centos4.4.hvm
kernel = "/boot/vmlinuz-2.6-xenU"
vcpus = 2
memory = 512
name = "centos.4-4"
vif=[ 'bridge=xenbr0,mac=','bridge=xenbr1,mac=' ]
dhcp = "off"

disk = ['file:/vmimages/centos4.4.img,sda1,w', 'file:/vmimages/centos4.4.swap,sda2,w']
root = "/dev/sda1 ro"
```

설정 파일 작업이 모두 끝났으면, xm 명령으로 가상 서버를 기동시킨다. 기동된 후, 최초의 root 패스워드는 'password'이다.

```
# cd /etc/xen
# xm create -c centos.4.4.hvm
```

부팅이 완료된 후, 임시로 IP 주소를 설정하고, DNS도 설정하여 yum 등을 이용해서 필요한 패키지를 설치 할 수 있도록 한다. (아래의 명령은 가상 서버에서 root 권한으로 실행한다.)

```
$ ifconfig  eth0 **.**.**.* netmask 255.255.255.0 up
$ route add -net 0.0.0.0 netmask 0.0.0.0 gw **.**.**.*
$ cat > /etc/resolv.conf
nameserver **.**.**.*
```

네트웍 설정이 완료되면, yum을 이용해서 필요한 몇몇 패키지를 추가로 설치한다. 최초의 디스크 이미지에는 vi, gcc 등이 설치되어 있지 않다.

```
$ yum groupinstall "Administration Tools" "Development Tools" "Editors"
```

- **Xen DomU에서 윈도 설치**

Xen은 반가상화를 지원하며, 리눅스 이외에 x86 기반의 윈도와 같이 다양한 OS를 설치할 수 있다. (이러한 기술과 관련하여, MS와 SuSE는 서로 기술 협력을 맺기로

계약을 체결했다.)

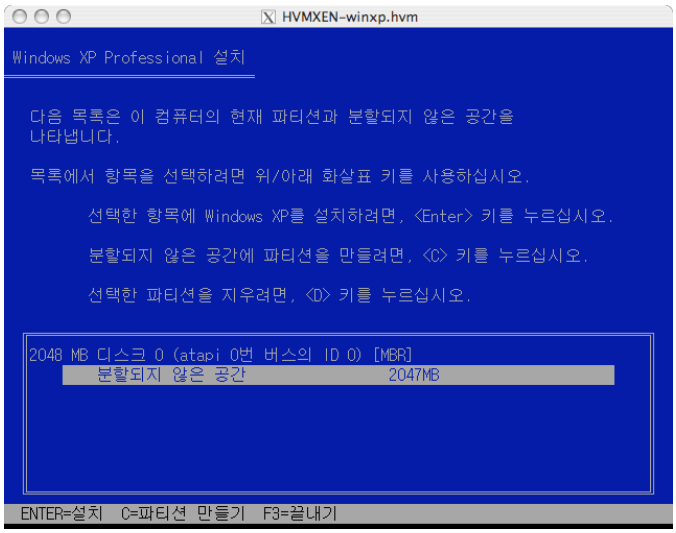
윈도의 경우, 리눅스 커널과 완전히 다른 구조를 가지고 있고, 외부에서 커널의 수정도 불가능하므로 Xen에서 지원하는 반가상화보다는 전가상화에 가까운 방법을 제공한다. Xen으로 이미지를 부팅하면 VMWare와 같이 BIOS 부팅화면부터 시작해서 일반적인 PC 부팅과정과 동일하게 이루어진다.
Xen에서는 실제로 윈도 설치 등을 위해서
qemu http://fabrice.bellard.free.fr/qemu/라는 시스템 가상화 기술을 사용한다.
현재까지의 상황으로 봐서는 Xen에서 윈도 구동은 리눅스에 비해서 상당히 성능이 떨어질 것으로 예상된다.

테스트에서는 Windows XP 설치까지만 시도하였다.
윈도의 경우, 리눅스 설치와 달리 커널을 커널을 'hvmloader'을 이용해서 부팅을 시작해야한다. 디스크 등의 마운팅 방법은 리눅스의 경우와 동일하다.
윈도 설치를 위해서 CD 또는 ISO 파일을 이용할 수 있으며, 리눅스와 달리 부팅할 드라이브의 위치를 'boot' 옵션으로 지정해주어야 한다. (설치시에는 'c', 설치 완료 후에는 'd'로 변경한다.)

```
# dd if=/dev/zero of=/vmimages/winxp.img bs=1k seek=2048k count=1
# cat >> /etc/xen/winxp.hvm
kernel = "/usr/lib/xen/boot/hvmloader"
builder = "hvm"
vif = [ 'type=ioemu, bridge=xenbr0' ]
disk     =
[ 'file:/vmimages/winxp.img,ioemu:hda,w','file:/tmp/MSWindows_XP.iso,hdc:cdrom,r']
vcpus = 2
memory = 384
boot='d'
sdl=1
vncviewer=0
```

윈도 설치는 X-window에서 이루어 지므로, 콘솔에서 X 윈도우를 실행한 후에 xm 명령으로 가상 서버를 실행한다.

```
# xm create -c winxp.hvm
```



나머지 설치 과정은 기존의 윈도 설치 과정과 동일하다.

http://www.planetjoel.com/viewarticle/568/HOWTO:+Windows+XP+running+under+Xen+3.0+on+Ubuntu+Dapper+Drake
http://en.opensuse.org/Xen_Full_Virtualization_Example#Create_a_Blank_Disk_Image_to_install

2. 결론

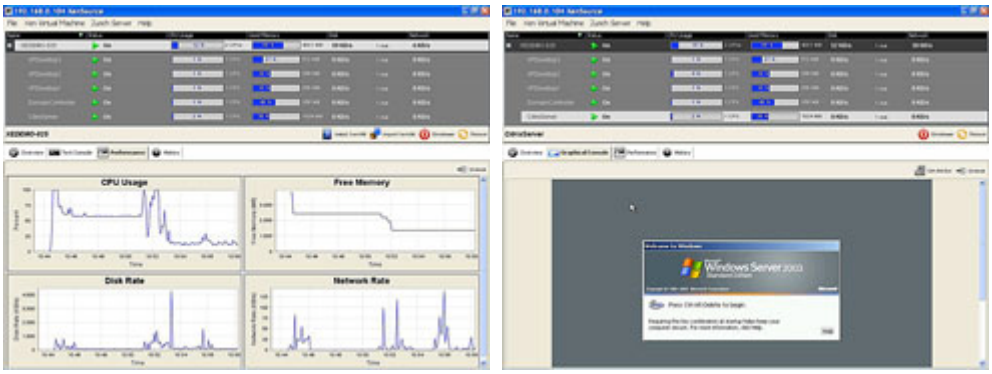
Xen은 반가상화를 사용하는 리눅스 기반의 가상기술이다. VMWare와 같이 호스트 시스템과 게스트 시스템이 서로 다른 OS를 사용할 수 있어, 2가지 이상의 리눅스 버전을 사용하거나, 윈도와 리눅스를 동시에 가상화 할 경우 상당한 장점을 가진다.
또한, VMWare와 달리 모든 하드웨어 리소스를 가상화하는 것이 아닌 CPU의 가상화 기술을 이용해서 Domain0와 DomainU로 구분하여 프로세스 영역을 나누어 성능을 극대화 하고, 디스크 등의 경우 리눅스 커널 레벨에서 가상화를 지원하여 오버헤드를 최소화 시켰다.

호스트 시스템의 경우, Xen용으로 수정된 커널을 사용해야 하며, 게스트의 경우 Xen용 커널을 사용하거나, 윈도와 같이 수정되지 않은 일반 OS도 사용할 수 있지만 성능은 그리 높지 않을 것으로 예상된다.
게스트 시스템이 설치되면, 모든 환경이 기존의 시스템 환경을 그대로 사용할 수 있으므로, 마이그레이션이 상당히 쉬운 편이다.

신규 장비 도입을 위해서는 미리 템플릿 디스크 이미지를 생성해 놓고, 그 이미지 파일을 복사하면 간단히 새로운 서버용 디스크를 만들어 낼 수 있다. Xen의 설정 파일은

Python을 그대로 사용하므로, 적절히 변형하면 게스트 서버 클러스터를 자동화 시킬 수도 있다.
만일, NAS 또는 GFS와 같은 디스크 스토리지가 있을 경우, 디스크 이미지를 모두 공통의 스토리지에 저장하고, 각각의 서버는 디스크 없이(Host OS가 올라갈 정도의 최소한의 디스크만 가지고) 스토리지의 이미지를 읽어서 서버를 동작시키면, 자원 할당에 따른 서버 풀링도 가동할 수 있다. 이 경우, 기존의 하드웨어 고장에 의한 서비스 중지 현상은 최소화 시킬 수 있다.

Xen은 자체의 상용 제품도 판매하고 있는데, 상용으로 판매되는 제품에서는 Xen을 위한 서버 커널이 포함된 특화된 리눅스 배포판과 리모트에서 서버의 통합 관리가 가능한 콘솔 클라이언트를 포함하고 있다. Xen Express의 경우 무료이며, 2 CPU, 4개의 가상 머신까지 지원하며, Enterprise의 경우 448달러부터 가격이 부가된다.
원격에서 각각의 가상 머신을 관리할 수 있으며, 원격 클라이언트는 자바로 작성되어 있으므로, 윈도에서도 관리가 가능하다.



http://www.xensource.com/products/xen_enterprise/xenenterprise_demo.html

이러한 제품을 사용하기 위해서는 호스트 시스템을 Xen에서 제공하는 리눅스로 다시 설치해야 한다.(이 호스트 Dom0는 오직 Xen 서버를 동작하기 위한 역할만을 제공한다.) 하지만, OS 이미지를 위한 디스크는 NFS 또는 GFS 등을 이용하여 마운트하여 사용할 수 있다.

오픈 소스에서도 이런 관리 콘솔을 제공하는 프로젝트들이 있다. 레드햇의 경우, RHEL 5에서 패키지에 포함되어 있으며, SuSE 10에도 GUI 콘솔이 포함되어 있다. (Python을 기반으로 하여, X-Windows에서 작성되었다.)
http://www.enomalism.com/ : Open Source/상용 관리자
http://virt-manager.et.redhat.com/ : RedHat 공식 Xen 관리자

- 호스트 시스템과 관계없이 리눅스의 설치, 관리가 완전히 독립되어 있으므로, 기존의 시스템 관리자는 매우 편리하게 설치 및 관리가 가능하다/
- 게스트 시스템은 모두 독립적으로 동작하므로, 호스트 시스템에서 게스트 시스템으

로 직접 접근하는 것은 거의 불가능하다. ssh, rlogin 등을 통해서 접근해야 한다. 이러한 이유로, 통합적인 시스템 관리는 한계가 있다.

- 설정 파일 자체가 python으로 작성되어 있으므로, 동적으로 시스템 관리를 수행할 수 있다. 단, python 자체에 대한 어느 정도의 이해가 필요하다. 또한, 모든 관리 툴이 python을 기준으로 작성되어 있다.
- 많은 리눅스 관계자들(배포판 업체 및 해커)가 Xen을 지지하고 있으며, 최근 공식 커널 소스 트리에 포함되기도 하였다. 하지만, 아직 문서자료는 공식적인 문서 자료가 부족한 편이며, 대부분 사용자의 How-to 문서를 참조해야한다. 하지만, 이 부분도 Xen 버전이 변경되면서 실제로 동작하지 않는 부분도 있다. (특히 윈도 설치 및 네트워크 설치 부분은 자료가 상당히 부족해서, 많은 시간을 소요하였다.)
- 가상화의 방식이 VMWare와 다른 방식이므로, 처음 Xen을 접할 때 상당한 개념상 차이를 이해해야 한다.
- CPU가 가상화를 지원해야 하므로, 구형 하드웨어에서는 동작이 불가능하다. (사실상 큰 의미는 없다.)
- 디스크가 이미지 또는 물리적/논리적 파티션을 그대로 이용할 수 있으므로, 디스크 관리가 좀더 유동적이고 관리가 편리하다.