ISBN 978-89-6211-646-5

가상화 기술 및 가상화 시스템 관리 기술 분석

2010년 11월

슈퍼컴퓨팅본부
기반기술개발실

최영리, 최동훈, 박상배, 박동인

KiSTi 한국과학기술정보연구원
Korea Institute of Science and Technology Information
www.kisti.re.kr

목 차

제 1 장 개요

가상화 기술은 하나의 물리적 서버에 여러 개의 가상 머신(VM)을 생성하고 각 VM이 독립적인 개별 하드웨어를 가지고 있는 것처럼 만들어 준다. 가상화 기술을 이용하여 실제 이용률이 최대 사용 가능한 양에 비해 현저히 떨어지는 여러 서버를 통합함으로써 서버 자원을 효율적으로 사용할 수 있다. 또한, 단일 물리적 서버에 다양한 운영체제를 동시에 실행시킬 수 있어 다양한 시스템 구성을 가능하게 한다.

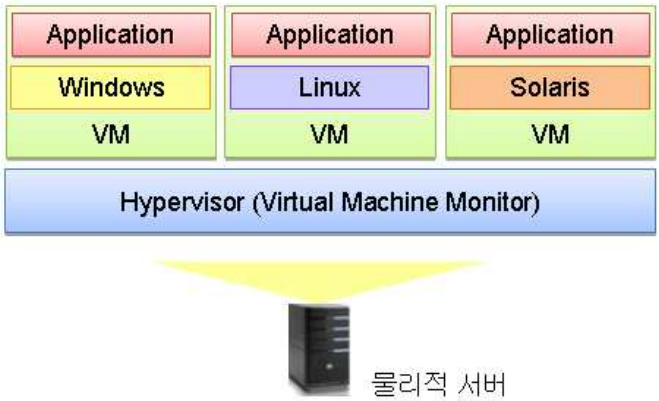
이런 가상화 기술은 자원 이용률(resource utilization), 관리성(manageability) 등을 높이고 가상 머신간의 결함을 격리(fault isolation)하는 이점을 제공한다.

범용 x86 서버의 가상화를 위해서는 hypervisor라는 software가 필요하다. 하이퍼바이저는 물리적 서버의 hardware를 가상화하고, 물리적 서버의 자원을 동적으로 가상 머신들에 할당하는 역할을 한다. 또한 가상 머신의 가상 CPU를 물리적 core/CPU에 스케줄링하는 것이 요구 된다. 제 2장에서는 현재 가상화 기술 동향과 하이퍼바이저 기술에 대해 소개한다. 또한 하이퍼바이저의 가상 머신 스케줄링 기술에 대해 소개한다.

가상 머신의 효율적인 관리와 제어는 가상화 시스템의 성공을 위하여 반드시 필요하다. VM migration 기술은 downtime 없이 실행 중인 가상 머신을 다른 물리적 서버로 이주하는 기술로, 자원을 유연한 할당과 서버 유지 보수 등에 매우 유용하게 쓰인다. 가상화 시스템에서는 자원을 보다 유연하게 사용하는 것이 가능한데 이를 위해 한 서버 안에서의 자원 관리 기술 및 여러 서버의 자원을 함께 고려하는 기술들이 사용되고 있다. 가상화 시스템에서 발생하는 결함을 위한 HA(high availability)와 fault tolerance와 같은 기술들을 이용하여 가상화 시스템을 reliable하게 만드는 것이 가능하다. 또한 가상화 시스템에서 application software와 operating system을 package로 쉽게 deploy 할 수 있게 virtual appliance를 이용 할 수 있다. 제 3장에서는 가상화된 여러 서버로 구성된 시스템을 효율적으로 관리하고 제어하기 위한 여러 기술을 소개한다.

제 2 장 가상화 기술 분석

가상화 기술은 하나의 물리적 서버에 여러 개의 가상 머신(VM)을 생성하고 각 VM이 독립적인 개별 하드웨어를 가지고 있는 것처럼 만들어 준다. 또한, <그림 1>과 같이 단일 물리적 서버에 다양한 운영체제를 동시에 실행시킬 수 있어 다양한 시스템 구성을 가능하게 한다.



<그림 1> 가상화 기술

가상화 기술은 다음과 같은 이점을 제공한다.

- 자원 이용률(resource utilization) 증가: 가상화 기술을 이용하여 실제 이용률이 최대 사용 가능한 양에 비해 현저히 떨어지는 여러 서버를 통합(consolidate)함으로써 서버 자원을 효율적으로 사용할 수 있다.
- 결함 격리(fault isolation): 같은 물리적 서버에서 동시에 실행되고 있는 가상 머신에 failure가 발생하더라고 다른 가상 머신에 영향을 끼치지 않는다.
- 관리성(manageability) 증가: 여러 종류의 operating system을 동시에 지원하는 것이 가능하다. 가상 CPU의 수, 메모리 양 등 다양한 가상 머신 구성이 가능하다. 또한, 자원을 동적으로 실행 중인 가상 머신에 할당하는 것이 가능하여 자원을 유연하게 이용하는 것이 가능해 진다.

따라, 가상화 기술을 통하여 시스템 운영 비용을 절감하고, 시스템 자원 낭비를 막을

수 있다. 또한 같은 양의 자원으로 더 많은 사용자를 지원 하는 것이 가능 할 뿐만 아니라 다양한 사용자 요구 사항을 만족 시키는 시스템을 제공하는 것이 가능해 진다.

제 1 절 가상화 기술 동향

기본적으로 범용 x86 서버의 가상화를 위해서는 가상화를 가능하게 하는 software인 hypervisor 또는 Virtual Machine Monitor(VMM)가 필요하다. Open source로 시작한 hypervisor와 그렇지 않은 hypervisor로 구분 할 수 있다.

- Xen[15] (open source)
- VMware ESX[16], Microsoft Hyper-V[17]

Xen과 VMware ESX 하이퍼바이저에 대해서는 다음 절에 자세히 설명하겠다.

가상화 기술을 효율적으로 사용하기 위해서는 여러 가상 머신을 관리하는 가상 머신 관리 layer (Virtual machine management layer)가 반드시 필요하다. 이를 위한 software가 개발되고 있고 이 또한 open source로 시작한 관리 software와 그렇지 않은 software로 구분된다.

- Eucalyptus[18], OpenNebula[19]
- VMware vSphere[1], Microsft System Center Virtual Machine Manager (SCVMM)[10]

특히 VMware의 관리 software는 가장 먼저 시작하겠기 때문에 다른 software에 비해 현재 가장 성숙한 상태이고 매우 다양한 기능들을 제공한다. 따라 이 문서에서 현재 기술을 파악하기 위해 여러 VMware의 기술을 참조 하였다.

범용 x86 서버 가상화는 기업용 서버에서 먼저 받아들여지고 있는데 Fortune 100에 속하는 모든 기업들이 기업 데이터센터를 가상화하여 사용하고 있다. 이는 데이터센터를 유지하고, 운용 관리하는데 비용을 많이 절약 할 수 있기 때문이다. 따라, 현재 가상화 기술은 기업용 application 지원에 중심을 두고 발전 해 오게 되었다.

가상화 기술의 발전으로 가상화 기반 클라우드 컴퓨팅 서비스가 등장 하였다. 예로 Amazon EC2[14]와 VMware vCloud Express[20]가 있는데 앞으로 더 많이 확산 될 것으로 예상되고 있다. 클라우드 컴퓨팅은 utility 컴퓨팅 모델을 가지고 있는데 사용자는 물리

적 자원을 구매, 운영, 유지 보수 할 필요 없이 클라우드 컴퓨팅 서비스에서 제공하는 자원을 이용하고, 사용한 만큼만 돈을 지불하게 된다.

제 2 절 하이퍼바이저와 가상 머신

하이퍼바이저 또는 VMM (Virtual machine monitor)는 다양한 operating system을 가진 여러 가상 머신이 하나의 물리적 머신에서 동시에 실행되는 것을 가능하게 하는 software로 물리적 hardware와 가상 머신의 guest operating system 사이에 존재한다. 이 hypervisor는 각 가상 머신에 가상 hardware를 제공하여 독립적인 hardware를 가진 것처럼 만들어 주는 역할을 한다. 하이퍼바이저는 그 위에서 실행되고 있는 가상 머신들에게 물리적 머신의 자원을 동적으로 할당한다.

가상 머신은 processor, memory, networking, storage 등의 hardware부터 operating system 및 application software까지 모든 요소를 encapsulate하고 있다. 기본적으로 가상 머신은 물리적 머신과 똑같이 동작하여, 사용자는 가상 머신과 물리적 머신을 구분 할 수 없다. 물리적 머신과 마찬가지로 멀티코어를 가질 수 있는데 이 경우 가상 머신은 여러 개의 가상 CPU를 가지게 된다.

가상 머신은 여러 개의 파일로 구성 된다. 가상 머신의 hardware, disk 정보들을 포함하는 meta 파일과, 가상 머신의 disk 파일들로 주로 구성 된다. (예 vm1-meta.vmdk, vm1-volume1.vmdk, vm1-volume2.vmdk)

다음은 Xen hypervisor와 VMware의 ESX의 구성 요소에 대하여 설명 한다.

Xen Hypervisor

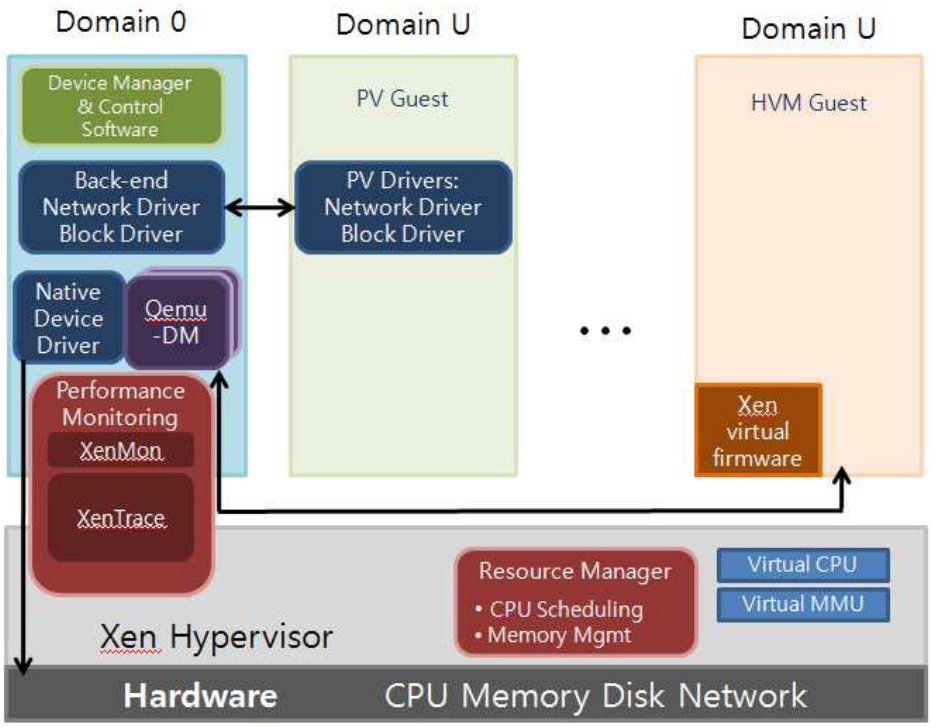
<그림 2>는 Xen 3.2 architecture overview 문서[8]를 바탕으로 그린 그림이다. Xen 가상화 시스템은 다음과 같은 구성 요소들이 함께 작동한다[8].

- Xen hypervisor: guest operating system 아래 hardware 위에서 동작하는 software로 hardware를 가상화 해준다. 여러 가상 머신을 위해 CPU 스케줄링과 메모리 partitioning을 해주어 여러 가상 머신들이 물리적 서버의 자원을 공유하지만 서로 격리되어 실행될 수 있게 만들어 주는 역할을 한다.

- Domain 0: 수정된 Linux kernel로 Xen hypervisor에서 실행되는 특별한 가상 머신으로 물리적 I/O 자원을 직접 접근할 수 있고 다른 가상 머신들과 interact하는 특별한

권을 가진다. Domain 0는 다른 가상 머신의 I/O request를 처리 하는 역할을 하므로 어떤 다른 가상 머신 보다 먼저 실행되어야 한다.

다른 가상 머신으로부터의 네트워크와 disk request를 처리하기 위해 두 driver, Network Backend Driver와 Block Backend Driver를 포함한다. Network Backend Driver는 local networking hardware로 직접 통신하여 request들을 처리하고, Block Backend Driver는 local storage disk에 데이터를 읽고 쓰는 request들을 처리한다.



<그림 2> Xen hypervisor(version 3.2)의 architecture

- Domain Management and Control: 이 서비스는 Domain 0에서 실행되는 daemon으로 가상 시스템을 전반적으로 관리하고 제어하는 모듈이다.

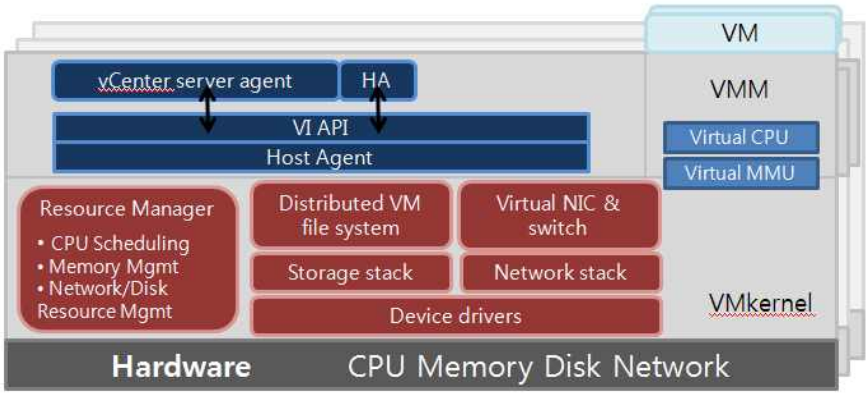
- Domain U (Dom U) PV Guest: Paravirtualized 된 가상 머신을 PV guest라 한다. Paravirtualized 된 가상 머신은 수정된 operating system을 가지고 있는데 이 수정된 operating system은 하이퍼바이저 위에서 실행되는 것을 알고 하이퍼바이저에 system call을 요청한다. 또한 물리적 hardware에 직접 접근하는 권한이 없으며 다른 가상 머신과 함께 자원을 공유하며 실행되고 있는 것을 알고 있다. Guest operating system을

수정함으로써 가상화에 의한 성능의 저하를 없애는 것이 용이해 진다.
PV guest domain은 network과 disk 접근을 위해 두 driver, PV Network Driver와 PV Block driver를 가지고 있다. I/O 접근 필요시 Domain 0에게 요청한다.

- Domain U (Dom U) HVM Guest: fully virtualized 된 가상 머신을 HVM guest라 하는데 가상화를 지원하는 hardware가 반드시 필요하다. HVM guest는 가상 머신이라는 것을 인식하지 못하며 다른 가상 머신과의 공유를 모른 상태로 실행된다. HVM guest를 수정된 operating system을 필요로 하지 않기 때문에 Windows와 같은 source가 공개 되지 않은 operating system을 실행 하는 것이 가능해 진다. HVM guest는 Qemu-dm이라는 special daemon이 필요한데, 이 daemon은 HVM guest의 network와 disk request를 지원해 준다. Xen virtual firmware는 수정되지 않은 operating system이 예사하는 BIOS를 simulate 해준다.

Xenmon과 Xentrace는 각 VM에 대한 성능 모니터링을 위한 툴이다. 이를 이용하여 각 가상 머신 별 CPU 사용량, waiting time, blocking time 등을 측정하는 것이 가능하다.

VMware ESX hypvervior



<그림 3> VMware ESX hypervisor의 architecture

<그림 3>는 VMware ESX에 관한 문서[11]를 바탕으로 그린 그림이다. VMware ESX 가상화 시스템은 기본적으로 다음과 같은 구성 요소들이 함께 작동한다[11].

- Virtual Machine Monitor (VMM) and VMkernel: 이들은 CPU, memory, disk, network 등의 physical hardware를 가상화 한다. 각 가상 머신은 같은 서버에 있는 다른 머신들과는 격리된 virtual hardware device들을 가지게 된다.

- Resource Manager: 서버의 physical 자원을 분할하여, 각 가상 머신에 CPU, network, disk, network 자원을 할당한다. CPU scheduling, memory management, disk/network device resource management를 포함한다.

- Hardware interface component: hardware specific한 서비스를 제공 해준다. Virtual Machine File System(VMFS)를 포함하는데 VMFS은 ESX hypervisor에 알맞은 VM을 위해 최적화된 파일 시스템이다. VMkernel은 물리적 서버의 network와 disk request 지원을 위한 물리적 device 접근을 하는 device driver를 포함한다. ESX hypervior 설치 시 물리적 device를 발견하여 적절한 driver를 설치한다.
  Xen hypervisor와는 달리 ESX hypervisor는 device driver를 직접 포함하여, Xen hypervisor 보다는 "두꺼운" 구조를 가지고 있다.

ESX hypervisor는 가상 머신 관리 서버(vCenter Server[1])의 관리와 제어를 받는데 이를 위해 vCenter server agent를 포함하게 된다. 이 agent는 vCenter Server에서 request를 받아 처리하고 reply 하는 역할을 한다. vCenter Server의 관리를 받지 않을 경우에서는 자동으로 uninstall 되는 agent이다. ESX hypervisor에 직접 연결하여 관리 제어하는 것도 가능 한데 이 경우를 위해서 VI (virtual infrastructure) API를 제공한다. HA component는 뒤의 3장 3절에서 설명할 High Availability 기능을 위한 agent이다.

제 3 절 가상머신 scheduling

한 물리적 서버에 여러 개의 가상 머신이 동시에 실행 되게 되는데 VMM은 가상 머신들을 효율적으로 scheduling 해야 한다. 기본적으로 (각 VM에 우선순위에 대한 설정이 없는 경우) 각 VM에게 fairness와 성능 isolation을 제공하여야 한다. 따라서 같은 서버에서 함께 실행되고 있는 다른 가상 머신에 의해 성능 영향을 받지 않아야 하고, 각 VM에게 공평하게 자원이 할당되어야 한다. Xen과 VMware의 VMM이 가상 머신에게 동적으로 자원을 할당하고 scheduling 하는 방식은 다음과 같다.

Xen hypervisor의 scheduler

Xen은 credit-based CPU scheduler를 가지고 있다. 이 scheduler는 CPU 자원을 동적으로 가상 머신들에게 할당하기 위해 proportional-share based algorithm을 사용한다. 사용자는 CPU 자원에 대하여 각 가상 머신에 대해 2가지 parameter를 결정 할 수 있다.

1) weight: 가상 머신의 상대적인 중요함이다. 1부터 65535까지의 값을 선택하는데 256이 기본 값이다.
2) Cap: 가상 머신에 대해 할당 할 수 있는 자원양의 upper bound 이다. 기본 값은 0인데 이는 제한이 없다는 의미이다.

Multicore의 VM을 지원 하는데 여러 개의 virtual CPU를 스케줄링 하는데 있어서 work conserving, 즉 주어진 계산 작업이 끝날 때 까지 물리적 core를 idle 하게 하지 않는다. 또, core들 사이에 load balancing을 지원한다.

Global accounting mechanism이 있어 서버에서 함께 실행되는 VM들의 weight과 cap을 고려하여 각 가상 CPU 별 credit을 계산한다. 주기적으로 가상 CPU가 얼마만큼의 credit을 얻고 사용했는가를 계산하는데 이에 따라 두 가지 종류의 우선순위를 결정한다. Credit을 다 사용하여 negative인 경우는 "over“의 priority, credit이 아직 남은 경우는 "under" priority를 준다.

각 물리적 core 또는 CPU는 local queue를 유지하는데 이는 “under"와 "over" 순으로 정렬된 FIFO queue 이다. 각 가상 CPU는 물리적 CPU에 의해 선택될 시 30ms 동안 실행이 가능하다. 각 물리적 CPU가 스케줄링 선택을 하는 시점에서 다음의 순으로 가상 CPU를 선택한다.

1) local VCPU with under fair share
2) remote VCPU with under fait share
3) local VCPU with over fair share
4) remote VCPU with over fair share

VMware ESX hypervisor의 scheduler

한 가상화된 물리적 서버의 CPU와 memory 자원을 그 위에서 실행되고 있는 가상 머신들에게 동적으로 분배하는데, ESX는 proportional-share based algorithm을 사용한다.

사용자는 CPU와 memory 자원에 대하여 각 가상 머신에 대해 3가지 parameter를 결정 할 수 있다. (이 parameter에 설정이 없으며 기본적으로 서버 자원을 공평하게 분배 하게 된다.)

1) share: 가상 머신의 상대적인 중요함이다.
2) Reservation: 가상 머신에 대해 보장되는 최소 할당량이다.
3) Limit: 가상 머신에 대해 할당 할 수 있는 자원양의 upper bound 이다.

위의 share는 Xen의 weight과 같은 역할을 한다. 예를 들어 한 서버에 3개의 가상 머신이 실행되고 있고 share의 비율이 2:1:1 이라며 첫 번째 가상머신에 1/2에 해당하는 자원이 할당되고, 나머지 두 가상 머신에 각각 1/4에 해당하는 자원이 할당 되는 것이 가능하다. 만약 첫 번째 가상 머신이 1/2에 해당하는 모든 자원을 사용하지 않는 경우 그 자원은 나머지 두 가상 머신에 의해 사용 될 수 있다. 따라서 share는 자원에 대한 경쟁이 있을 경우에만 그 역할을 하게 된다.

Reservation은 보장되는 최소 할당량으로 서버에 작업양이 많은 경우라도 반드시 할당해 주어야 하는 자원 양이다. 따라, 서버에 새로운 가상 머신을 power-on 하고자 할 때 CPU와 memory 자원에 대하여 서버에 reserve 되지 않은 자원의 양이 이 가상 머신의 reservation 보다 많은가는 확인하고, 많은 경우에만 가상 머신을 power-on 할 수 있다. 결국 reservation value를 이용하여 admission control을 한다.

Limit은 가상 머신이 사용할 수 있는 최대 자원 사용량으로 물리적 서버의 최대 자원 사용량에 의해 제한이 된다. 설정 할 수 있는 최대값은 물리적 서버의 자원량이지만 이보다 작게 설정 되는 것이 가능하다. 이 limit value를 이용하여 중요도가 낮은 application이 자원을 많이 사용하는 것을 제한하여 중요도가 높은 application에 나쁜 영향을 주지 않도록 한다.

VMware ESX hypervisor의 CPU scheduling 방법은 다음과 같다. 가상 머신의 각 virtual CPU에 CPU 자원의 배당 몫을 계산한다. 이 배당 몫은 사용자가 명시한 share, reservation, limit으로부터 계산한다. CPU scheduling 결정을 내릴 때, 각 virtual CPU의 priority를 계산하는데 한 period 동안 이미 사용한 CPU 자원과 period 동안 주어진 배당 몫의 비율이 priority가 된다. 각 가상 CPU의 priority를 계산 한 후 가장 priority가 높은 것부터 스케줄 하여야 한다(위의 계산한 비율이 작은 것이 priority가 가장 높음).

ESX에서는 한 가상 머신의 여러 virtual CPU에 대해여 기본적으로 co-scheduling을 하는데, 고성능을 위하여 virtual CPU의 set을 동시에 실행시킬 수 있도록 스케줄 하는 것이다. 이로써 synchronization에 의한 latency를 줄일 수 있고 한 가상 머신의 virtual CPU

에 대한 synchronous 한 progress를 제공 할 수 있게 된다.

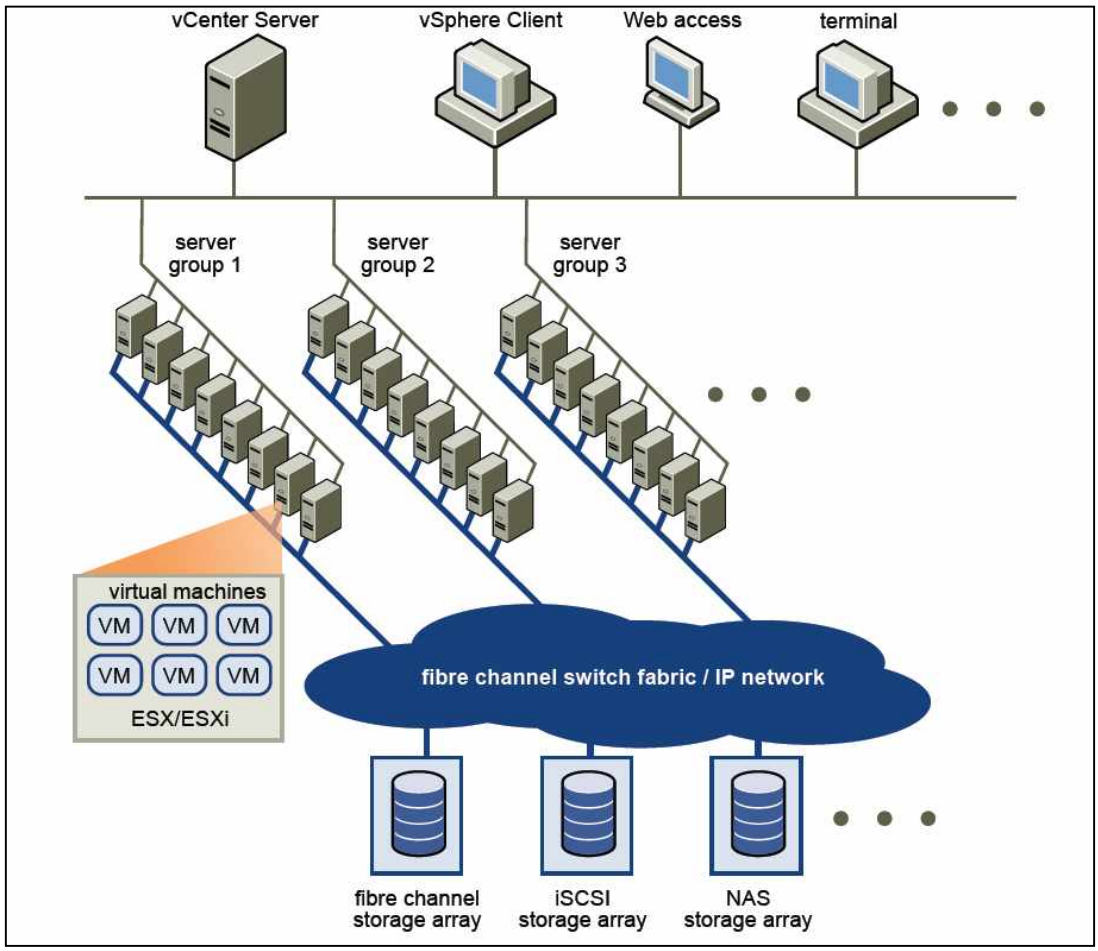
VMware의 ESX 또한 물리적 core들 사이에 load balancing을 하는데 다음과 같은 요소를 고려한다[12].

- CPU topology
- Multicore/Last Level Cache(LLC)
- 비대칭형 메모리(NUMA)
- Hyperthreading

또한 물리적 core 사이에 가상 CPU를 migration 하는데 있어서 migration cost를 고려하고, migration throttling을 위해 너무 자주 core간에 migration이 일어나 주어진 계산하는 것보다 이주하는 비용이 더 커지지 않도록 하는 기법을 사용한다.

제 3 장 가상화 시스템 관리 기술 분석

가상화 시스템 또는 가상 infrastructure에서는 여러 물리적 머신의 자원이 가상화되고 통합되어 가상화된 계산, 네트워크, 저장 자원이 하나의 자원 pool을 제공된다. 사용자는 자원 pool안의 자원들을 유연성 있게 사용하여 비용을 절감하고, 또한 시스템 유지 보수를 쉽게 할 수 있다.



<그림 4> 가상 데이터센터의 물리적 topology

그림 출처: VMware vSphere Introduction[1]

이런 가상화 시스템은 기업의 가상 datacenter나 클라우드 컴퓨팅 시스템에서 사용 되는데 주로 물리적 서버, 기업 형 네트워크 및 스토리지로 구성 되어 있다. <그림 4>는 가

상 데이터 센터의 물리적 topology의 한 예를 보여준다. Hypervisor가 설치된 가상화된 서버로 구성된 여러 서버 그룹, SAN 또는 NAS array의 스토리지, 서버 및 스토리지 등 전체 데이터센터를 연결하는 네트워크로 구성된다.

이런 물리적 topology는 가상 infrastructure에서는 가상화되어 논리적인 자원으로 취급되는데 가상 infrastructure은 다음의 요소들로 구성 된다.

- 계산 자원: 물리적 서버가 제공하는 CPU/memory 자원, 또는 여러 물리적 서버로 구성된 cluster가 제공하는 CPU/memory 자원
- 스토리지 자원: SAN 또는 NAS array가 제공하는 저장 자원
- 네트워크 자원: 네트워크가 제공하는 자원

가상 infrastructure 안에서는 하나의 물리적 서버도 CPU와 memory 자원을 제공하지만, 여러 물리적 서버의 자원을 통합한 cluster도 자원을 제공하는 하나의 개체로 다루는 것이 가능하다. 가상화 시스템의 관리를 쉽게 하고, business goal의 효율적 달성, 또는 시스템 사용 정책을 지원 하기위해 subset의 물리적 서버를 모아 cluster로 만들어 관리한다. 이 cluster는 가상 머신들로 구성된 가상 cluster와는 달리 물리적 서버로 구성된 cluster를 말한다. 이 (물리적) cluster의 자원들은 하나로 통합된 계산 및 메모리 자원을 제공하게 된다. 한 cluster가 제공하는 자원 양은 이에 속하는 물리적 서버 자원 양의 합이다. 이런 cluster는 가상 infrastructure에서 자원을 논리적으로 나누고 또한 hierarchy를 주는 것이 가능해져 자원을 더 효율적으로 사용하고 사용하기 쉽게 해줄 수 있다.

다음 <표 1>은 데이터센터와 서버의 설정을 보여준다. 작은 규모의 데이터센터에서도 한 rack에 배치 될 VM의 수를 1000개 이상으로 예상하고 있다. 이런 대규모의 시스템을 효율적으로 구축, 관리, 제어하는 것은 매우 중요하지만 쉽지 않은 일이다.

Configuration	Number of hosts in rack	Cores in rack	VMs in rack
Small	10	320	1280
Medium	20	640	2560
Large	30	960	3840
Future	40	1280	5120

<표 1> 데이터센터와 서버 설정

표 출처: Soundararajan과 Anderson의 ISCA 2010 논문[2]

이 장에서는 가상 infrastructure 관리를 위한 VM migration, 자원 관리, high availability와 fault tolerance를 지원하는 장애 관리 기술 등을 분석한다.

제 1 절 VM Migration

VM migration은 VM을 다른 물리적 서버로 이주하는 기능이다[3 ,4]. 이주하려는 VM의 상태에 따라 live (or hot) migration과 cold migration으로 구분 한다. Live migration 경우 현재 power-on 상태로 실행되고 있는 VM을 downtime 없이 다른 물리적 서버로 이주한다. 반면에 cold migration 경우 현재 power-off 상태인 VM을 다른 물리적 서버로 이주하여, 이후 power-on 될 경우 이주된 서버에서 실행하게 된다. 이런 migration 경우에는 source와 destination 서버가 공유 storage가 있어 이주 하려는 VM 파일을 두 서버가 모두 접근 할 수 있어야 한다.

VM migration을 위해서는 compatibility를 확인하여 destination 서버에서도 source 서버에서와 같은 instruction을 이용하여 VM이 실행 가능한지를 확인하여야 한다. Processor는 같은 vendor class에 속해야 하며 (Intel vs. AMD), compatible한 같은 processor family에 속해야 한다. Processor family는 processor vendor에 의해 정의 되는데, 같은 family의 다른 processor version은 processor 모델, extended feature등을 비교 하여 구분하는 것이 가능하다.

VM을 다른 물리적 서버로 이주시키기 위해서는 3가지 state 정보를 destination 서버에 보내야 destination 서버에서 VM 실행을 계속하는 것이 가능하다[4].

1) CPU, motherboard, networking과 storage adapter 등을 포함하는 가상 device state 정보
2) networking, USB, SCSI storage device 등과의 external connection 정보
3) VM의 물리적 메모리

실제 migration process는 다음과 같은 여러 step으로 구성 된다[4].

1) 이주 할 VM과 destination 서버를 선택하여 migration process를 시작한다.
2) 선택된 VM이 source 서버에서 계속 실행되는 동안, VM의 메모리 state을 pre-copy 한다.
3) VM을 정지시키고, non-memory state을 destination 서버에 보낸다.

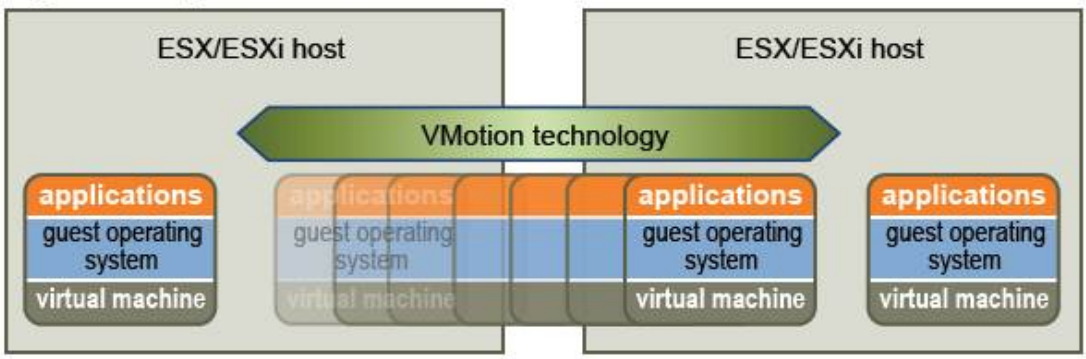
4) VM의 control을 destination으로 이임하고, VM을 destination server에서 다시 시작한다.

5) 남아 있는 메모리 state이 있다면 보내고, source 서버에서의 VM에 대해 남아 있는 정보는 제거한다.

VM이 source 서버에서 계속 실행되는 동안 위의 2번째 step의 과정은 iterative 하게 진행된다. 첫 번째 pre-copy 경우에는 전체 물리적 메모리가 모두 destination 서버에 copy 된다. 각 메모리 page가 copy 되기 전에 그 page를 read-only로 표시한다. 따라 이 page에 대한 수정은 VMM에 의해 발견되고 VMM은 수정된 page를 별도로 보관한다. 첫 번째 pre-copy가 끝난 후, 그 사이에 수정된 페이지들을 다시 destination 서버에 copy 한다. 이 과정을 수정된 페이지의 수가 threshold 보다 작아지거나 진전이 없는 경우까지 계속한다.

VM 파일들을 다른 storage로 이주시키는 storage migration도 가능하다. 실행되고 있는 VM 파일을 다른 storage로 downtime 없이 이주하는 것인데, 위에 설명한 VM migration과 함께 사용하면 한 VM이 실행되고 있는 물리적 서버와 storage에서 다른 서버와 storage로 이주 시킬 수 있다.

Figure 5. Migration with VMotion



<그림 5> 가상 머신의 migration

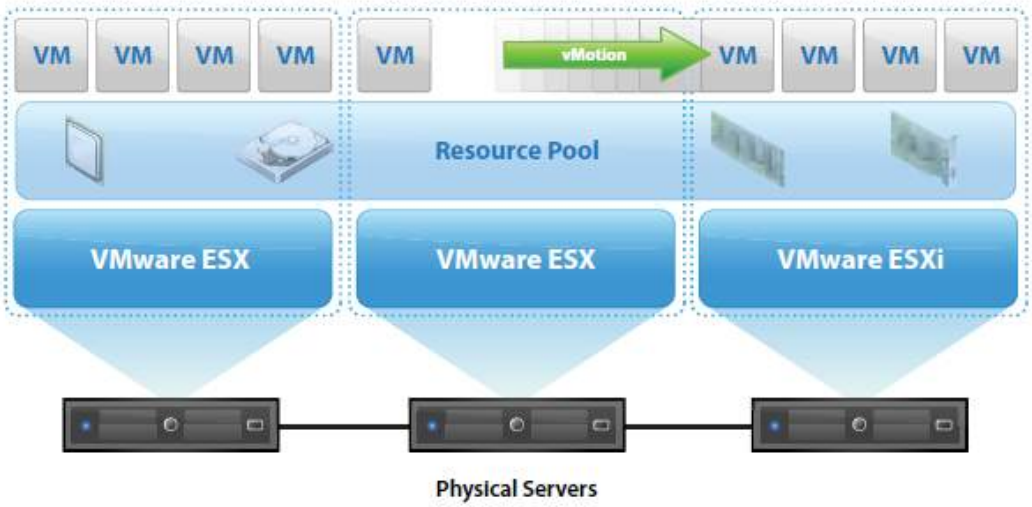
그림 출처: VMware vSphere Introduction[1]

제 2 절 가상화 시스템 자원 관리

가상화 시스템에서는 여러 물리적 서버의 자원들이 통합되어 resource pool이 된다.

가상화 시스템의 resource pool의 자원을 효율적으로 관리하기 위해서는 동적으로 필요한 자원을 VM들에게 할당하고 물리적 서버들 간의 load balancing을 해주는 것이 필요하다. 또한 가상화 시스템의 서버 사용량을 바탕으로 필요한 물리적 서버만을 계속 유지하고 나머지 서버들을 power-off 함으로 가상화 시스템의 에너지 소비량을 줄이는 것이 가능하다.

VMware의 virtual infrastructure 관리 시스템에서는 이를 위해 두 가지 기능(Distributed Resource Scheduling과 Distributed Power Management)을 제공한다[5].



<그림 6> Distributed Resource Scheduling

그림 출처: VMware Distributed Resource Scheduling product datasheet [5]

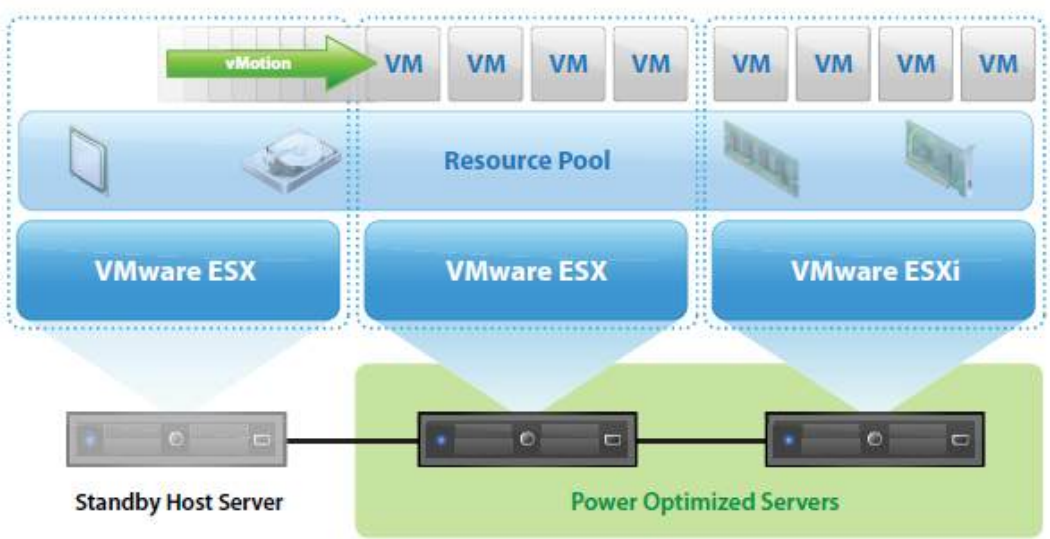
Distributed Resource Scheduling(DRS)

DRS 기능은 여러 물리적 서버로 구성된 cluster에 적용될 수 있다. Cluster의 통합된 하드웨어 자원은 resouce pool이 된다. VM을 DRS cluster에 power-on 할 경우 DRS는 admission control을 행하여 VM을 실행 시킬 충분한 자원이 있는가를 확인하고 initial placement, 즉 배치 될 서버를 결정한다. 지속적으로 cluster의 각 서버들의 사용량을 모니터링하여 자원을 동적으로 VM에게 할당하고 서버들 간의 load balancing을 해준다. 한 VM의 load가 증가되면 VM들을 cluster안에서 redistribute하는 것을 통해 그 VM에게 더 많은 자원을 제공한다. 이때 실행되고 있는 VM을 다른 서버로 이주시키기 위해 VM migration 기능을 사용한다.

자원 할당과 load balancing의 결정은 사용자에 의해 미리 정의된 rule 과 DRS migration threshold을 이용한다. 사용자는 affinity rule과 anti-affinity rule을 정의 할 수 있는데 affinity rule은 두 개 이상의 VM이 같은 물리적 서버에 배치되는 것을 선호 한다는 것을 명시한다. 반대로 anti-affinity rule 은 두 VM이 같은 물리적 서버에 배치되어 같이 실행되어서는 안 된다는 것을 명시한다. Migration threshold는 어느 정도의 서버간의 load imbalance를 받아들일 수 있는가를 나타내는데 conservative한 경우에는 반드시 필요한 load balancing을 위한 재배치를 하고, aggressive 한 경우에는 서버 간의 더 많은 재배치를 하게 된다.

VM 이주 시 compatibility 확인에 대해 언급하였는데 DRS의 기능을 충분히 활용하기 위해서는 DRS cluster 내의 서버 사이의 VM migration이 가능 하도록 잘 구성 하는 것이 중요하다.

Distributed Power Management (DPM)



<그림 7> Distributed Power Management

그림 출처: VMware Distributed Resource Scheduling product datasheet [5]

DRS cluster의 서버들의 자원 사용량을 계속적으로 모니터링하여 현재 power-on되어 실행되고 있는 VM들을 위한 자원 필요량이 줄어든 경우, DPM은 에너지 소비량을 줄이기 위해 VM들을 consolidate 하여 불필요한 서버를 stand by mode로 power down 한다. 이

후 자원 필요량이 증가되는 경우 power down한 서버들을 다시 power on 하고 VM들을 재배치한다. VM들을 consolidate 하고 또는 재배치하기 위해서 VM migration 기능을 이용한다.

사용자는 DPM 기능에 대해 threshold를 선택할 수 있는데 이는 어느 정도의 over 또는 under utilization의 상태를 원하는가와 power 상태를 바꾸는 것으로 얼마만큼의 성능 증가를 기대하는 가를 표현하는 것이다. 이 선택된 threshold에 따라 자동적으로 가장 필요하고 성능의 많은 향상이 기대되는 경우만 power 상태를 바꾸어 가상 머신들을 재배치하기도 하고 또는 작은 향상이 기대되는 경우에도 power 상태를 바꾸어 재배치하기도 한다. 자주 서버의 power 상태를 바꾸는 경우에는 이에 따른 overhead가 많이 발생하기 때문에 시스템 사용 목적에 따라 신중하게 선택할 필요가 있다.

제 3 절 장애 관리

가상화 시스템에서 서버 failure, VM failure 등의 결함 또는 장애가 발생하였을 경우 복구하는 기능이 반드시 필요하다. 기업의 가상 데이터 센터나 클라우드 컴퓨팅 시스템과 같은 가상화 시스템은 많은 수의 가상화된 서버로 구성 되는 경우가 많은데 시스템 관리자가 이런 결함이나 장애를 수동으로 감시하고 복구하는 것은 거의 불가능한 일이다. 따라, 결함/장애 발생 시 자동으로 복구가 가능해야 한다.

안정적 시스템을 위한 기술로 high availability와 fault tolerance가 있다. High availability는 uptime의 percentage로 표현 할 수 있다. Fault tolerance는 결함 발생 시에도 지속적으로 운영이 가능하게 하는 시스템 기능이다. 가상화 시스템에서도 high availability와 fault tolerance를 제공하는 것이 필요하다. 결함 또는 장애 관리 기능들은 virtual infrastructure를 관리하는 management 서버에 장애가 발생하여 기능을 수행 할 수 없게 될 경우라도 물리적 서버나 가상 머신 failure를 발견하여 복구 할 수 있어야 한다.

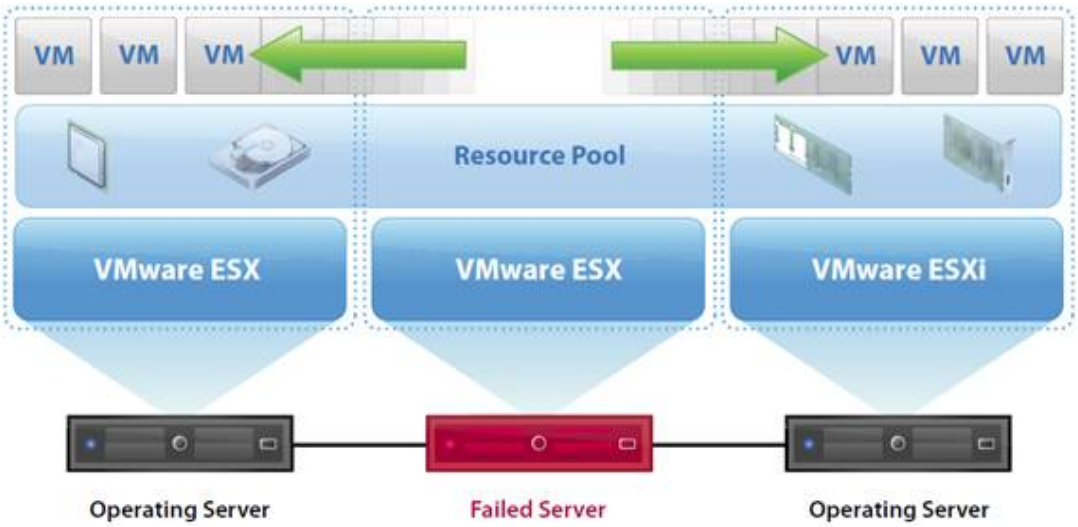
VMware의 가상화 시스템 관리 solution에 서는 high availability와 fault tolerance를 다음과 같은 방식으로 제공하고 있다.

High availability

VM에서 실행되고 있는 application에 대해 high availability(HA)를 제공하기 위해서 HA를 위해 구성된 cluster안의 서버들의 health 상태를 모니터링하고, 물리적 서버 failure

가 발생하여 기능을 수행 할 수 없게 되면, 그 서버에 있던 가상 머신들을 짧은 시간 안에 (분단위) 같은 cluster안의 남는 자원이 있는 다른 서버에서 재시작 한다. 또한 한 서버안의 가상머신의 health 상태를 모니터링 하여 가상 머신에서 실행되고 있는 guest operation system에 failure가 발생한 경우 그 가상 머신을 재시작 한다.

HA cluster 안의 서버들은 서로 heartbeat을 주기적으로 주고받는다. 한 서버로 부터의 heartbeat이 미리 정의된 시간동안 도달 하지 않을 경우, 그 서버에 결함이나 장애가 생긴 것으로 생각하고 그 서버에서 실행되고 있던 가상 머신들을 다른 서버들에서 자동적으로 재시작 한다. 위에서 설명한 DRS 기능과 같이 사용되면 DRS는 가상 머신들에 대한 최적 배치를 결정 한다. 이런 방식으로 application에 대한 high availability를 제공하기 위해서는 cluster 안의 서버들을 storage를 공유하고 있어야 가능하다.



<그림 8> High Availability

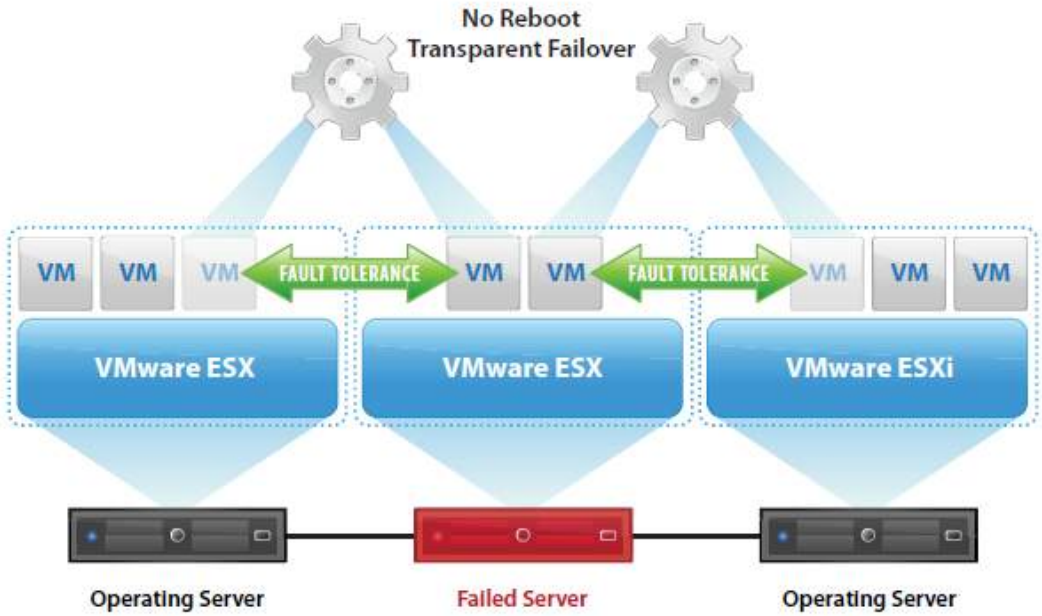
그림 출처: VMware High Availability product datasheet [6]

Fault Tolerance

개별 가상 머신에 대해 fault tolerance 기능을 적용하는 것이 가능한데, fault tolerance가 적용된 가상 머신에 대해 다른 물리적 서버에 별도의 live shadow instance를 유지하는 것이다. 해당 가상 머신에 결함 장애 등의 발생 하였을 경우, shadow instance가 바로 active 하게 상태를 변화 시키고 실행 한다. 즉 transparent 한 failover를 제공하게 된다.

또한, shadow instance가 active 하게 실행을 시작 할 때 바로 이 가상 머신을 위한 shadow instance를 다른 물리적 서버에 생성한다.

현재 실행되고 있는 VM을 primary VM이라 하고 shadow instance를 secondary VM 이라 한다. Secondary VM에 대해서 사용자는 primary VM에서 행할 수 있는 여러 VM operation (power-on, power-off, suspend 등)을 행할 수 없다. 이 두 VM은 같은 disk에 접근이 가능하여야 하며, 같은 IP address, MAC address를 가지는 single entity처럼 관리 되어야 한다.



<그림 9> Fault Tolerance

그림 출처: VMware Fault Tolerance product datasheet [7]

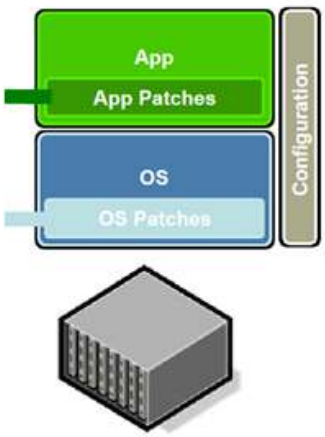
이 두 VM은 정확하게 같은 input과 같은 event들을 가져야 하는데 이를 위해 VM record and replay 기술을 이용한다. 가상 머신 record and replay 기술은 한 가상 머신의 전체 instruction sequence를 아주 적은 overhead로 reconstruct 하는 기술이다[21].

다른 물리적 서버에서 VM의 replay를 지원하기 위해서는 가상 머신 memory, register, disk의 state에 포함 되지 않는 CPU로의 모든 input을 record 해야 한다. External device와 asynchronous event까지 포함 하는 모든 input을 record 하여 second VM에서 모든 instruction을 재생성 하는 것이다.

제 4 절 Virtual appliance

Virtual appliance는 software appliance의 subset으로 가상화 시스템에서 실행될 수 있는 가상 머신 image 이다. 가상 머신에 software appliance를 설치함으로써 virtual appliance를 생성 할 수 있다.

물리적 서버 provisioning을 위해서는 <그림 10>와 같이 물리적 서버에 operating system 설치, 필요한 application software stack 설치, configuration, operating system patch와 application patch 설치의 과정을 거치게 된다.

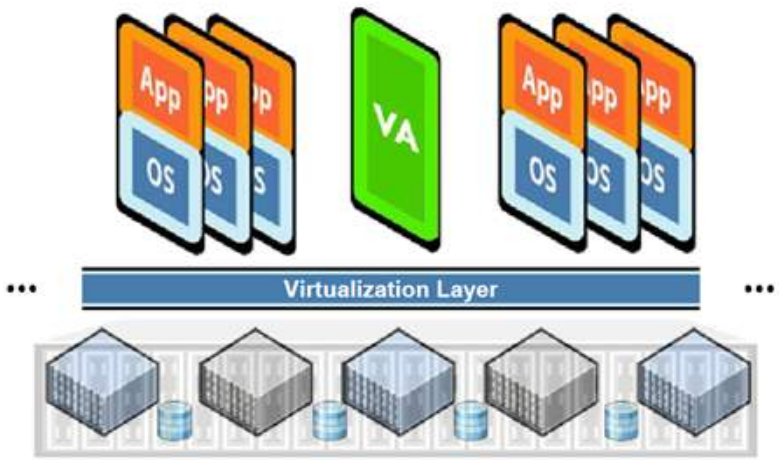


<그림 10> 물리적 서버의 provisioning
그림 출처: Rosenblum의 프리젠테이션 자료[9]

Virtual appliance를 이용하여 각 가상 머신에 따로 여러 software를 설치하는 수고를 없앨 수 있을 뿐 아니라, 가상 머신의 operating system 까지도 별도의 설치를 필요로 하지 않게 된다. 따라, 물리적 머신을 구매하여 operating system을 설치하고 필요한 software를 설치한 후 patch로 update 하여 사용자에게 제공하는 provisioning 과정을 <그림 11>과 같이 virtual appliance를 이용하여 가상화 시스템에 바로 provisioning 할 수 있게 된다.

더구나 VMware에서 제공하는 solution처럼 virtual appliance는 여러 VM이나 appliance를 포함 할 수 있다. 이런 여러 VM으로 구성된 virtual appliance는 하나의 VM처럼 기본 operation을 수행 하는 것이 가능하다. 즉, virtual appliance을 구성하는 여러 VM을 한번에 power-on, power-off 할 수 있으며 한번에 clone 해 낼 수 있다. 이 기능을 이용하면 multi-tier application을 package로 사용하고 관리 할 수 있다. 또한, 여러 가상

머신으로 구성된 가상 클러스터를 쉽게 생성하고 사용 및 관리 할 수 있어 시스템을 운영하는 비용을 절감하고 사용자의 요구에 따른 시스템을 제공할 수 있게 된다.



<그림 11> Virtual appliance의 provisoning
그림 출처: Rosenblum의 프리젠테이션 자료[9]

참고문헌

[1] Introduction to VMware vSphere,
http://www.vmware.com/pdf/vsphere4/r40_u1/vsp_40_u1_intro_vs.pdf.

[2] Vijayaraghavan Soundararajan, and Jennifer M. Anderson, "The impact of management operations on the virtualized datacenter", In *Proceedings of the 37th annual international symposium on Computer architecture*, 2010.

[3] Christopher Clark, Keir Fraser, Steven Hand, Jakob Gorm Hansen, Eric Jul, Christian Limpach, Ian Pratt, and Andrew Warfield, "Live Migration of Virtual Machines", In *Proceedings of the annual conference on USENIX '06 Annual Technical Conference*, 2006.

[4] Michael Nelson, Beng-Hong Lim, and Greg Hutchins, "Fast Transparent Migration for Virtual Machines", In *Proceedings of the annual conference on USENIX '05 Annual Technical Conference*, 2005.

[5] VMware Distributed Resource Scheduler(DRS) Product Datasheet,
http://www.vmware.com/files/pdf/VMware-Distributed-Resource-Scheduler-DRS-DS-EN.pdf.

[6] VMware High Availability Product Datasheet,
http://www.vmware.com/files/pdf/VMware-High-Availability-DS-EN.pdf.

[7] VMware Fault Tolerance Product Datasheet,
http://www.vmware.com/files/pdf/VMware-Fault-Tolerance-FT-DS-EN.pdf.

[8] Xen Architecture Overview,
http://wiki.xensource.com/xenwiki/XenArchitecture?action=AttachFile&do=get&target=Xen+Architecture_Q1+2008.pdf.

[9] Mendel Rosenblum, Distinguished Lecture - The Impact of Virtualization on Modern Computing Environments,
http://www.cs.vt.edu/DistinguishedLectures/MendelRosenblum.

[10] Microsoft System Center Virtual Machine Manager, http://www.microsoft.com/systemcenter/en/us/virtual-machine-manager.aspx.

[11] VMware ESX Server 2 Architecuter and Performance Implications, VMware white paper, https://www.vmware.com/pdf/esx2_performance_implications.pdf.

[12] VMware vSphere 4: The CPU Scheduler in VMware ESX 4, VMware white paper, http://www.vmware.com/files/pdf/perf-vsphere-cpu_scheduler.pdf.

[13] vSphere Resource Management Guide, http://www.vmware.com/pdf/vsphere4/r40_u1/vsp_40_u1_resource_mgmt.pdf.

[14] Amazon EC2, http://aws.amazon.com/ec2/.

[15] 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. In *Proceedings of the nineteenth ACM symposium on Operating Systems Principles*, 2003.

[16] VMware ESX 4.0, http://www.vmware.com.

[17] Microsoft Hyper-V Server, http://www.microsoft.com/hyper-v-server/en/us/default.aspx.

[18] Eucalyptus Systems, http://www.eucalyptus.com/.

[19] OpenNebula, http://www.opennebula.org/.

[20] VMware vCloud Express, http://www.vmware.com/solutions/cloud-computing/public-cloud/vcloud-express.html.

[21] Jim Chow, Tal Garfinkel, and Per M. Chen, "Decoupling dynamic program analysis from execution in virtual environments", In *Proceedings of the annual conference on USENIX '08 Annual Technical Conference*, 2008.