Daum Tistory 로그인

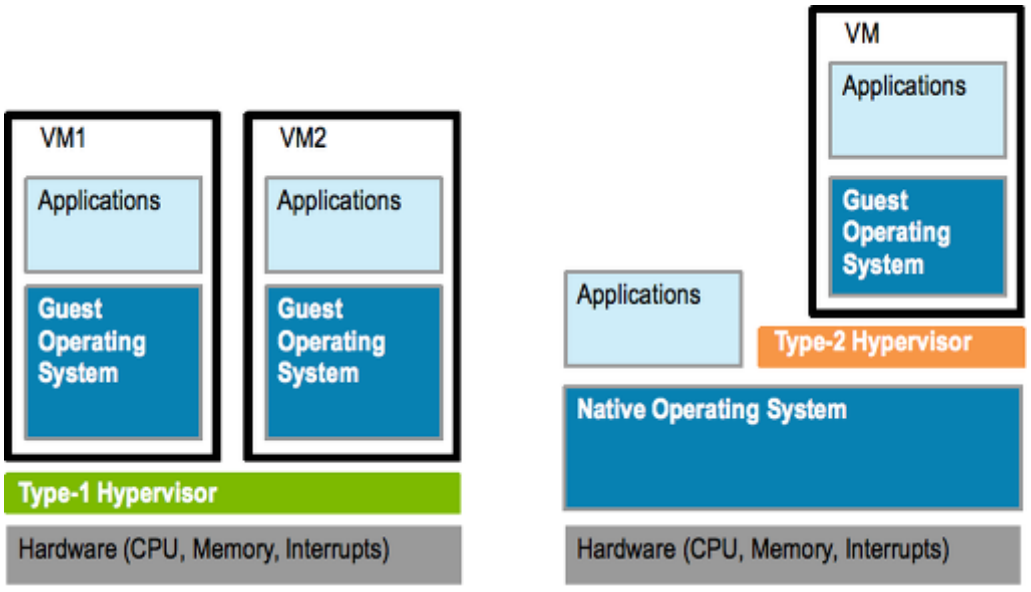
한걸음씩

Xen과 KVM

가상화 2013.01.24 04:57

핫이슈를 넘어서 이제는 당연시 되고 있는 클라우드 시스템(Cloud System). 클라우드를 구성하는 요소 중에서 가상화(Virtualization) 기능은 빠질 수 없는 핵심 요소 중에 하나라고 할 수 있다. 가상화를 하기 위해서는 하이퍼바이저(Hypervisor, Virtual Machine Monitor이라고도 한다)라는 별도의 시스템 프로그램이 필요하다. 이미 서버단의 가상화는 자원의 효율성 측면에서 기본이 되어가고 있다.

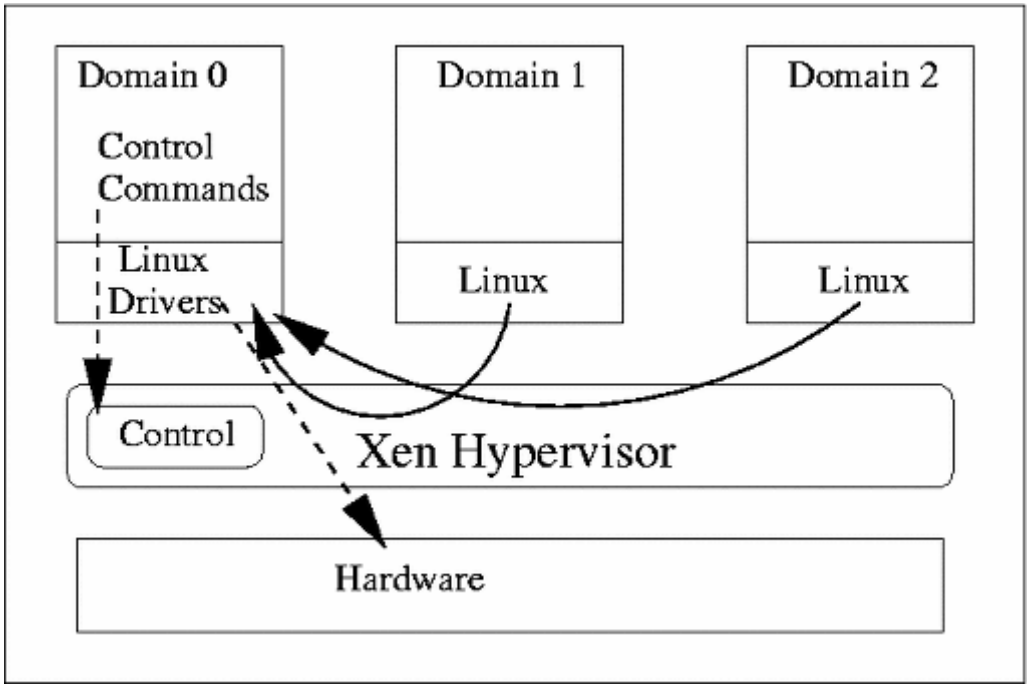
가상화 분야에서 '가상화 대상이 되는 시스템'을 의미하는 용어는 여러 용어가 혼재되어 사용되니 정리하고 넘어가자. '가상머신(Virtual Machine)'-약어로 'VM', '도메인(Domain)'-약어로 'Dom', '게스트(Guest)' 이들 용어 모두가 '가상화 대상이 되는 시스템'을 의미한다. '게스트 운영체제'와 대비되는 '호스트 운영체제'는 가상화 프로그램을 실행하는-즉, 가상화 하지 않은- 실제 운영체제를 의미한다. 호스트 운영체제는 Type 2에서만 존재하는데, Xen은 Type 1 하이퍼바이저이므로 호스트 운영체제가 존재하지 않는다. Type 1은 하이퍼바이저 위에 모든 도메인이 동작하는 방식이고, Type 2는 호스트 운영체제에서 가상화 프로그램을 실행하여 게스트를 실행하는 방식을 말한다.



<그림 1. Type 1과 Type 2 하이퍼바이저>

대표적인 오픈소스 하이퍼바이저로는 Xen과 KVM이 있다. Xen은 'Xen and the Art of Virtuallization'이라는 가상화 분야에서 매우 유명한 논문과 함께 대표적인 반 가상화(Para Virtualization) 하이퍼바이저로 등장하였다. 반 가상화는 게스트 운영체제 일부를 수정하는 시스템을 말한다. 반 가상화의 대비되는 전 가상화(Full Virtualization)은 게스트 운영체제를 수정하지 않고 실제 머신과 동일하게 사용하는 것을 말한다. Xen의 반 가상화의 게스트 운영체제는 특권 명령을 실행하려면 '하이퍼콜(Hypercall)'로 하이퍼바이저에게 서비스를 요청하는 해야 한다. 운영체제에서 어플리케이션이 커널에게 '시스템 콜(System Call)'로 서비스를 요청하는 방식과 동일하다. 실제로 하이퍼 콜은 시스템 콜과 동일한 방식으로 구현되어 있다. 서비스를 요청이 (어플리케이션 -> 커널) 인지, 아니면 (게스트 운영체제 -> 하이퍼바이저) 인지에 따라 용어가 달라질 뿐이다.

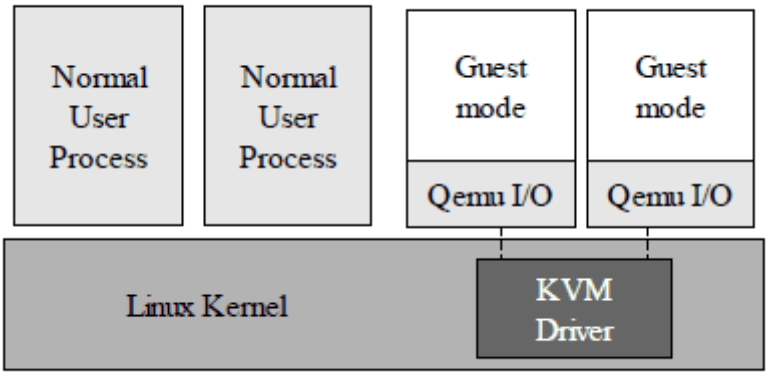
Xen에는 도메인 0라는 특수한 도메인이 존재한다. 도메인 0는 실제 물리 디바이스와 통신하는 디바이스 드라이버가 있고, 각 도메인을 제어한다. 도메인 0는 기능상 Type 2 하이퍼바이저의 호스트 운영체제와 유사하지만, Type 2에서 호스트 운영체제는 하이퍼바이저를 하나의 어플리케이션으로 관리하고, Type 1에서 도메인 0는 하이퍼바이저에서 하나의 도메인으로 관리되므로 전혀 의미가 다르다. Xen 반 가상화에서 장치 입/출력을 예를 들면, 각 도메인은 하이퍼 콜로 Xen 하이퍼바이저에게 입/출력 요청을 하게 되고, Xen 하이퍼바이저는 이를 도메인 0에게 전달하여 실제 장치와 입/출력을 하게 된다. 그러면, Xen 하이퍼바이저는 입/출력 결과를 다시 입/출력을 요청한 도메인에게 입/출력 결과를 응답한다. Xen의 반 가상화 입/출력을 할 시에 사용되는 매커니즘으로 I/O 링, 이밴트 채널(Event Channel), 프론트-앤드/백-앤드 드라이버, 그랜트 테이블(Grant Table), XenStore등이 있지만 모두 설명하려면 내용이 너무 길어지니 그런 것이 있다고만 알고 넘어가도록 하자.



<그림 2. Xen 아키텍처>

KVM은 Xen과 다른 관점에서 가상화를 제공한다. KVM은 가상화를 제공하는 하이퍼바이저를 메모리 관리자나 파일 시스템등과 같은 커널의 '서브 모듈'로 취급한다. 개인적으로는 가상화 기능이 운영체제의 기본이 되어가는 시점이므로 KVM과 같은 접근 방식이 옳다는 생각이다. KVM에서 가상화를 제공하기 위해서는 한 가지 전제 조건이 붙는데, 사용하는 CPU에서 HVM(Hardware Virtual Machine) 기능을 제공해야 한

다는 점이다. 과거 가상 메모리(Virtual Memory)를 지원하기 위해 CPU에서 페이징(Paging) 기능을 하드웨어 차원에서 제공했던 것과 같이, 최근 가상화 기능이 많이 사용되므로 CPU에서 가상화 기능을 하드웨어 차원에서 제공해주는 것이다. x86 아키텍처의 HVM으로는 Intel의 VT-x와 AMD의 SVM가 있다. 같은 x86 아키텍처이라고 할지라도 가상화 기능은 벤더마다 다르므로 벤더별로 구현해야 하는 단점이 있다. 최근에는 임베디드 시스템에서 주로 사용되는 ARM 아키텍처도 Cortex-A15 이후로 가상화 확장(Virtualization Extension) 기능이 추가되어 HVM을 제공한다.



<그림 3. KVM 아키텍처>

이렇듯 Xen과 KVM은 가상화를 제공하기 위해 애초에 접근 방법이 다르고 물론 구현도 매우 달랐'었다'. 당연히 프로젝트 초창기에는 Xen과 KVM 양 진영에서 서로 각자가 좋다고 주장하며 많이 싸웠다. 상용 하이퍼바이저인 VMWare, MS 하이퍼-V와 오픈 소스 Xen, KVM 등의 '하이퍼바이저 전쟁'이 한창 이다. (Xen의 반 가상화는 VMWare의 바이너리 변환(Binary Translation)에 대비되는 기술이다. 초창기에는 기술적인 측면에서 '반 가상화 Vs. 바이너리 변환'로 대립하였다.)

여기서 중요한 점은, Xen과 KVM은 오픈 소스라는 것이다. 원한다면 누구나 소스를 볼 수 있고, 필요에 따라 코드를 수정 할 수 있다. 리눅스가 BSD의 장점을 흡수하면서 발전해온 것처럼, Xen과 KVM도 서로 장점을 흡수하면서 발전해 나가고 있다. 이미 Xen도 매우 오래전부터 전 가상화를 지원해 왔다. Xen의 전 가상화는 KVM과 같이 HVM 기능을 활용하여 구현되어 있다. 즉, Xen의 전 가상화 기능만 떼고 봤을 때는 KVM이나 Xen이나 유사하다는 것이다. 더구나 VMWare나 KVM도 반 가상화 기능을 제공한다. Xen의 하이퍼 콜, VMWare의 VMI등의 각 하이퍼바이저 별로 다르게 구현된 반 가상화 인터페이스를 일관된 인터페이스로 추상화 하기 위하여 Paravirt Operation가 등장하였다.

결론은 이제는 가상화 시스템을 구축하기 위하여 어떤 하이퍼바이저를 선택하던간에 비슷하다는 것이다. Xen의 "하이퍼바이저 위에서 모두 놀아라." 라던지, KVM의 "커널이 곧 하이퍼바이저이다." 라던지 현 시점에는 더 이상 의미가 없어졌다. 돈이 좀 들더라도 높은 완성도에 기술 지원을 바라면 VMWare가 정답이고, MS빠라면 하이퍼-V가 정답이다. 무료로 필요에 따라 소스 수정도 할 생각이면 Xen이나 KVM 어느 것도 괜찮다. 그냥 익숙한 것을 선택하면 된다. 지금 사용 중인 하이퍼바이저가 다른 하이퍼바이저보다 성능이 떨어진다면 하이퍼바이저를 의심 할 것이 아니라 튜닝이 잘못 된 것이므로 담당 엔지니어의 능력을 의심해 보라.

크리에이티브 커먼즈 라이선스

이 저작물은 크리에이티브 커먼즈 코리아 저작자표시-비영리-변경금지 2.0 대한민

국 라이선스에 따라 이용하실 수 있습니다.

Posted by Taehun Kim
kvm, XEN, 가상화
Trackback 0 Comment 2
‹ 1 … 3 4 5 6 7 8 9 10 11 … 72 ›

open close

- Total 132,204
- Today 48
- Yesterday 88
-

RSS FEED
티스토리 가입하기! TISTORY
Taehun Kim's Blog is powered by Daum / Designed by Tistory