다중 멀티미디어 서비스 제공을 위한 OS 독립적 시스템 가상화 기술 개발

서해찬, 이세규, 김건호, 정승원
동국대학교 멀티미디어공학과
e-mail : 2012112860@dongguk.edu

Development OS-independent system virtualization technology to provide multi-media service

Hae-Chan Seo, Se-Gyu Lee, Geon-Ho Kim, Seung-Won Jung
Dept. of Multimedia Engineering, Dong-Guk University

요 약

최근 OS 독립적 가상화 기술이 다양해지고 있고, 이에 대한 관심도 높아지고 있다. 하지만 가상화 기술을 잘 알지 못하면, 이를 잘 사용하기란 쉽지 않다. 본 논문에서는 OS 독립적 가상화 기술들 중 하나인, 도커를 활용하여, 기존의 가상화 기술의 단점들을 극복하고, 사용자가 이를 쉽게 사용하기 위해 편리한 유저인터페이스를 제공한다. 다음으로 서버의 안정성과 효율적인 관리를 위해 컨테이너 관리 및 자원 데이터 처리 및 분석기술에 대해 기술한다.

1. 배경 및 소개

최근 사람들이 서버를 개인 자료보관, 자료배포, 자료 다운로드용으로 많이 사용하고 있는 추세이며, 개인서버의 필요도 또한 높아지고 있다. Host서버로부터 가상머신을 통해 개인서버를 부여 받는다면, 하이퍼바이저에 의해 무겁고 실행속도가 느려질 수 있다. 하지만 가볍고 실행속도가 빠른 도커방식을 사용한다면, 여러 운영체제에서 개인서버를 효율적으로 이용할 수 있을 것이다.

하지만 도커를 처음 본 사람들은 도커를 이용하기 어려울 뿐만 아니라, Host서버가 여러 대 일 때, 그리고 많은 사용자가 컨테이너를 요청할 때, Host서버의 안정성과 효율적인 관리가 굉장히 중요하다. 그러므로 본 프로젝트는 위의 문제를 해결하기 위해 웹 형태의 편리하고 직관적인 사용자 인터페이스를 제공하여 쉽게 사용 할 수 있도록 하려고 한다. 또한 컨테이너 관리, 서버가 여러 대일 때의 효율적인 자원 관리, 자원 데이터 분석 및 시각화를 통해 Host 서버의 안정성과 효율성을 증대시키고자 한다.

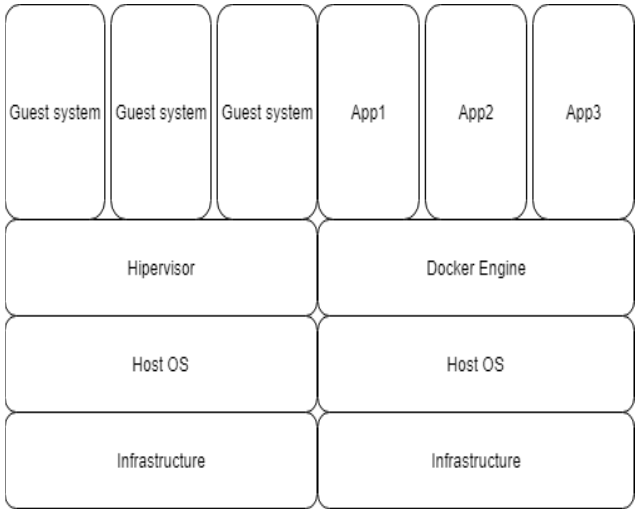
본 논문에서는 먼저 기존 OS 독립적 가상화 기술에 대해 소개하고, 두 번째, 본 논문이 활용한 도커 기술을 설명한다. 세 번째, 본 프로젝트의 목적과 세부사항을 설명하고, 마지막, 결론으로 마무리하는 구조로 되어 있다.

2.1 기존 가상화 기술과 단점

가상화란 컴퓨터에서 컴퓨터 리소스의 추상화를 일컫는 광범위한 용어이다. "물리적인 컴퓨터 리소스의 특징을 다른 시스템, 응용 프로그램, 최종 사용자들이 리소스와 상호 작용하는 방식으로부터 감추는 기술"로 정의할 수 있다.

가상화의 목적은 사용자의 어플리케이션을 저렴한 비용으로 빠르고, 안정적으로 제공하기 위한 기반 환경이다. 가상화 기술의 대표적인 특징은 사용자의 시스템 환경을 격리시킬 수 있고 컴퓨팅 자원을 사용자가 요구에 따라 유연하고 확장성 있게 지원할 수 있다는 점이다. 대표적인 기술인 클라우드 컴퓨팅은 OS 기반의 가상화로 호스트머신과 가상머신 사이에 하이퍼바이저(Hypervisor)를 통해 자원을 할당하고 관리하게 된다.

가상머신에서 자원 사용량이 많은 작업을 수행할 경우, 하이퍼바이저의 부하가 높아져 전체적인 성능 저하를 야기할 수 있다[1]. 이를 위해 기존에는 호스트 머신에 별도의 독립적인 운영체제를 올리고 네트워크와 디스크 프로세스가 또한 격리시켜 서비스를 제공했다. 그러나 어플리케이션 몇 개의 동작을 위해서 별도의 운영체제를 올리는 것은 자원의 낭비이다.

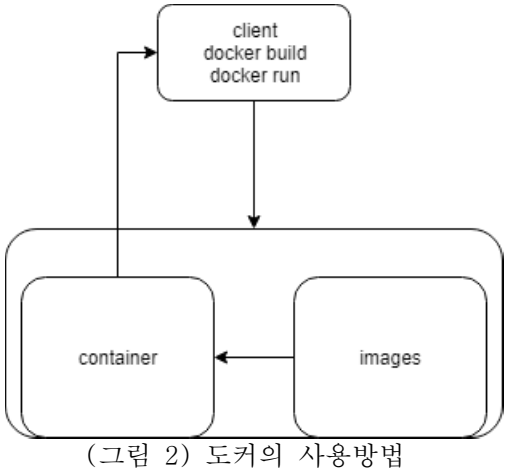


(그림 1) 하이퍼바이저 방식과 도커엔진 방식

2.2 도커 기반의 OS독립적 가상화 기술

도커는 LXC(linux Container)기반으로 만들어진 가상화 기술이다. 컨테이너는 프로세스를 격리시킨 기술로써, 어플리케이션을 실행할 수 있는 환경을 가상화하는 기술이다. 기존의 OS 독립적 가상화 기술은 하이퍼바이저 위에서 독립적인 OS 시스템을 구축하여 하나의 완벽한 시스템을 가상화한 반면, 컨테이너는 하나의 OS 시스템 위에서 어플리케이션 실행환경만 가상화한다.

이렇게 하면, 도커는 기존의 OS 독립적 가상화 기술보다 가볍고, 빠르다. 또한 설치, 삭제 시간 또한 훨씬 빠르다. 마지막으로 어플리케이션 구동환경을 컨테이너로 만들어주기 때문에 배포가 단순하고, 편리하다. 다음으로 도커의 사용방법(그림2)을 살펴보겠다.



(그림 2) 도커의 사용방법

'docker pull' 명령어로 도커 이미지를 받는다. 도커 이미지를 바탕으로 ‘docker build’, 혹은 ‘docker run’ 명령어를 통해 컨테이너를 만든다. 컨테이너를 만들고, 이를 저장하기 위해 'docker save'나 'docker commit ‘ 명령어를 사용한다.

위와 같이 이미지를 받고, 컨테이너를 만드는 등의 도커의 핵심기능은 Docker Engine에서 담당하고 있다. Docker Engine은 Host서버와 컨테이너 사이에 위치해있다.

2.3 프로젝트의 목적

본 프로젝트의 목적은 다음과 같다. 첫째, 편리한 유저인터페이스 제공이다. 만약 사용자가 개인서버를 요청한다면, 웹을 통해 개인서버를 쉽게 사용할 수 있다. 두 번째, 컨테이너에 할당된 자원(CPU, GPU, Memory등)을 효율적으로 관리할 수 있다. 셋째, 하나의 Host서버에 있는 많은 자원을 사용하지 않게끔 여분의 서버에 사용자의 요청을 분산시킬 수 있다.

2.3.1 유저인터페이스

도커를 완전히 활용하기 위해서는 도커의 명령어와 많은 옵션들을 알고 있어야 한다. 도커의 명령어와 옵션은 리눅스의 명령어보다 쉽고 직관적이지만 Mac이나 Windows에서 명령 프롬프트를 통한 작업에 익숙치 않은 사용자들은 도커 사용에 어려움을 느낄 것이다. 따라서 그런 사용자들이 도커를 사용하는데 어려움을 느끼지 않도록 친숙한 GUI방식과 아주 간단한 CLI방식으로 도커를 사용할 수 있는 웹을 구성하였다.

2.3.2 서버 자원 데이터 처리, 분석,시각화

데이터 처리 및 분석을 하는 이유는 상황, 시간에 따라 Host 서버에 요청하는 자원이 각각 다르다. 어떤 상황에서는 서버 자원의80%를 이용하는 반면 어떤 상황에서는 서버의 자원을 20% 정도 만을 이용할 수 있다. 따라서 자원이 많이 필요한 상황, 시간, 조건에 따라 서버의 자원 할당률도 그에 맞게 유동적으로 진행되어야 한다.

예를 들면 오후1시부터 4시까지는 Host서버의 자원 사용률이 90%가 넘어가는 과이용 상태라면 이 시간대의 자원사용률을 분석하여 Host서버에서 자동적으로 자원이 여유가 있는 다른 여분의 서버에게 그 사용률을 전달해준다.

Host서버에서 컨테이너를 사용하게 되면 컨테이너에 할당되는 컴퓨팅자원(Cpu, Gpu, Network, Memory등)을 로그기록으로 볼 수 있다. 이 로그기록에서 사용자에게 필요한 자원들의 데이터를 바탕으로 시계열 분석(시간의 흐름에 따라 기록된 데이터를 분석하고 여러 변수들간의 인과관계를 분석하는 방법론)등을 통해 웹에서 한눈에 분석결과를 알아볼 수 있도록 구성하였다.

2.3.3 컨테이너 매니지먼트

하나의 컨테이너가 너무 많은 자원을 독점하게 되면 Host서버의 자원을 효율적으로 활용할 수 없게 된다. 또한 한 서버가 많은 컨테이너에게 자원의 90%를 준다고 한다면, 전체적인 자원 이용률이 많으므로 다음 컨테이너를 실행할 때 그 컨테이너에 할당해 줄 자원의 양을 줄여 전체적인 자원의 균형을 맞춰야 한다. 따라서 Host서버 자원의 안정화와 개개인의 컨테이너 사용의 안정성을 위해서 효율적인 자원 할당 알고리즘을 개발하였다.

결론

컨테이너 기반의 오픈소스 가상화 플랫폼인 도커는 컨테이너 생성과 배포에 있어 다양한 장점을 가지고 있다. 또한 Docker Hub를 통해 요구사항에 맞는 컨테이너 이미지를 무료로 배포 관리하고 있다. 사용자는 이 곳에서 원하는 이미지를 간단하게 가져올 수 있다. 그리고 도커의 컨테이너 가상화 기술은 기존에 OS를 통한 클라우드 환경의 가상머신(예를 들어 VM)에 비해 Cgroups와 네임스페이스를 활용하여 프로세스를 격리함으로써 더욱 빠르고 유연하게 독립된 어플리케이션 실행 환경을 제공할 수 있다.
이런 장점이 있음에도 불구하고 아직까지 사람들이 도커를 모르는 경우가 많다. 본 프로젝트는 이러한 도커를 사용자가 쉽게 사용할 수 있도록 웹 기반의 편리한 UI 제공과 사용자의 서버 사용 자원 데이터 분석을 바탕으로 효율적인 알고리즘을 통해 사용자에게 최적화된 개인 서버를 제공하고 있다. 앞의 장점을 바탕으로 다양한 개발환경을 쉽게 개발, 구축하고 배포할 수 있어 시간과 비용의 절감을 기대할 수 있다.

Acknowledgments
"본 연구는 과학기술정보통신부 및 정보통신기획평가원의 SW중심대학지원사업의 연구결과로 수행되었음" (2016-0-00017)

참고문헌

[1] Daniel J. Abadi, “Data Management in the Cloud:Limitations and Opportunities”, In IEEE DE Bulletin, vol 32(1), pp.3-12, Feb 2009.