다양한 사용자 환경을 위한 도커 컨테이너 기반 스트리밍 서비스 구현

김성환+, 이왕광+, 염성웅+, 최진태+, 송왕철*, 김경백+
전남대학교 전자컴퓨터공학부+, 제주대학교 컴퓨터공학과*
e-mail :tjdghks0531@gmail.com, kwang9092@gmail.com, yeom9876@naver.com,
jefron1100@gmail.com, kingiron@gmail.com, kyungbaekkim@jnu.ac.kr

Implemetation of Docker Container based Streaming Service for Heterogeneous User Environment

Seonghwan Kim+, Wangkwang Lee+, Sungwoong Yeom+, Jintae Choi+,
Wangcheol Song*, Kyungbaek Kim+
Dept. Electronics and Computer Engineering, Chonnam National University+,
Department of Computer Engineering, Jeju National University*

요 약

최근 스트리밍 서비스에 대한 관심이 증가함에 따라, 다양한 환경에서 여러 가지 기기를 활용하는 스트리밍 서비스 환경이 활성화되고 있다. 또한, 중앙집중형 방식인 클라우드 컴퓨팅 기반의 스트리밍 서비스보다 사용자 및 수요자에 가까운 곳에서 스트리밍 서비스를 손쉽게 시작 할 수 있는 엣지 클라우드 컴퓨팅에 대한 연구가 활성화 되고 있다. 본 논문에서는 다양한 사용자 요구를 충족시킬 수 있는 스트리밍 서비스를 손쉽게 제공하기 위해 도커를 활용한 스트리밍 서비스 자동화기법을 구현하였다. RTMP를 사용하여 비디오 데이터를 안정적으로 전달하고, 사용자의 다양한 수요를 맞춰줄 수 있도록 스트리밍 소스를 여러 가지 형태로 인코딩하여 제공하는 서비스를 설계하고 구현하였다. 이를 통해, 스트리밍 서비스 제공시 사용자 환경에 알맞은 서비스를 제공하면서 네트워크를 효율적으로 활용하는 것이 가능함을 확인하였다.

1. 서 론

최근 사용자들은 동영상 서비스에 많은 관심을 가지고 있는데, 이를 제공하기 위해서 실시간으로 사용자에게 제공하기 위한 스트리밍 기술이 발달하게 되었다. 사용자들에게 시간에 구애받지 않고 실시간으로 동영상 서비스를 제공하기 위해서는 일반 PC와는 기능이 다른 고가의 스트리밍 전용서버가 필요하며, 업체들은 이를 위해 아낌없는 투자를 하고 있다.[4] 즉 미래 사회와 산업이 요구하는 동적인 자원 공급과 이에 근간한 서비스 제공을 감당할 수 있는 유연하면서도 경제적인 인프라의 필요성이 대두되고 있으며, 이에 대응하여 거대규모(hyper-scale)의 클라우드 데이터센터들을 중심으로 ICT 인프라의 하드웨어적인 모습이 나타날 것이다. 즉 아마존 AWS, 구글 GCE, MS Azure 등의 명칭으로 불리는 거대규모 클라우드 데이터센터들이 통신사업자나 대형 서비스 회사의 데이터센터들과 점차적으로 연계되면서 미래 ICT 인프라의 중심으로 자리를 잡을 것으로 예상된다. 그렇지만 소비자나 기업 어플리케이션과는 달리, 산업 분야의 어플리케이션의 경우 중앙 집중방식의 데이터 저장 및 처리에 전적으로 의존하는 것이 불가능하다. 산업용 어플리케이션의 경우, 즉각적으로 반응해야 할 경우가 자주 있는데, 원격지로 데이터를 전송할 때 병목현상에 영향을 받지 않아야 한다. 이렇게 실시간으로 대응하려면 엣지(Edge)와 클라우드를 결합한 아키텍처가 요구 되는데, 이 모습이 21세기 산업인터넷 구현을 위해 필요한 아키텍처이다.

엣지 컴퓨팅은 데이터 소스와 더 가까운 위치에서 데이터를 분석하고 통찰력을 얻을 수 있게 해준다.[5] 이는 네트워크 가장자리에 위치한 기기에서 컴퓨팅을 지원하는 것이다. 최소한, 엣지는 산업 데이터를 최적화하는 게이트웨이 역할을 하며, 자산과 물리적 운영의 모든 측면을 최적화할 수 있는 데이터와 분석적 통찰력을 제공한다. 지금까지는 데이터를 단순히 저장만 할 뿐, 이를 활용할 기회를 놓치는 경우가 많았다. 하지만 엣지 디바이스 덕분에 데이터를 분석하여 통찰력을 얻고 이를 기반으로 여러 작업과 관련된 조치를 위한 쉽게 취할 수 있게 되었다. 엣지 센서, 제어장치, 게이트웨이, 인프라 기계, 사내 기기와 서버 랙, 클라우드 등 다양한 수준에서 실시간 분석과 어플리케이션 로직이 구현될 수 있다.

본 논문에서는 이러한 엣지 컴퓨팅을 활용하여 스트리밍 서비스를 지원해야 하는 상황을 위한 시스템 설계 및 구현을 수행하였다. RTMP와 Nginx, FFmpeg를 사용하여 다양한 사용자 요구를 충족시킬 수 있는 스트리밍 서비스를 만들고 엣지 클라우드 환경에서 성능검사를 해보았다.

또한 도커를 활용하여 서비스 제공자가 수요자들의 환경에 맞게 쉽게 자동화할 수 있도록 하였다.

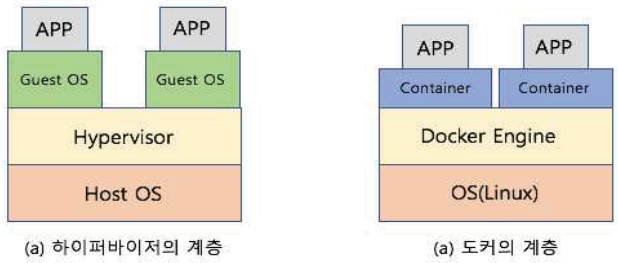
2. 관련 연구

2.1 RTMP

리얼 타임 메시징 프로토콜(Real Time Messaging Protocol)[1]은 어도비 시스템즈사의 독점 컴퓨터 통신 규약이다. RTMP는 주로 오디오, 비디오 및 기타 데이터를 인터넷을 통해 스트리밍할 때 쓰인다. RTMP는 어도비 플래시 플레이어와 서버 사이의 통신에 이용된다. 처음 목표는 오직 플래시(Flash)에만 쓰이는 것이었다. 플래시 이외에도 어도비 라이브사이클 데이터 서비시즈 ES와 같은 다른 응용프로그램에서 RTMP이 쓰이고 있다. RTMP의 최근 규격은 2009년 1월 20일에 어도비에서 발표되었다.

2.3 도커(Docker)

도커[3]는 오픈소스 기반 리눅스 컨테이너 관리 도구이다. 컨테이너란 리눅스 커널에서 제공하는 기술로, 가상화(Virtualization) 기술과 비슷하지만 가상화보다 훨씬 가벼운 기술이라고 볼 수 있다. 과거에는 리눅스 컨테이너를 이용했지만 지금은 도커 자체 컨테이너 기술을 개발하여 사용하고 있다.



(그림 1) 하이퍼바이저와 도커의 계층 구조

호스트 컴퓨터에서 다수의 운영체제를 돌리기 위한 플랫폼을 하이퍼바이저(Hypervisor)라 하며, 하이퍼바이저의 가상화 기술은 VMware, Xen, 리눅스 KVM 등 여러 종류가 있다. 그림 1의 (a)는 하이퍼바이저의 계층을 나타낸 것이다. 가상 머신 자체는 완전한 컴퓨터를 생성하는 것이라 볼 수 있기 때문에 게스트 OS(Operating System)가 있어야 하고, 그 위에 프로그램들이 실행된다. 따라서 호스트 컴퓨터의 자원을 사용할 때 하이퍼바이저를 통해 호스트 OS에게 자원을 요청해야하므로, 그 시간 동안 오버헤드가 발생해 호스트 컴퓨터에서 프로그램을 수행할 때보다 성능이 느리다는 단점이 있다. 그러나 도커는 게스트 OS를 설치하지 않고, OS 이미지가 동작하도록 CPU와 메모리 영역만 가상화한다. 그림 1(b)의 도커 계층구조를 보면 도커는 도커 엔진 위에 OS 전체를 설치하는 것이 아닌 실행하고자 하는 OS 이미지의 실행파일과 라이브러리를 컨테이너 이미지로 만들면 된다. 그 외에 구동하는 데 필요한 OS자원이나 라이브러리는 호스트 컴퓨터의 환경을 공유한다. 따라서 하드웨어를 가상화하는 계층이 없고 호스트 자원을 공유하기 때문에 메모리, 파일시스템, 네트워크 속도 등 가상 머신에 비해 빠르다.

2.3 FFmpeg

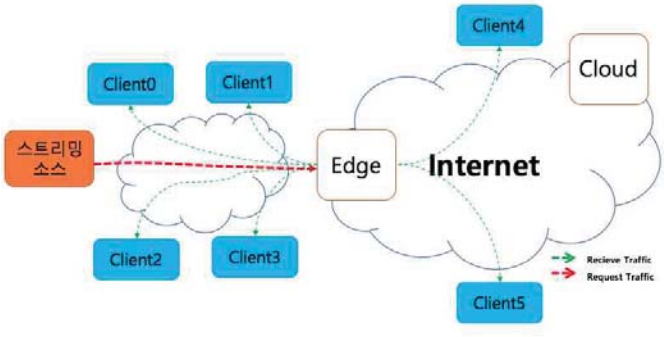
FFmpeg는 Michael Niedermayer의 주도하에 개발되고 있는 모든 동영상, 음악, 사진 포맷들의 디코딩과 인코딩을 목표로 만들어지고 있는 LGPL또는 GPL라이센스를 따르는 오픈소스 프로젝트로 만들어졌다. FFmpeg 프로젝트 자체가 모든 영상의 디코딩/인코딩을 지향하기 때문에 추가적인 코덱의 설치가 필요없다. ffmpeg.exe파일 하나로도 인코딩이 가능하다. 또한 멀티코어를 공식적으로 지원한다. FFmpeg는 쉬운 인코딩을 위해 각종 프리셋을 포함하고 있다. 그러므로 다른 인코더들을 쓰는 것보다 FFmpeg을 이용해서 프리셋을 활용하는 것이 더 나은 화질을 보장한다.

2.3 Nginx

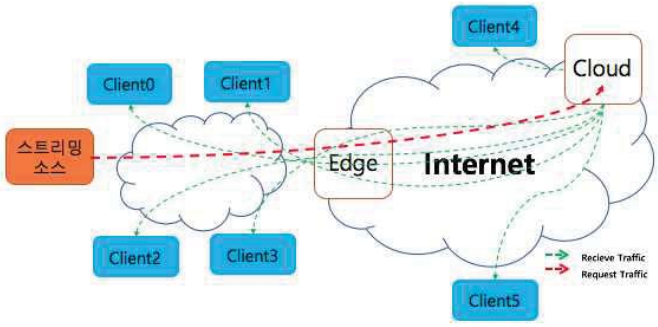
Nginx는 웹 서버 소프트웨어로, 가벼움과 높은 성능을 목표로 한다. 웹 서버, 리버스 프록시 및 메일 프록시 기능을 가진다. Nginx는 요청에 응답하기 위해 비동기 이벤트 기반 구조를 가진다. 이것은 아파치 HTTP 서버의 스레드/프로세스 기반 구조를 가지는 것과는 대조적이다. 이러한 구조는 서버에 많은 부하가 생길 경우의 성능을 예측하기 쉽게 해준다.

3. RTMP기반 스트리밍 서비스 설계

본 논문에서는 엣지 클라우드환경에서 사용자들의 환경에 맞게 스트리밍 서비스를 제공하고 도커(Docker)를 이용하여 RTMP스트리밍 서버를 쉽게 구축할 수 있는 방법을 제시하였다. 그림2와 그림3은 엣지환경과 클라우드환경에서의 전체적인 네트워크 흐름도이다. 그림과 같이 비교했을 때 만약 서비스 이용 대상이 서비스 제공자와 가까운 네트워크에 있다면 엣지환경을 이용하는 것이 클라우드 환경을 이용하는 것보다 네트워크 비용 감소 및 성능 향상들의 이점들이 있다.



(그림 2) Edge 환경의 서비스 네트워크 흐름도



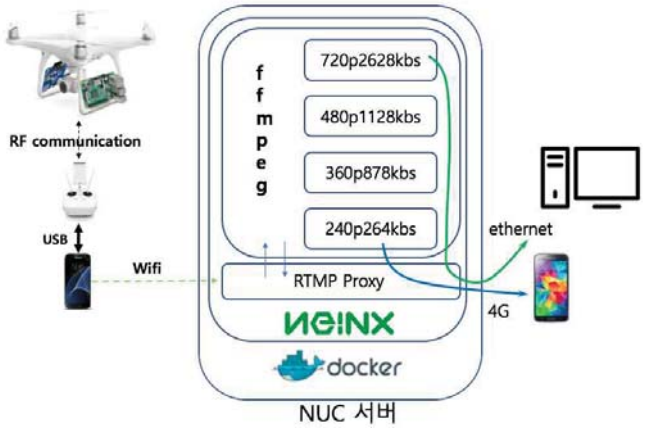
(그림 3) Cloud 환경의 서비스 네트워크 흐름도

서비스 제공자는 엣지환경에 서버를 생성하고 도커를 이용하여 RTMP스트리밍 서버를 구현한다. 그리고 스트리밍 소스를 서버에 보낸 후 서버는 여러 사용자 환경에 대비하여 받은 스트리밍 소스를 여러 환경에 맞게 인코딩을 하여 준비한다. 그리고 사용자 측에서 사용자 기기에 적합한 스트리밍 정보를 서버 측에 요청을 하고 서버 측에서는 요청에 맞는 스트리밍 소스를 제공한다.

사용자측에서 스트리밍 소스를 확인하기 위해서는 SMTP프로토콜을 제공하는 플레이어가 필요하다. SMTP를 제공하는 플레이어들은 웹, 휴대폰 어플리케이션 등등 다양한 환경에 존재하기 때문에 사용자는 어떤 환경에서든 스트리밍 정보를 확인할 수 있다. 예를 들어서 VLC나 JW Player와 같은 플레이어들이 있다.

4. 구현 및 검증

본 논문에서는 앞서 설계한 시스템을 가지고 RTMP스트리밍 서버를 구현 해 보았다. 서버 환경으로는 NUC을 사용하였으며 스트리밍 소스는 드론에서 촬영한 영상정보를 이용하였다. 드론은 DJI사에서 나온 PHANTOM 4를 사용하였다. 사용자측에서는 휴대폰과 웹을 통해서 스트리밍 소스를 확인해 보았다. 서버에서는 Linux14.04LTS 버전을 설치하고 인코딩 설정을 한 후 도커 컨테이너를 실행시켜 RTMP스트리밍 서버를 구현하였다. 시스템 구성도는 그림4과 같다.

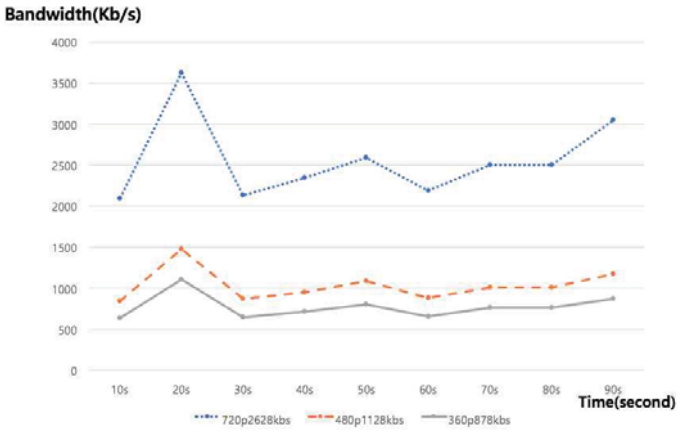


(그림 4) 시스템 구성도

드론에서 촬영한 영상정보를 첫 번째 핸드폰에서 받은 후 영상정보를 WIFI 이용하여 NUC서버에 있는 NGINX로 전달한다. 그리고 그 내에서 FFmpeg 인코더를 이용하여 720p2628kbs, 480p1128kbs, 360p878kbs, 240p264kbs의 설정들로 인코딩을 해준다. 720p는 화질이고 2628kbs는 평균 대역폭을 설정한 것이다. 그리고 컴퓨터나 핸드폰에서 요청한 정보에 맞는 영상정보를 제공해준다. 영상 수신기기에서는 RTMP프로토콜을 지원해주는 플레이어를 이용해서 수신정보를 확인할 수 있다. 컴퓨터에서 VLC플레이어상에서 스트리밍 서비스를 받는 화면은 그림5와 같다.

(그림 5) 컴퓨터 테스트 화면

또한 인코딩 설정에 따라 대역폭을 비교 분석 해 보았다. 그림6는 720p2628kbs와 480p1128kbs, 360p878kbs로 인코딩을 했을 때 대역폭을 비교해본 그래프이다. 보다시피 720p2628kbs일 때 해상도는 좋지만 대역폭이 높아서 네트워크 환경이 좋지 않을 때는 적합하지 않다. 또한 핸드폰에서는 컴퓨터에 비해 낮은 해상도가 적합하기 때문에 상대적으로 낮은 해상도의 영상정보를 제공해주는 것이 적합하다. 이러한 사용자 환경에 알맞은 서비스를 제공함으로써, 네트워크 자원 낭비를 줄일 수 있다.



(그림 6) FFmpeg 설정에 따른 요구되는 네트워크 대역폭 비교

5. 결론

본 논문에서는 도커(Docker), RTMP를 이용하여 다양한 사용자 환경에서 각 사용자환경에 적합한 컨텐츠를 제공함으로써 트래픽을 최적화 할 수 있는 스트리밍 서비스 자동화 기법을 설계하고 구현하였다. 제안되는 기법에서는 엣지 클라우드에 서버 자원을 이용하기 때문에 전체 네트워크 비용을 감소할 수 있고 Docker를 이용하기 때문에 서비스 제공자가 수요에 맞게 쉽게 수정 가능하며, 자동화 시스템도 쉽게 구현할 수 있다.

향후 제안된 기법을 기반으로 사용자의 요구를 자동으로 인지하고 네트워크 설정 및 서버설정을 할 수 있도록 발전시킬 계획이다. 또한 Koren Playground에 적용하여 확장테스트를 수행할 계획이다.

감사의 글

본 연구는 한국정보화진흥원(NIA)의 미래네트워크선도시험망(KOREN) 사업 지원과제의 연구결과로 수행되었음 (17-951-00-001).

참 고 문 헌

[1] https://en.wikipedia.org/wiki/Real-Time_Messaging_Protocol

[2] http://lkcl.net/rtmp/RTMPE.txt

[3] D. MERKEL, “Docker: Lightweight linux containers for consistent development and deployment”. Linux J. 2014 (2014).

[4] 김수정, “Design of Cost-Effective Clustering System for Streaming Service”, 2001

[5] http://www.gereports.kr/edge-computing-door-iot-data-kingdom/, “엣지 컴퓨팅 - 사물인터넷과 생각하는 기계”, 2016