도커 컨테이너 기반 사용자 맞춤형 CCTV 서비스 설계 및 구현

염성웅, 권태용, 김경백
전남대학교 전자컴퓨터공학부
e-mail : yeom4032yeom4032@gmail.com, taaii6569@naver.com, kyungbaekim@.jnu.ac.kr

Design and Implementation of Docker Container based User Customized CCTV Service

Sungwoong Yeom, Taeyong Kwon, Kyungbaek Kim
Department of Electronics and Computer Engineering, Chonnam National University

Abstract

최근 네트워크 서비스의 발달과 카메라 장치의 소형화에 따라 다양한 스트리밍 서비스들이 각광을 받고 있다. 특히, 특정구역을 모니터링 하는 CCTV 서비스의 경우 다양한 사용자의 요구사항을 만족시키기 위해 여러 오픈소스 CCTV 플랫폼들이 개발되고 있다. 본 논문에서는 네트워크 카메라를 사용하는 사용자 맞춤형 CCTV 서비스를 제공하기 위한 도커 컨테이너 기반 CCTV 서비스 관리 프레임워크 설계 및 구현에 대한 내용을 기술한다. 제안하는 프레임워크는 다양한 사용자 요구를 충족시킬 수 있도록 오픈소스 CCTV 플랫폼들을 도커 컨테이너기반으로 관리하는 서버를 포함하고, 해당 관리 서버에 접근하기 위한 Restful API 를 제공한다. 또한, 네트워크 카메라는 RSTP 기반의 스트리밍 데이터를 오픈소스 CCTV 플랫폼에 제공함으로써 스트리밍 서비스의 확장성을 확보할 수 있다.

1. 서 론

과거의 1 인 미디어는 어느 정도의 팬층과 영상 기술에 대한 전문성이 있는 일부 사람들에게만 한정되었다. 네트워크 카메라 설치 및 영상 전송 기법 등에 대한 지식이 부족하면, 개인이 미디어 서비스를 제공하는 것이 힘들었다.

최근, 1 인 미디어는 특별한 기술에 대한 이해가 없더라도, 스마트폰만 있으면 언제나 본인의 생활을 생생하게 다른 사람에게 보여줄 수 있는 진정한 의미의 '1 인 미디어'가 되어가고 있다. 이러한 변화는 실시간으로 사용자의 영상을 전달하는 라이브 스트리밍 기술의 발달에 기인한다. 라이브 스트리밍은 무엇보다 단순하고 쉬운 것이 매력으로, 스마트폰에서 앱을 열고 시작버튼만 누르면 실시간 라이브 스트리밍을 시작할 수 있다. 이러한 추세에 따라, 온라인 게임 방송, 여행지의 모습을 전해주는 라이브 방송, 길거리 버스킹 라이브 방송, 그리고 팬과 소통하고 싶은 유명인들의 라이브 방송이 활성화되고 있다.

이러한 라이브 방송 외에도, 라이스 스트리밍 기술의 발달에 따라 CCTV 와 같은 모니터링 어플리케이션 또한 활성화 되고 있다. 라이브 방송과 비슷하게, 네트워크 카메라를 설치하고 스마트폰의 어플리케이션을 통해 해당 카메라의 영상을 손쉽게 볼 수 있다. 또한, 다양한 사용자들의 요구사항을 만족시키기 위해 다양한 분석 기능들이 포함된 오픈소스 CCTV 플랫폼들이 개발되고 있다. 예를 들어, 특정한 지역을 모니터링 하는 CCTV 서비스의 경우, 제공되는 스트리밍 데이터를 기반으로 움직임 인식, 개체 인식 및 상황 인식과 같은 분석 서비스가 필요하다.

하지만, 사용자들에게 시간에 구애 받지 않고 실시간으로 스트리밍 서비스 및 분석 서비스를 제공하기 위해서는 일반 PC 와는 기능이 다른 고가의 스트리밍 전용서버가 필요하다. 또한, CCTV 와 같은 개인의 사생활이 보장되어야 하는 서비스의 경우, 일반적인 상용 미디어 서비스를 이용하는 것보다는 프라이버시가 보장되는 개인 컴퓨팅 환경에서 서비스를 실행하는 것이 낫다.

이러한 문제를 해소하기 위해, 엣지 컴퓨팅 개념을 활용하여, 사용자가 원하는 분석서비스를 제공하는 CCTV 소프트웨어를 엣지 컴퓨팅 서버에서 동적으로 실행하여 원하는 CCTV 비디오 스트리밍 및 분석 서비스를 제공하는 방안을 고려할 수 있다.

본 논문에서는 이러한 엣지 컴퓨팅 환경에서 사용자 맞춤형 CCTV 서비스를 제공하고자 할 때 필요한

도커 컨테이너 기반 CCTV 서비스 관리 프레임워크의 설계를 제안하고 그 초기 구현에 대한 내용을 기술한다.

2. 관련연구

1.RTSP

실시간 스트리밍 프로토콜(Real Time Streaming Protocol, RTSP)은 IETF 가 1998 년에 개발한 통신 규약으로, RFC 2326 에 정의되어 있다. RTSP 는 스트리밍 시스템에 사용되며, 또한 미디어 서버를 원격으로 제어하기 위해 할 때 사용된다. 명령어는 "PLAY", "PAUSE" 같이 VCR 동작과 비슷하며, 시간 정보를 바탕으로 서버에 접근한다. RTSP 는 실제 미디어 스트리밍 데이터를 전송하지는 않고, 대부분의 RTSP 서버는 RTP 규약을 사용해서 전송 계층으로 실제 오디오/비디오 데이터를 전송한다.

2.NodeJS

본 프레임워크 설계에서는 서비스 관리 서버 및 Restful API 관리 서버 구현에 Node.js 를 활용한다. Node.js 는 확장성 있는 네트워크 어플리케이션 개발에 사용되는 소프트웨어 플랫폼으로, 서버 사이드에서 많이 사용되고 있다. 개발에는 주로 자바스크립트(Javascript)가 사용되며, Non-blocking I/O 와 단일 스레드 이벤트 루프를 통한 높은 이벤트 처리 성능을 가지고 있다. 또한, 다양한 웹 프레임워크 패키지들을 제공하고 이를 활용하여 간단한 웹 서버 혹은 API 서버를 쉽게 구성 할 수 있다.

3.RESTful API

REST 란 어떤 자원에 대해 CRUD(Create, Read, Update, Delete) 연산을 수행하기 위해 URI(Universal Resource Indicator)로 요청을 보내는 것으로, Get, Post 등의 방식을 사용하여 요청을 보내며, 요청을 위한 자원은 특정한 형태로 표현된다. 위와 같이 CRUD 연산에 대한 요청을 할 때, 요청을 위한 Resource (URI)와 이에 대한 Method (Get/POST) 그리고 자원의 형태(JSON)를 사용하면 표현이 명확해지므로 이를 REST 라 하며, 이러한 규칙을 지켜서 설계된 API 를 Rest API 또는 Restful 한 API 라고 한다.

4. 도커(Docker)

도커는 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. 이러한 도커의 컨테이너를 로컬에서 제어하기 위해 도커 데몬과 클라이언트는 유닉스 소켓을 사용한다. 이때 도커데몬을 TCP 소켓으로 실행 할 경우 도커가 제공하는 API 를 통하여 원격에서 컨테이너 생성 제거 및 이미지 생성이 가능하다. 해당 API 는 HTTP REST 형식으로 구현되어있다.

5. Open CCTV Software

일반 사용자들이 쉽게 IP camera 를 사용 할 수 있게 됨에 따라 다양한 영상분석 오픈 소프트웨어들이 개발 되었다. Open CCTV Sotfware 를 설치함으로써 IP camera 에서 제공하는 영상을 이용해 움직임 감지, 출입 횟수 체크가 가능하며 이벤트 당시의 영상을 저장할 수 있다. 또한 영상에서의 움직임이 감지된 위치와 시간을 사용하여 HeatMap 과 시간대별 움직임 횟수 데이터를 Line, Radar, Donut 그래프를 통하여 시각화 할 수 있다. 뿐만 아니라 해당 IP camera 의 영상을 스트리밍하는 기능을 제공함으로써, 많은 사람들이 접근해 영상을 볼 수 있는 프록시 서버로써의 기능을 제공 할 수도 있다.

3. 도커 컨테이너 기반 사용자 맞춤형 CCTV 서비스 설계 및 구현

본 논문에서 제안하는 도커 컨테이너 기반 사용자 맞춤형 CCTV 서비스 프레임워크는 엣지 컴퓨팅 환경에서 사용자가 네트워크 카메라를 설치하고, 요구사항을 만족하는 CCTV 서비스를 선택하여 엣지 서버에 자동으로 실행하는 것을 지원하는 프레임워크이다. 그림 1 은 해당 프레임워크의 전체적인 개념을 나타낸다. 에지 레이어에서는 여러 사이트(Site)의 사용자가 다양한 네트워크 카메라와 다양한 소프트웨어 모듈을 활용하며, 해당 서비스들은 Core Network 를 통해 중앙 클라우드 서비스 및 기타 네트워크 사용자들에게 제공된다.

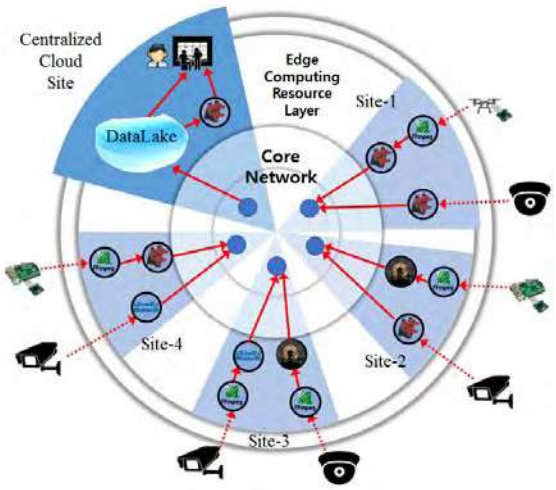

그림 1. 사용자 맞춤형 CCTV 서비스 개념

제안하는 프레임워크는 사용자로부터 네트워크 카메라의 스트리밍 서비스 정보와 서비스 요구사항정보를 받는다. 이후, 프레임워크는 도커 remote API 를 이용하여 엣지 컴퓨팅 리소스에 서비스 요구사항에 알맞은 오픈소스 CCTV 도커 컨테이너를 실행한다. 이때, 해당 도커 컨테이너는 도커 허브에서 가져오거나 도커파일을 통해 생성 할 수 있다. 실행된 도커 컨테이너는 제공된 네트워크 카메라의 스트리밍 서비스를 위한 프록시 서버로 동작할 수 있으며, 자체적인 스트리밍 피드를 제공한다.

사용자는 제안하는 프레임워크에서 제공하는 API 를 활용하여 해당 컨테이너를 실행하거나, 생성된 컨테이너 및 스트리밍 피드 정보를 Json 형식으로 받을 수 있다. API 는 GET 방식과 POST 그리고 DELETE 방식으로 구성 하였다. GET 메소드를 통하여 각 엣지 컴퓨팅 리소스의 컨테이너 정보를 받을 수 있고 카메라의 이름을 파라미터로 넘겨줌으로써 특정 카메라의 정보를 받는다. POST 방식은 컨테이너를 생성하는 메소드로 아이피 카메라의 RTSP 의 URL, Open CCTV 의 종류 그리고 카메라의 이름을 받아 해당 소프트웨어 컨테이너를 실행한다. DELETE 방식의 메소드는 클라우드 서버와 카메라의 이름을 전달 받아 해당 서버의 컨테이너를 중지하고 제거한다. 그림 2 는 해당 API 에서 활용하는 Json 데이터 형식을 나타낸다.

```
{
  "Obox": [{
    "name": "JNU",
    "container": [{
      "name": "containerName",
      "type": "container type",
      "webport": "port the container using",
      "camera": [{
          "name": "cameraName",
          "url": "http proxy url"
        }
      ]
    }]
  }]
}
```

그림 2. 프레임워크 API 에서 사용하는 JSON 데이터 형식

제안된 프레임워크 설계의 검증을 위해 컴퓨팅 리소스(NUC)와 네트워크 카메라를 포함하는 로컬네트워크 상에서 해당 프레임워크를 구현하였다. 컴퓨팅 리소스로는 Intel NUC 을 사용하였고, 도커 서버를 NUC 에 설치하고, 도커 Remote API 를 사용하여 도커 컨테이너를 관리하였다. 네트워크 카메라로는 Foscam 과 라즈베리파이 캠을 사용하였다.

오픈소스 CCTV 플랫폼으로는 Kerberos IO 와, Zoneminder 를 사용하였고 Dockerfile 을 통하여 각 CCTV 플랫폼 실행을 위한 이미지를 생성하였다. Kerberos IO 는 하나의 컨테이너에 하나의 네트워크 카메라 설정이 가능하고 부가기능으로 스트리밍 서버 및 움직임 검출 데이터의 시각화 데이터 그래프를 제공한다. Zoneminder 는 하나의 컨테이너에 다수의 네트워크 카메라 설정이 가능하여 하나의 컨테이너를 통해 다수의 네트워크 카메라에서 제공되는 영상을 스트리밍 할 수 있다

사용자는 네트워크 카메라를 설치하고 카메라 스트리밍을 위한 RTSP URL 을 제공하고 Keberos IO 또는 Zoneminder 를 선택하여 CCTV 서비스 컨테이너를 실행한다. CCTV 서비스 컨테이너 실행이 성공하면, 해당 서비스에서 제공하는 대쉬보드를 접근하거나 제안하는 API 를 통해 스트리밍 피드 URL 을 받을 수 있고, 이 스트리밍 피드를 통해 서비스의 확장성을 확보할 수 있다.

4. 결 론

본 논문에서는 다양한 사용자 요구사항을 지원하는 CCTV 서비스를 제공하기 위한 도커 컨테이너 기반 사용자 맞춤형 CCTV 서비스 프레임워크에 대한 설계를 제안하였다. 해당 프레임워크는 Docker API 및 Restful API 를 활용해 서비스 제공자는 수요에 맞추어 서비스를 쉽게 수정할 수 있으며 사용자는 요구사항에 알맞은 서비스를 자동으로 제공받을 수 있도록 한다. 향후, 사용자 분포 및 서비스 리소스 분포를 고려한 효율적 서비스 실행 방안에 대해 연구할 계획이다.

Acknowledgment

This research was one of KOREN projects supported by National Information Society Agency (18-951-00-001).

References

[1] https://ko.wikipedia.org/wiki/Node.js (Nodejs)
[2] https://www.kerberos.io/ (Kerberos)
[3] https://zoneminder.com/ (Zoneminder)
[4] http://mangkyu.tistory.com/46 (Restful API)
[5] https://docs.docker.com/develop/sdk (Docker Remote API)
[6] http://www.itworld.co.kr/news/104173 (모바일 라이브 스트리밍)
[7] https://ko.wikipedia.org/wiki/실시간_스트리밍_프로토콜