컨테이너 기반 대규모 IoT 센서 데이터 수집 서비스 구현

이왕광, 김성환, 염성웅, 최진태 김경백
전남대학교 전자컴퓨터공학부

e-mail : kwang9092@gmail.com, tjdghks0531@gmail.com,
yeom9876@naver.com, jefron1100@gmail.com, kyungbaekkim@jnu.ac.kr

Implementation of Container based Scalable IoT Sensor Data Collection Service

Wangkwang Lee, Seonghwan Kim, Sungwoong Yeom, Jintae Choi, Kyungbaek Kim
Dept. Electronics and Computer Engineering, Chonnam National University

요 약

최근 모바일 등 스마트 기기의 확산으로 인해 스마트 센서 증가와 함께 기기 간의 융합 및 연결성을 확보하면서 급속도로 사물인터넷 환경에 대한 관심이 고조되고 있는 추세이다. 사물인터넷 기술은 PC, 스마트폰 등 컴퓨팅 단말을 넘어 사실상 모든 종류의 사물에 센서 네트워크의 작은 장치를 포함하여 생활 속 기기들이 실시간으로 인터넷에 연결된 환경으로 스마트 헬스케어 시스템과 산업 설비의 제어 시스템과 같은 스마트 서비스를 활성화 시키고 있다. 본 논문에서는 대규모 IoT 센서 데이터 수집을 위한 시스템을 로그 수집기 Flume, 메시징 시스템 Kafka, 데이터베이스 InfluxDB를 이용하여 설계하고, 이를 Docker Container에 올려 확장성 있고 자동화까지 할 수 있는 시스템을 초기단계까지 구현하여 검증하였다.

1. 서 론

사물인터넷(IoT)은 가트너(Gartner)가 선정하는 10대 전략기술에 2012년부터 매년 선정되어 ICT 시장의 신산업을 이끌어가는 핵심 부가가치 산업으로 급부상하고 있다. 특히, 모바일 등 스마트 기기의 확산으로 인해 스마트 센서 증가와 함께 기기 간의 융합 및 연결성을 확보하면서 ICT 융합 분야 전반에 걸쳐 급속도로 사물인터넷 환경에 대한 관심이 고조되고 있는 추세이다. 현재 ICT 산업에서 가장 이슈가 되고 있는 ICBM(IoT, Cloud, Big Data, Mobile)이 차세대 성장동력으로 주목받고 있는 가운데 인터넷 기반의 융합중심에서 사물인터넷이 실제 생활영역에 적용되면서 다양한 경제적 가치와 더불어 효율성 및 편의성이 한층 높아질 것으로 기대되고 있다.

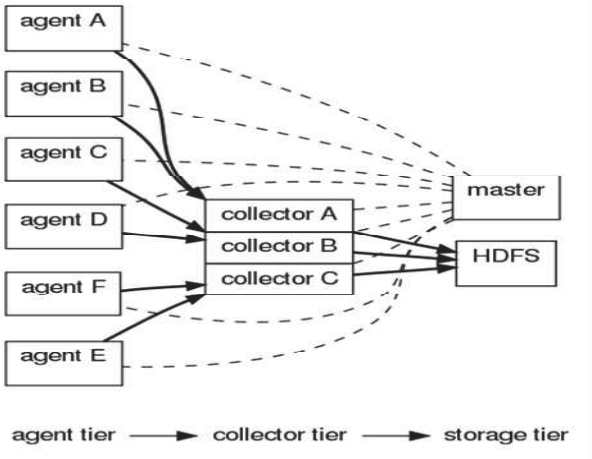
사물인터넷 산업은 시장 범주가 모호하고 타산업과의 융합을 전제로 성장하는 산업 특성상 시장 규모 예측 결과가 주요 기관별로 차이가 있지만 시스코(Cisco)는 네트워크에 연결된 사물 수가 2014년 144억 개에서 2020년 501억 개로 약 3.5배 증가할 것이라고 예측하였고, Machina Research는 M2M 시장이 2014년 45억 개에서 2024년 290억 개로 증가할 것이라고 예상하는 등 긍정적인 전망을 쏟아내고 있다. 국내의 경우에도 2015년 3조 8,000억 원에서 2022년 22조 9,000억 원까지 성장할 것이며, 특히 서비스 관련 매출의 비중이 52.6%까지 증가할 것으로 전망되고 있다. 사물인터넷의 주요 서비스 시장에서는 주요 글로벌 ICT 기업(Apple, Google 등)들이 디바이스 경쟁을 넘어 사물인터넷 기기에서 수집한 데이터를 수집・관리・분석하기 위한 플랫폼 분야의 경쟁구도가 본격화되고 있다.

이와 더불어 사물인터넷은 가전, 의료, 교통 등 모든 분야에 적용될 것으로 전망됨에 따라 대부분의 기기에 정보 획득 및 네트워크 연결 기능이 탑재되고 이를 바탕으로 스마트홈, 스마트가전, 스마트카, 스마트헬스케어, 스마트시티, 스마트물류, 스마트그리드 등 다양한 분야에서 새로운 제품과 서비스가 출현될 것이다. 사물인터넷 활용에 따른 전 세계 부가가치 규모는 시장기관에 따라 2020년까지 1.9조 달러에서 19조 달러에 달할 것으로 전망되었다.[1] 이처럼 IoT서비스의 사용이 늘어나고 있고, 많은 기업에서도 센싱 기술을 활용한 여러 시스템을 키워나가고 있다.

본 논문에서는 이러한 흐름에 맞추어 대규모 IoT 센서 데이터 수집을 위한 시스템을 설계하고, 실 구현을 통해 그 가능성을 검증하였다. 수많은 양의 센서 값들을 관리할 로그 수집기(Log Aggregator)인 Flume과 메시징 시스템(Messaging System)인 Kafka, DB로 InfluxDB를 이용하여 대규모 IoT 센서 데이터 수집을 위한 시스템을 설계하였고, 이후 검증을 위하여 앞서 설계한 시스템을 Docker Container를 이용하여 구현 및 검증하였다.

2. 배경 및 관련 기술

최근 IoT(Internet of Things)기기들이 많아지고 이를 이용한 센싱이 보편화되고 있다. IoT기기가 늘어남으로써 관리해야 할 기기들이 많아지고 수집하는 데이터의 양 또한 많아진다. 많은 양의 센서 값들을 관리하기 위해서는 이를 수집할 로그 수집기와 수집한 값을 보낼 메시징 시스템 그리고 이를 저장할 데이터베이



(그림1) Flume Architecture

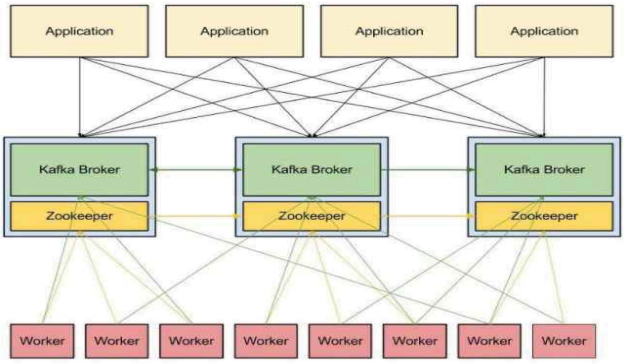
스가 필요하다. 본 논문에서 사용한 시스템에서 로그 수집기로는 Flume을 사용했고, 메시징 시스템으로는 Kafka, 데이터베이스로는 InfluxDB를 사용했다. 이후 확장성과 자동화를 위하여 위의 시스템들을 Docker Container에 올려 구축하였다.

2.1 Flume

Flume[2]은 대량의 로그데이터를 효율적으로 수집, 집계 및 이동하기 위한 분산형의 안정적인 서비스 이다. 스트리밍 데이터 흐름을 기반으로 하는 간단하고 유연한 아키텍처이며 이 시스템은 중앙에서 관리되며 동적관리가 가능하다. Flume의 계층은 3계층으로 구성되어 있으며 Agent, Collector, Storage 계층으로 총 3 가지이다. Agent계층에서 각 Agent 는 수집할 로그데이터가 생성되는 머신에 설치하는 것이 일반적이다. Agent계층에서 수집한 데이터는 Collector계층으로 전송이 된다. Collector계층은 보통 다른 머신에 있으며 여러 Collector계층으로 구성이 가능하다. Agnet계층에서 Collector계층으로 데이터를 전송할 때는 어떤 데이터를 어디로 보내고 어떻게 처리할 것인지 등에 대한 data flow를 설정할 수 있고 이 설정대로 데이터를 이동시켜 storage 계층에 저장을 한다.[3]

2.2 Kafka

Messaging System인 Kafka[4]는 분산 형 스트리밍 플랫폼으로 시스템과 시스템 혹은 시스템과 애플리케이션 사이에 안전한 데이터 전송을 위한 실시간 스트리밍 데이터 파이프라인 구축과 데이터 스트림을 변환하거나 이를 받아서 즉시 처리하는 실시간 스트리밍 애플리케이션의 구축을 할 수 있다. 카프카 클러스터는 Topic이라고 부르는 파이프라인에 데이터 레코드 스트림을 저장한다. Producer는 선택한 Topic에 스트림 레코드를 게시한다. Producer는 Topic 내에서 어떤 파티션에 할당 할 것인지 선택해야 하고, 이는 로드 밸런싱(Load Balancing)을 위하여 라운드 로빈(Round-Robin)방식으로 수행되거나 일부 의미적 파티션 함수에 따라 수행될 수 있다. Consumer는 Consumer 그룹 이름을 사용하여 레이블을 지정하고 Topic에 게시된 각 레코드는 구독하는 각 Consumer 그룹 내의 하나의 Consumer인스턴스에 전달된다. Consumer인스턴스는 별도의 프로세스 또는 별도의 시스템에 있을 수 있다. 모든 Consumer인스턴스가 동일한 Consumer그룹을 갖는 경우 레코드는 Consumer인스턴스보다 효과적으로 로드 밸런싱이 된다.



(그림2) Kafka Architecture

2.3 InfluxDB

InfluxDB[5]는 Time-series DB로 시계열 데이터를 저장하고 활용하는데 특화된 DB이다. 시계열 데이터를 위해 특별히 작성된 TSM엔진을 사용하여 높은 수신 속도 및 데이터 압축을 허용하고 다양한 플러그인을 제공한다. 시계열 데이터란 시간의 흐름에 따라 저장하는 데이터로서 서버, DB, Network, Storage와 같은 IT인프라 모니터링을 위한 각종 데이터들, 서비스 반응을 확인하기위한 각종 지표, IoT기기들의 센서 데이터 등 활용 목적에 따라 다양하다. InfluxDB는 이러한 시계열 데이터들을 효율적으로 저장할 목적으로 사용되며 Grafana같은 대시보드 tool과 연계하여 모니터링 용도로 주로 사용된다.

2.4 Docker

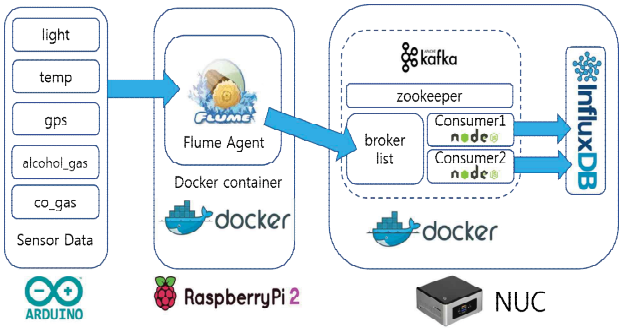
Docker[6]는 container platform 소프트웨어이다. 컨테이너는 운영체제에서 격리된 형식으로 소프트웨어를 패키지화하는 방법이다. VM(Virtual Machine)과 달리 컨테이너는 전체 운영체제를 번들로 제공하지 않으므로 소프트웨어 작업에 필요한 라이브러리 및 설정만 필요하다. 이는 매우 효율적이며 가볍고 독립적인 시스템을 구축하고 배포위치에 관계없이 소프트웨어가 항상 동일하게 실행되도록 보장 할 수 있고 개발환경을 설정하고 구성하는 반복적인 작업을 자동화할 수 있다.

3. 확장성과 자동화를 위한 시스템 설계

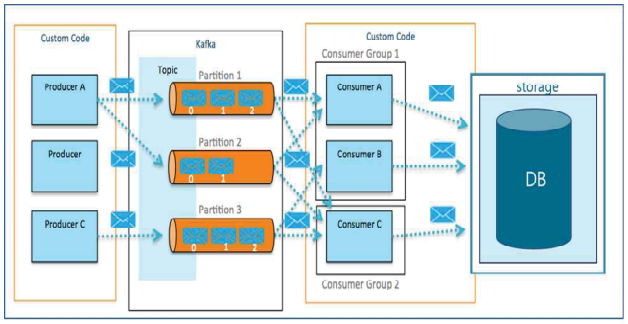
본 논문에서는 많은 IoT기기에서 들어오는 대규모 센서 값들을 수집하기 위해 Flume, Kafka, InfluxDB, nodeJS, Docker를 이용하여 설계하고자 한다. 시스템을 설계할 때 확장성과 자동화를 중점에 맞추어 설계하였다.

많은 센서 값들을 관리하기 위해 로그 수집기로 Flume을 이용하였는데 Flume의 구조는 그림1과 같다. Flume Agent는 다양한 빈도로 들어오는 센서 값들을 Collector쪽으로 전달을 할 때 무결성을 보장해야한다. 무결성을 보장 할 때 센싱 하는 단계에서 할 수도 있고, Agent 단계에서 할 수도 있다. 현재 본 논문의 설계에서는 Agent 단계에서 파일 스왑(File Swap)을 통하여 진행하였다.

확장성을 위해서라면 많은 Agent 즉 IoT 센서 기기들이 계속 추가되어도 Collector 계층에서는 많은 수의 에이전트 노드에서 오는 많은 양의 데이터를 처리할 수 있도록 확장 할 수 있어야



(그림3) 시스템 설계도 및 구성도



(그림5) 시스템 흐름도

no	name	type	range	error value	m/o	desc.
1	device_id	string			m	디바이스의 고유 ID
2	status	integer	[0,1,2]		m	0: Success 1: Sensing Error (일부에서 에러나 누락 발생시에도 1로 표시하고, 해당 센서 측정치에 error value 설정) 2: Comm. Error
3	time	datetime			m	저장된 시간
4	lat	float	-90.0~90.0		m	위도
5	lon	float	-180.0~180.0		m	경도
6	alcohol_gas	integer	0~1023	-1	o	알코올 가스량 (ppm)
7	co_gas	integer	0~1023	-1	o	일산화탄소량 (ppm)
8	temp	float	-40.0~125.0	-999	o	온도
9	light	integer	0~1023	-1	o	조도
10	dust	float	0.0~750.0		o	먼지(ug/m3)

(그림4) 센서 Metric


```
root@KOREN-DEV:/home/koren# curl -G 'http://210.114.90.176:8086/query?pr
etty=true' --data-urlencode "db=station" --data-urlencode "q=SELECT * FR
OM JNU_rasp"
{
    "results": [
        {
            "statement_id": 0,
            "series": [
                {
                    "name": "JNU_rasp",
                    "columns": [
                        "time",
                        "alcohol_gas",
                        "co_gas",
                        "dust",
                        "id",
                        "latitude",
                        "light",
                        "longitude",
                        "temp"
                    ],
                    "values": [
                        [
                            "2017-09-18T08:52:56.794724369Z",
                            9,
                            117,
                            0.05,
                            "RaspberryPi",
                            35.17959,
                            554,
                            126.90822,
                            26.52
                        ],
```


(그림6) InfluxDB Data 확인

하고 파티션이 가능하며 병렬처리가 가능해야한다. Messaging System인 Kafka가 Flume에서 Collector 계층에 해당한다. Kafka의 broker는 topic을 기준으로 메시지를 관리한다. Producer는 특정 topic의 메시지를 생성한 뒤 해당 메시지를 broker에 전달한다. Broker가 전달받은 메시지를 topic별로 분류하여 쌓아놓으면, 해당 topic을 구독하는 consumer들이 메시지를 가져가서 처리하게 된다. consumer는 Apache Kafka에서 제공하는 JavaScript 클라이언트를 통하여 구현 할 수 있다. consumer는 구독한 값을 InfluxDB기반으로 만들어진 객체에 값을 보낸다. 이후 이 객체에서 InfluxDB에 값을 저장한다. 다수의 센서들이 들어오게 되면 Topic별로 이를 분류하는 체계가 필요하다. Topic의 구성에 따라 DB의 설계 또한 달라지게된다., 본 논문에서는 Topic Name을 [Site_index_type]의 형태로 제안하였다. 기존 Sensor별 Topic Name이 아니라 Site별 Topic Name으로 함으로써 분산되고 넓은 지역에서 오는 다양하고 많은 센서들을 관리하기 용이해진다.

Kafka는 확장성과 높은 가용성을 위하여 broker들이 클러스터로 구성되어 동작하도록 설계되어있다. broker가 1개 밖에 없을 때에도 클러스터로써 동작을 하며, 클러스터 내의 broker에 대한 분산 처리는 그림2와 같이 ZooKeeper가 담당한다.

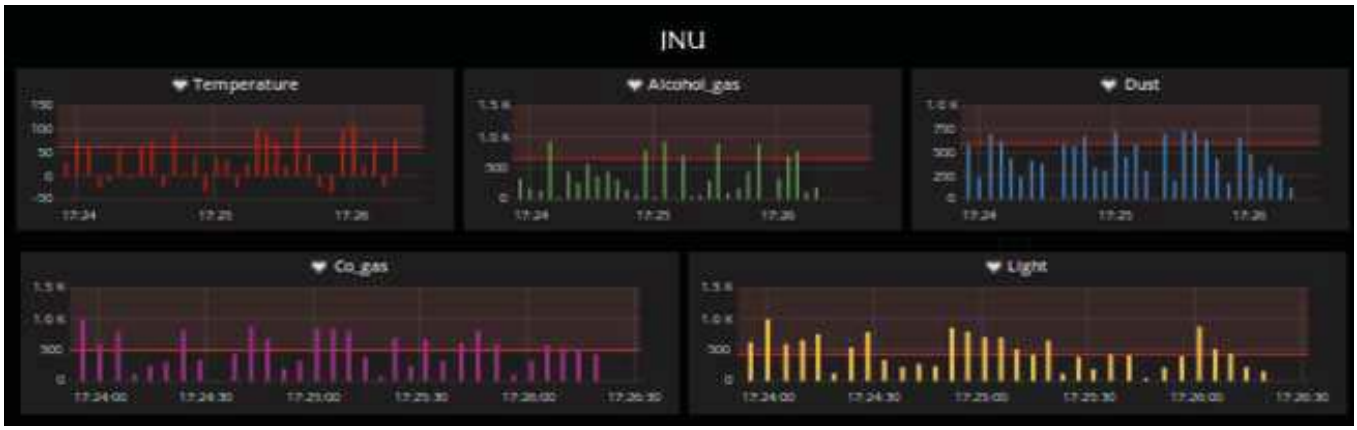
이로써 Flume과 Kafka를 이용하여 확장성을 보장하고, 수많은 센서 값들을 쉽게 저장할 수 있는 InfluxDB와 Kafka간 연동과정 자동화를 위하여 Docker Container를 이용하였다. Container에서 시스템을 구현하고 설정을 끝마친 후 commit을 하면 이미지파일이 생성되고 이후 생성된 이미지파일을 이용하여 빌드 하면 commit한 container상태와 똑같은 container가 만들어진다. 새로운 IoT기기를 추가하여 Flume Agent를 다시 만들어야 할 때 Docker image를 이용하여 build만 시켜주면 자동적으로 Agent가 추가가 된다. Kafka broker와 consumer도 이와 같은 방식으로 추가 할 수 있기 때문에 시스템 구성 자동화를 보장한다. 이를 이용한 시스템 설계도는 그림3과 같다.

4. 구현 및 검증

본 논문에서는 앞서 설계한 시스템을 가지고 컨테이너 기반 대규모 IoT 센서 데이터 수집 서비스위한 시스템을 구현 해 보았다. 시스템 구성도는 그림3와 같다.

아두이노에서 사용한 센서 메트릭(Metric)은 그림4와 같다. 조도, 온도, gps, 알코올가스, 일산화탄소가스 센서를 통해 수집한 값들을 Serial Communication을 통하여 라즈베리파이 기기로 보낸다. 이후 라즈베리파이에서는 python으로 작성된 간단한 코드를 통해 Serial 통신으로 온 센서 값들을 파일로 기록하여 저장하고 이후 Flume Agent가 이를 읽어 오게 된다. Flume Agent는 Source에 새로운 값이 있을시 이를 읽고 Intel NUC에 있는 broker list중 하나의 broker를 통해 값을 바로 전달한다. 즉 센서 값이 들어오는 속도가 Flume에서 Kafka로 보내는 속도가 된다. 이후 NUC 내부의 NodeJS로 만들어진 Consumer가 자기가 맡은 Topic에 게시된 데이터들을 InfluxDB에 저장한다. Flume Agent와 Kafka broker, zookeeper, NodeJS[7]로 만든 consumer와 InfluxDB는 Container로 구축하였으며 Docker Container를 사용하였다.

시스템 흐름은 그림5와 같고 IoT기기들이 Producer들이 되고 각각의 Producer들은 Kafka에서 미리 만들어둔 Topic의 해당되는 Partition에 데이터 스트림을 게시하게 된다. broker list중의 한broker를 통하여 Topic 파이프라인에 스트림 레코드가 저장이 되고, 이후 NodeJS로 만들어진 Consumer들이 토픽에 저장된 스트림 레코드를 읽어와 이를 InfluxDB에 저장한다.



(그림7) 실시간 데이터 수집 결과

토픽에 저장된 스트림 레코드는 간단한 명령어를 통해 콘솔로도 확인이 가능하다.

InfluxDB에서 data값을 쓸 때 JSON protocol[8]을 지원하기 Topic에 저장되는 스트림 레코드 값을 JSON형식으로 하였다. DB에 저장된 센서 값을 확인한 결과는 그림6과 같다. 또한 대시보드를 쉽게 만들 수 있는 Tool인 Grafana를 이용하여 실시간으로 들어오는 데이터 들을 가시적으로 볼 수 있게 화면을 구성해보았다. 온도, 알코올 가스, 먼지, 일산화탄소 가스, 조도 센서 데이터를 확인한 결과는 그림7과 같다.

5. 결론

본 논문에서는 Flume, Kafka, InfluxDB, Docker Container를 이용하여 확장성과 자동화를 보장하는 컨테이너 기반 대규모 IoT 센서 데이터 수집을 위한 서비스를 설계하고, 해당 시스템의 초기모델을 구현하여 그 동작을 검증하였다. 향후 Koren Playground에 적용하여 확장 테스트를 수행할 계획이다.

감사의 글

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

참 고 문 헌

[1] 배상태, 김진경 “사물인터넷(IoT) 발전과 보안의 패러다임 변화”, 『KISTEP Inl』 14호, 2016년 6월 pp44-pp48
[2] Apache Flume, 2017 (https://flume.apache.org/)
[3] “Flume User Guide - Cloudera” (http://archive.cloudera.com/cdh/3/flume/UserGuide/index.html#_architecture)
[4] Apache Kafka, 2017 (https://kafka.apache.org/)
[5] InfluxDB, “Time Series Database Monitoring & Analytics” (https://www.influxdata.com/)
[6] Docker, 2017 (https://www.docker.com/)
[7] NodeJS, 2017 (https://nodejs.org/)
[8] JSON (http://www.json.org/)