K-ONE 기술 문서 #15

Cloud 환경 기반 LoRaWAN provisioning
: Controller Application Server

Document No. K-ONE #15
Version 0.1
Date 2016-05-01

Author(s) 김홍석

K-ONE

ONK
OpenNetworkingKorea

■ 문서의 연혁

버전	날짜	작성자	비고
초안 - 0.1	2016. 03. 03	김홍석	초안 작성
0.2	2016. 03. 11	김홍석	그림 추가
0.3	2016. 03. 22	김홍석	내용 추가
0.4	2016. 04. 10	김홍석	내용 수정
1.0	2016. 04. 23	김홍석	최종 점검(오타, 형식)

본 문서는 2015년도 정부(미래창조과학부)의 재원으로 정보통신 기술진흥센터의 지원을 받아 수행된 연구임 (No. B0190-15-2012, 글로벌 SDN/NFV 공개소프트웨어 핵심 모듈/기능 개발)

This work was supported by Institute for Information & communications Technology Promotion(IITP) grant funded by the Korea government(MSIP) (No. B0190-15-2012, Global SDN/NFV OpenSource Software Core Module/Function Development)

기술문서 요약

본 기술문서는 클라우드 기반의 LoRaWAN 시험 환경 설정에 대한 내용을 요약한다. SDN NFV가 접목되는 차세대 네트워킹 환경에서 효율적인 IoT 네트워킹을 위한 연구 개발을 진행하기에 앞서 OpenStack상 시험 환경을 구축하고 관련 핵심 모듈을 구현한다. OpenStack을 활용하여 LoRaWAN 인스턴스들의 가상화를 수행하며, Horizon 인터페이스 확장을 통해 구성 요소들의 Provisioning을 제공 및 활용 방법에 대해 기술한다. 구축된 시험 환경에서 LoRaWAN 인스턴스 통합 네트워킹 관리를 위한 Controller와 Application Server에 대한 구현 내용 및 기능을 서술한다.

IoT를 위한 프로토콜은 장거리, 저원가, 저전력의 특징을 지니는 LoRaWAN을 연구 분석하고 구성요소를 구현, OpenStack상에 구축하였다. 단말을 의미하는 End Node와 Concentrator/Gateway간에는 LoRa 변조 기술을 사용하여 무선 구간 통신이 이루어지며, Concentrator는 수집된 데이터를 Network Server로 전송한다. Network Server는 Application EUI에 따라 데이터를 해당하는 Application Server로 전송한다. End Node와 Concentrator는 오픈소스를 활용하였고 소스코드가 제공되지 않는 Network Server와 Application Server를 구현하여 OpenStack 환경에서 가상화 되어 구동되도록 하였다. 추가적으로 싱글 노드의 OpenStack상 LoRaWAN 인스턴스들과 Concentrator간의 네트워킹을 지정해주는 Controller를 구현하였다.

OpenStack의 Dashboard 프로젝트인 Horizon을 확장하여 LoRaWAN 시험 환경에 대한 Provisioning을 수행 하였다. 이에 따라, 사용자로부터 손쉽게 LoRaWAN 시험 환경 구동을 위한 정보를 Dashboard를 통해 수집하여 등록된 절차에 따라 자동 구동되도록 설정하였다.

Contents

K-ONE #1. K-ONE 기술문서 템플릿

그림 목차

K-ONE #1. K-ONE 기술문서 템플릿

1. LoRaWAN 개요

1.1. 개요

o 종래의 사물인터넷(Internet of Things)은 사물간의 연결의 속성보다는 대부분이 사물 자체에 초점을 맞추어 다양한 서비스 제품이 나오고 있다. 하지만 사물인터넷의 가장 큰 특징은 원격으로부터 데이터를 수집하여 이를 응용하는 서비스 어플리케이션이 매우 다양하다는 것이다. 예를 들어, 센서 장치가 교량이나 항만에 위치하여 진동 혹은 결함을 측정하며 기상 예측이나 농업 분야 등 그 사용성이 매우 다양하다. 이에 따라 사물인터넷 장치 간 연결이 급증하고 있으며, Gartner에서는 2020년까지 250억 가량의 장치가 네트워크로 연결될 것이라고 예측한다. 사물인터넷 서비스와 데이터가 고도화됨에 따라 서비스 제공자의 초점은 사물보다는 사물간의 연결 즉, 네트워킹 측면에서의 접근이 중요하다.

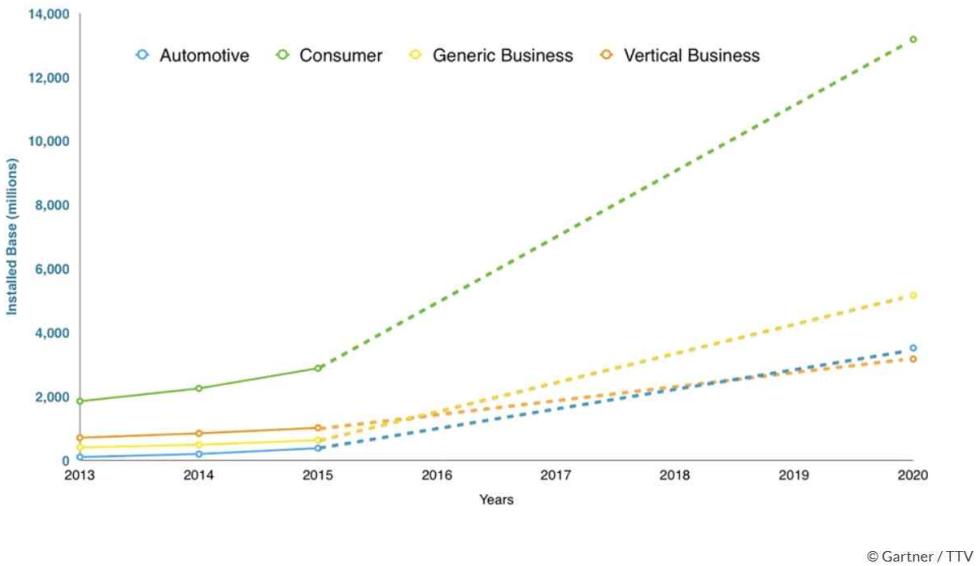

그림 1 소비 진영에서의 사물 간 연결 예상 그래프

o 사물인터넷을 구성하는 단말/센서와 같은 End Node는 크게 유선 선로를 통해 인터넷에 연결되는 단말과 무선 선로를 통해 연결되는 단말로 분류될 수 있다. 무선 구간의 전송 길이가 짧은 경우 Zigbee, Bluetooth, WiFi 기술 등이 일반적으로 사용되며, 무선 구간의 길이가 긴 경우에는 LTE 망과 같은 통신망을 이용한다. 하지만 LAN(Local-Area Network) 혹은 Cellular Network에서 멀티미디어 콘텐츠 소비를 지원함에 따라, 대량의 데이터를 효과적으로 전송하기 위한 특성으로 기지국 설치비용이 증가하고 신호 송수신 거리가 짧아지며 복잡한 동작에 의한 소비전력 증가와 배터리 라이프타임이 짧아지는 것과는 달리, 사물인터넷의

요구사항은 이와 반대의 특성을 지닌다.

o 사물인터넷은 매우 낮은 데이터 전송률로 저전력의 장시간 배터리 유지와 장거리 무선 신호 도달범위 확보가 핵심 이슈다. 따라서 사물인터넷 사업자는 제한된 자원으로 인프라를 구축해야하는 과제에 직면하게 되었다. 다양한 사물인터넷 서비스 형태와 장비가 출시되면서 사물들이 연결되는 인프라 구축의 필요성이 증가되었고, 이러한 사물들 간의 연결을 지원하는 방식이 바로 LPWAN(Low-Power Wide-Area Network) / LPN(Low-Power Network) 이다.

o LPWAN 기술은 배터리를 사용하여 운용되는 센서와 같은 사물들 간 낮은 데이터 전송률의 장거리 무선 통신 네트워크를 디자인한다. 예를 들어, Channel bandwidth를 매우 작게 하고 비면허 대역에서도 낮은 주파수를 사용하며 End Node의 신호 수신 슬롯의 주기를 조정하여 배터리 수명을 높이는 등 낮은 소비전력으로 전파 침투 범위를 넓히기 위한 방안을 제시한다.

o 대기상태에서 사물인터넷 단말은 대부분의 시간을 비활성화 상태에서 보내며, 주기적으로 활성화 상태로 전환하여 기지국으로부터 전송되는 데이터의 여부를 확인한다. 또한, 신호 송신의 목적지 기지국을 지정하지 않고 데이터를 전송하기 때문에 기지국을 찾는 동작에 소모되는 전력이 없다. End node는 저렴할수록 많은 기기에 적용이 가능하기 때문에 모뎀 칩이나 망 사용료가 저렴해지며, 3G 및 4G Cellular network에 비해 Embedded application의 더 높은 확장 가능성과 낮은 비용 및 전력으로 보다 넓은 커버리지를 확보하게 된다.

o 사물인터넷 LPWAN 망 표준화 경쟁 분야의 대표적인 기술은 프랑스 시그폭스(Sigfox)사의 UNB(Ultra Narrow Band)와 Semtech, IBM Research 연합 Lora Alliance의 LoRaWAN 기술 등이 있다. 시그폭스의 UNB 기술은 별도의 기지국 또는 중계 장비 없이 다양한 사물 간 가까운 거리에서 데이터를 교환할 수 있도록 하여 망 구축비용을 최소화 한다. LoRaWAN 기술에서는 Long range, Max lifetime, Multi-usage, 그리고 Low cost의 4가지 특징 요소를 목표로 한다. 현재 시장 초기단계의 LoRaWAN 사물인터넷 기술과 클라우드 환경에서 소프트웨어 정의 네트워크(Software-defined Network)기술 접목을 통한 성능향상을 목표로 연구 개발을 진행한다.

1.2. LoRa

o LoRaWAN은 사물인터넷 통신의 요구 특성에 따라 LoRa라는 무선 변조 기술을 이용한다. LoRa는 장거리에 걸쳐 저전력의 신호 송수신이 가능하게 하는 RF 인터페이스/Physical layer를 포함하는 기술이다.

o LoRa는 Chirp Spread Spectrum 변조 방식을 사용하며, 제한된 Channel Bandwidth에서 낮은 데이터 전송률 대신 신호 송수신 거리를 확보한다. Chirp Spread Spectrum 변조 방식은 시간이 지남에 따라 선형적으로 주파수가 증가/감소하는 Chirp 펄스로 주파수를 변조, 광대역에 걸쳐 정보를 인코딩한다. 보내고자 하는 원 신호에 확산 코드를 곱하면 신호의 에너지는 낮고 넓게 퍼지며, 노이즈는 확산되지 않은 채로 수신된다.

o 수신기는 원 신호로의 복원을 위해 다시 역확산 코드를 곱하게 되며, 노이즈에는 확산되는 효과가 발생하여 수신기로부터 무시됨에 따라 Coding Gain이 발생한다. 또한, 서로 직교하는 확산 대역 코드를 사용하여 동시에 여러 단말이 통신할 수 있는 Coexistance 측면의 이점이 있다.

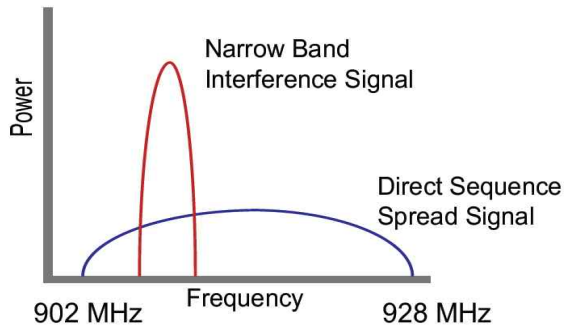

그림 2 확산 대역 코드와 간섭 신호

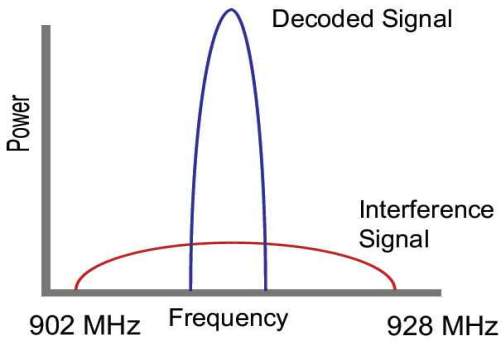

그림 3 디코딩된 원 신호와 간섭 신호

o 현재 사용되고 있는 무선 통신 프로토콜들은 저전력 기반으로 모듈의 공급가가 매우 저렴하지만 주로 단거리 기반의 서비스를 위한 기술로 이용되고 있다. 따라서 서비스의 범위 확장 시, 중계 장비의 개수 증가에 따라 많은 비용이 발생한다. 하지만 LoRa의 경우 가시거리가 확보된 환경에서는 최대 21Km의 통신 범위를 목표로 하기 때문에 많은 중계 장비가 불필요하며, 낮은 인프라 구축비용으로 중장거리 무선통신이 가능하다. 주파수 밴드는 2.4GHz, 5.8GHz 대역의 ISM(Industrial Scientific Medical) band 보다 낮은 868MHz, 915MHz, 433MHz 등으로 국제적으로 가용한 비면허 대역을 선택적으로 사용 가능하다.

1.3. LoRaWAN

o LoRaWAN은 물리 계층에 대한 신호 변조 기술인 LoRa의 상위 MAC 계층과 통신 규약에 대한 정의 포함하는 기술이다. LoRaWAN 프로토콜 클래스와 각 클래스별 통신 규격 및 MAC 메시지 포맷 등이 표준 문서에 정의되어 있다.

o 넓은 신호 송수신 범위와 낮은 전력 소비량을 극대화 하면서 수백만 개의 무선 센서 노드를 게이트웨이에 연결 할 수 있기 때문에 인프라 구축비용이 낮고 실외 환경에서도 안정적으로 작동함에 따라 광범위한 저속 무선 모니터링 및 제어 설계에 매우 적합하다고 설명되고 있다. 10년 이상 지속되는 배터리 수명으로

o 현재 LoRaWAN 표준에서는 단말의 수신 슬롯의 스케줄링 기능과 관련된 세가지 Class A, B, C 를 정의하고 있다. 각각의 Class들은 LoRaWAN 패킷의 수신 슬롯의 활성화 주기에 따라 나뉜다. 모든 LoRaWAN 단말은 최소 Class A에 정의된 기능을 지원해야 하며, Class B 와 Class C 기능을 부가적으로 지원하도록 설정할 수 있다.

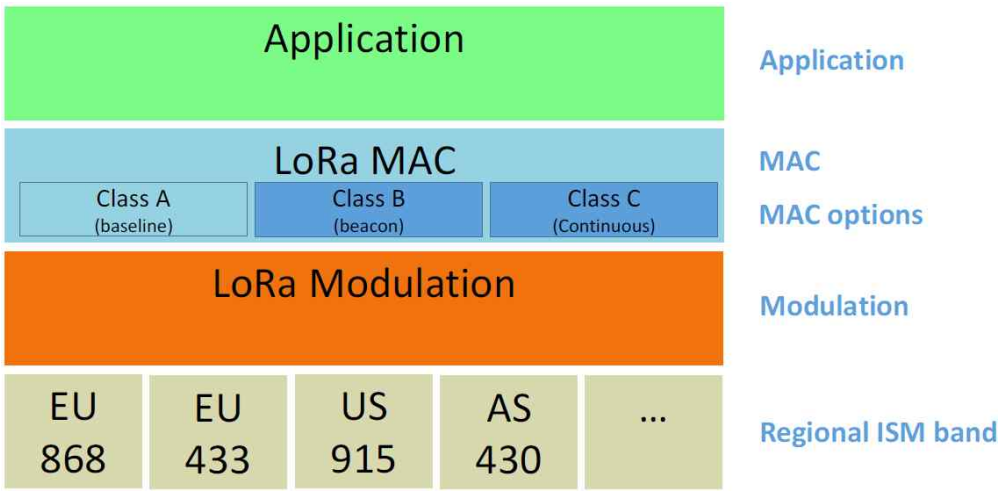

그림 4 LoRaWAN Class와 계층 구조

o LoRaWAN 네트워크 구성요소는 End Device와 Gateway, Network Server와 Application Server가 존재한다. 기본적으로 Star topology 형태의 구성으로 End device의 메시지를 Gateway가 Network Server로 전달하는 구조이다. 무선 구간에서의 End Device와 Gateway간 각각의 통신은 서로 다른 주파수 채널과 데이터 전송률을 사용하여 간섭을 줄이며 송수신 거리를 확보한다. 배터리 수명의 극대화를 위해 데이터 전송률과 RF 출력을 선택적으로 조정하는 ADR(Adaptive Data Rate) 방식을 사용한다.

1.4. 클라우드 환경의 LoRaWAN 구조

o SDI(Software Defined Infrastructure) 환경에서의 차세대 네트워크 구조에서는 LoRaWAN 구성 요소의 Network Server와 Application Server의 가상화 및 SDN

연동 구조를 통해 기능 향상이 이루어질 수 있다. 이를 위한 첫 번째 단계로 OpenStack 상 LoRaWAN 시험 환경을 설정하고 요구사항을 분석한다.

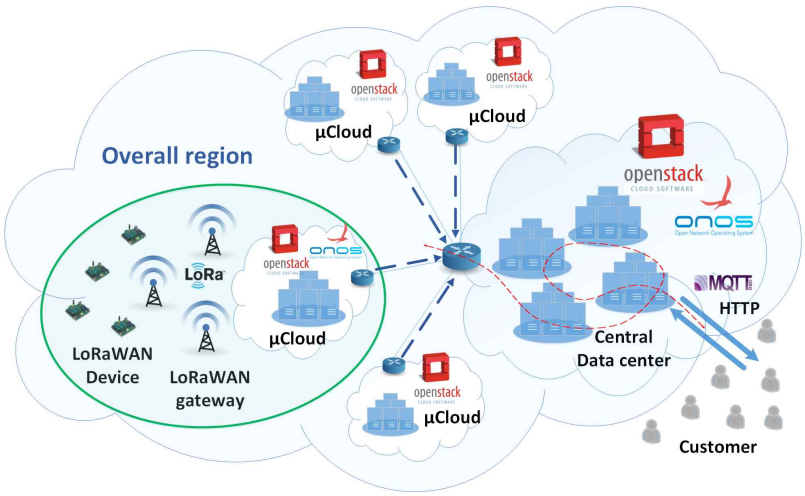

그림 5 클라우드 환경에서의 LoRaWAN 구성

2. 클라우드 환경의 LoRaWAN 테스트베드 구성 및 활용법

2.1. OpenStack Horizon

o OpenStack은 devstack을 이용하여 Liberty 버전을 설치하였고, 제공되는 Horizon development tool을 사용하여 LoRaWAN을 위한 구성요소 생성 및 수정 기능이 자동 수행되도록 하였다. Devstack은 OpenStack 개발 환경을 간편하게 설치할 수 있도록 제공되는 프로젝트이다. 사용자는 devstack을 이용하여 손쉽게 Openstack을 설치, 필요한 환경을 설정할 수 있다. Devstack 폴더에 설정파일인 local.conf 내용을 작성하여 stack.sh를 실행하면 OpenStack Ceilometer, Heat, Horizon, Keystone, Nova, Neutron 등의 구성 요소들을 연동하여 설치 및 구동이 가능하다.

o OpenStack dashboard 서비스인 Horizon은 관리자나 사용자가 서비스를 이용할 수 있도록 제공되는 Web기반 인터페이스로 VM 인스턴스의 생성과 삭제, 관리, 접속, 재부팅 등의 기능을 지원한다. Horizon dashboard는 Apache Web Server를 사용하며, python기반의 web application framework인 Django와 연동되어 동작한다. Horizon 모듈은 다른 모든 서비스의 API와 연동되어 사용자에게 web 서비스를 제공할 수 있으며, 새로운 버전이 릴리즈 될 때마다 기능이 확장되고 있다.

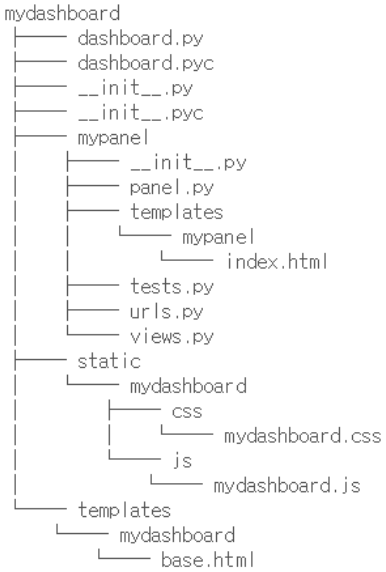

그림 6 Dashboard 소스 구조

o 기본적으로 Horizon의 main UI 요소인 Panel과 Tab, user input을 일련의 단계에 따라 수집하는 Workflows, 그리고 Action 등을 수정 및 확장할 수 있다. Devstack을 이용한 OpenStack 설치 후 제공되는 development server를 임의의 port로 실행하면, /opt/stack/horizon 디렉토리 내 추가 및 변경된 python 코드 내용이 별도의 unstack, stack을 통한 프로세스 재시작 과정 없이도 Horizon dashboard에 반영된다. Default port는 8000이며, 이 port는 이미 devstack의 heat-api-cfn에 의해 사용되기 때문에 별도의 port로 development server를 구동할 수 있도록 제공되는 run_test.sh를 실행한다.

```
#./run_tests.sh --runserver 0.0.0.0:8877
```

그림 7 Development server 구동

o Horizon 인터페이스 상 LoRaWAN 테스트베드 구축 및 수정 시 입력받는 정보를 Workflows에서 수집하여, Heat 모듈에 전달 할 template을 생성/수정하며, CLI 호출을 통해 인스턴스가 생성된다. 또한, 생성된 인스턴스들은 Heat template에 등록된 각각의 역할에 따라 생성과 동시에 구동되며, 네트워킹을 위한 정보를 요청받게 된다.

o Horizon Workflows에는 LoRaWAN 테스트베드 구축 시 수집되는 정보들을 토대로 수행될 Action Class들을 위와 같이 정의하였고, CreateLoRaWANNetwork Class에 등록된 절차에 따라 수행된다.

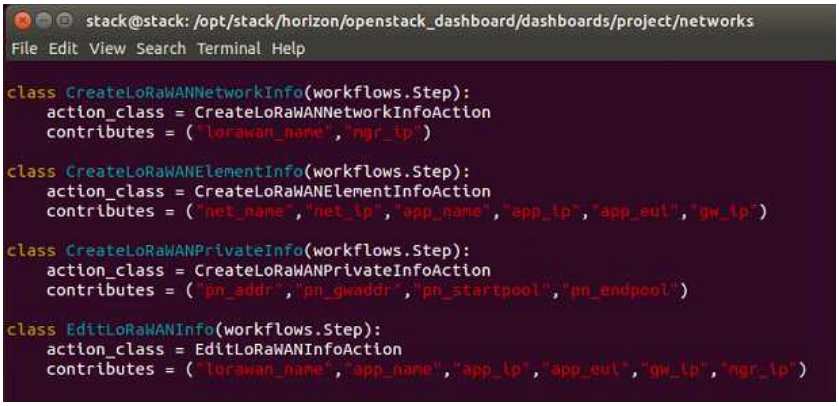

그림 8 LoRaWAN Action Class 정의

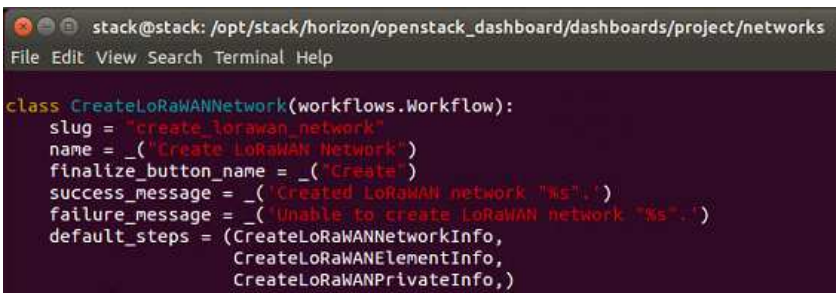

그림 9 LoRaWAN 생성을 위한 Class와 Action Step

2.2. OpenStack 기반 LoRaWAN 테스트베드 구성

o 과제 수행 중 첫 단계로 장거리, 저원가, 저전력의 특징을 지니는 LoRaWAN에 대해 표준을 제정하는 LoRa Alliance의 White Paper를 참고하여 구성 요소들을 구현하였고 시험 환경을 OpenStack상 구축하였다. LoRaWAN 망의 구조는 크게 4가지 요소들로 구성되며, 단말장치를 의미하는 End Node, 단말장치와 통신을 위해 무선 데이터의 송수신을 담당하는 Gateway/Concentrator, 데이터를 소비하는 Application Server, 그리고 Gateway/Concentrator와 Application Server 사이 데이터 중계를 담당, 무선통신 구간을 관리하는 Network Server가 있다. 시험 환경 구축을 위해 End Node와 Gateway/Concentrator의 경우 Semtech와 Kerlink에서 공개한 오픈소스를 수정 활용하였고, Network Server와 Application Server는 관련 오픈소스가 없으므로 LoRaWAN 표준에 따라 직접 구현 하였다.

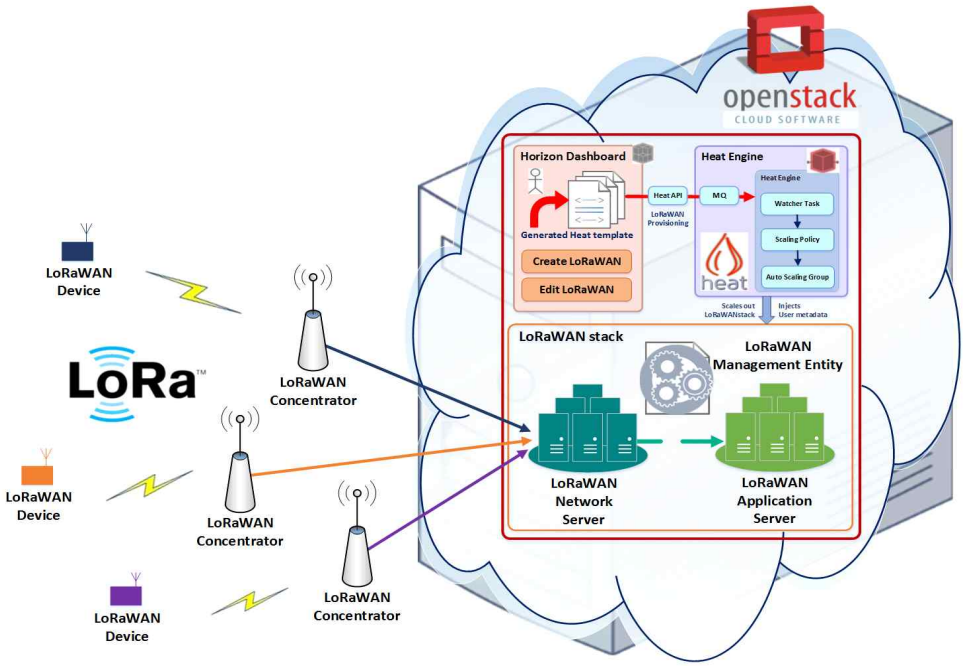

그림 10 LoRaWAN 테스트베드 구성 및 OpenStack 구조

o 추가적으로 OpenStack 상 생성되는 LoRaWAN 인스턴스들간 네트워킹을 통합 관리 및 제어하기 위한 Controller를 구현하였다. Gateway/Concentrator는 단말로부터 수집되는 데이터를 포워딩 할 목적지 정보를 Controller로부터 지정받게 되며, Network Server는 Controller에 의해 등록된 Gateway List에 이외의 Concentrator는 수용하지 않는다. 또한, 등록된 Device EUI와 Application EUI에 해당하는 Application Server로만 데이터를 전송하기 때문에 전반적인 IoT 네트워킹 관리 및 Access Control이 이루어진다. IoT 서비스 중심의 Access Control과 더불어 구성 요소들 간 연결을 Controller에서 통합 관리하게 되며, 제어를 위한 IoT Instance Map을 구축하여 구성 변경에 따른 이벤트 발생에 대해 상태정보를 동적으로 반영한다.

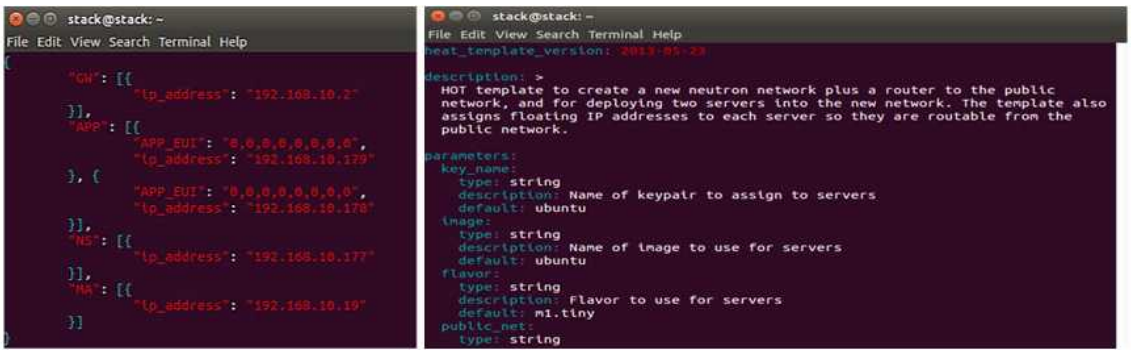

그림 11 JSON Instance Map 및 생성된 Heat template

2.3. OpenStack Horizon 확장을 통한 LoRaWAN Provisioning

o OpenStack Horizon Dashboard 확장을 통해 LoRaWAN 망 생성과 수정에 대한 기능을 구현하였다. LoRaWAN 망의 기본적인 구성요소로 Network Server와 Application Server, 그리고 인스턴스 간 네트워킹을 관리하는 Controller를 구현하였으며, 사전에 정의 된 절차와 Script에 따라 자동 Provisioning 되도록 하였다.

o LoRaWAN Network Server, Application Server, 그리고 Controller가 OpenStack 상의 인스턴스로써 구동되며 Heat template를 활용하여 Network 생성, VM Provisioning 등이 자동화 되도록 한다. 이에 따라 LoRaWAN에 필요한 template 생성 및 실행을 OpenStack GUI를 이용, 인스턴스의 Scale-up, Scale-down 수행이 가능한 시험환경을 구축하였고 LoRaWAN Application 개발자들에게 제공될 수 있는 서비스 형태인 PaaS(Platform as a Service)를 확장한다.

o OpenStack Web Dashboard상 Create LoRaWAN Network를 통해 생성 할 LoRaWAN 망의 이름과 Controller Node IP주소를 입력 받고, Network Server, Application Server의 이름과 IP주소, 그리고 Private Network의 Address Allocation Pool을 입력 받도록 구현하였다. 또한, Controller는 Gateway IP주소와 Application EUI를 참조하여 서비스 기반의 Access Control이 수행 되도록 하였다. 필요한 정보들을 사용자로부터 입력 받은 후 생성된 Script를 Heat Engine으로 전달하여 LoRaWAN 인스턴스의 자동 생성 및 네트워킹이 가능한 테스트베드가 구축된다.

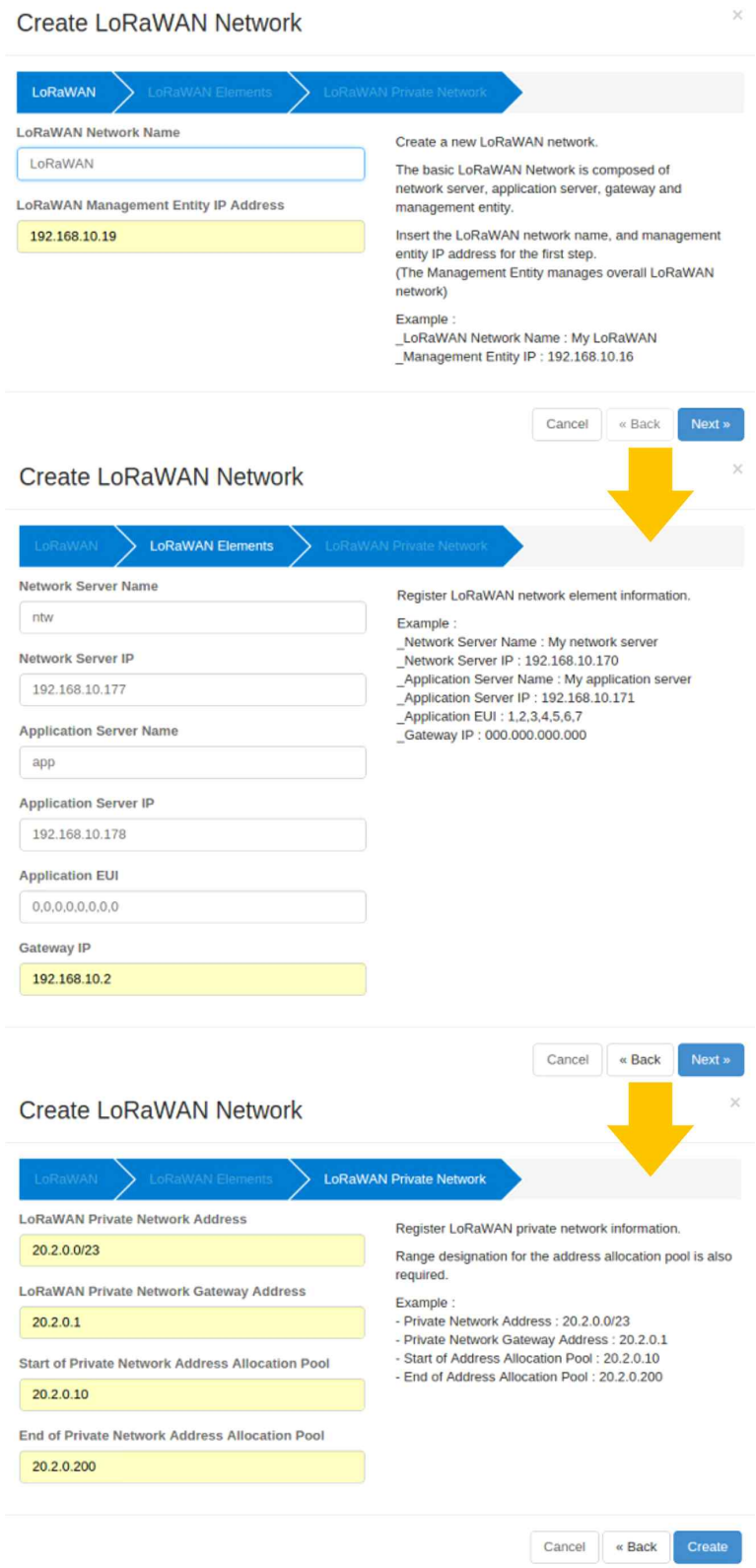

그림 12 Horizon을 통한 LoRaWAN 생성 절차

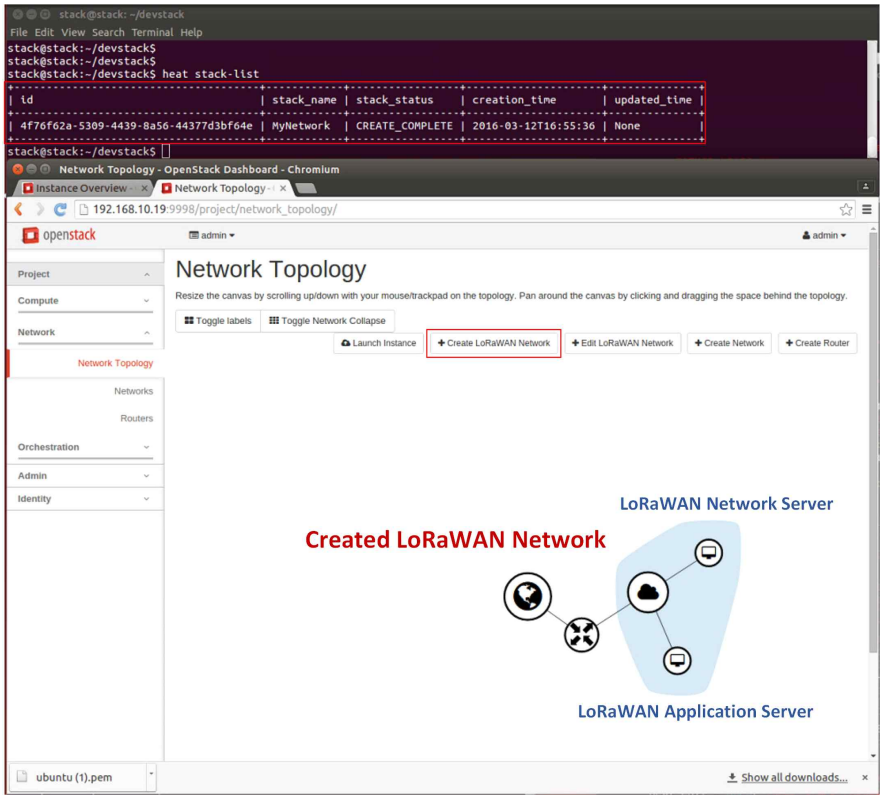

그림 13 생성된 LoRaWAN 인스턴스 및 스택

o Edit LoRaWAN Network를 통해서 구축한 LoRaWAN 구성을 변경 혹은 추가 할 수 있다. Application Server 추가 생성이 가능하며, 기존 등록된 Application EUI를 재사용하도록 증설하거나 새로운 Application EUI를 사용하도록 할 수도 있다. Gateway/Concentrator 또한 증설이 가능하며, 추가 증설하는 LoRaWAN 인스턴스를 관리 할 Controller 정보 또한 입력받아 관리한다.

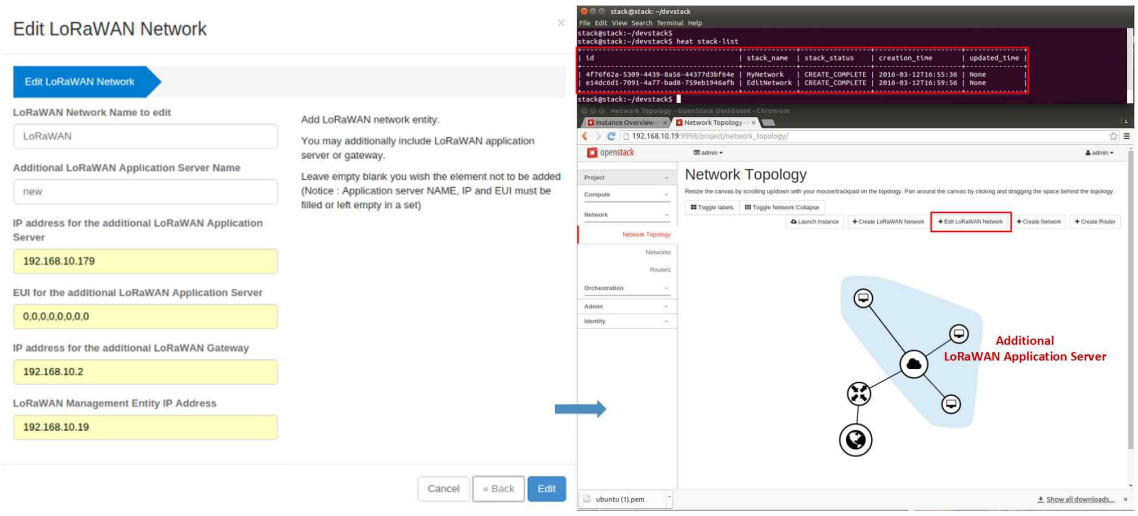

그림 14 LoRaWAN Application Server 추가 절차 및 결과

2.4. OpenStack 상 LoRaWAN 테스트베드 활용 방안

o 앞으로 진행 될 연구 개발의 목적은 Cloud 환경에 구축되는 Scalable IoT Network와 인스턴스간 네트워킹을 통합 관리 및 효과적으로 제어하여 SDN, NFV 기반의 IoT 서비스 가용성을 향상시키는 것이다. 이에 필요한 시험환경으로 OpenStack상에는 가상화된 IoT 인스턴스들과 이들의 네트워킹을 관리하는 Controller가 존재하며, OpenFlow Agent가 연동된 Concentrator가 ONOS SDN Controller로부터 제어되는 환경이 구축되어야 한다.

3. LoRaWAN Controller

3.1. LoRaWAN Controller 설계 및 구현

o LoRaWAN 시험 환경에서 End Device의 데이터는 Gateway/Concentrator에서 수집되어 Network Server로 포워딩된다. 이때, 중복되는 다수의 Concentrator coverage에 공통적으로 포함되는 End Device의 데이터는 중복되어 Network Server로 전달된다. 또한, Network Server는 Application EUI를 통해 Application Server를 구분하여 데이터를 포워딩할 수 있어야 한다. 따라서 LoRaWAN 인스턴스간 네트워킹을 위해 정보를 통합 관리하고 인스턴스간 연결을 중계 및 제어할 수 있는 요소가 필요하다.

3.2. LoRaWAN Controller 기능

o OpenStack상 구동되는 LoRaWAN Controller는 LoRaWAN 표준에서 정의된 구성요소가 아닌 Cloud환경에서의 LoRaWAN 인프라 내 인스턴스간 네트워킹을 통합 관리하기 위하여 구현한 Management entity이다. Horizon workflows를 통해 사용자로부터 수집되는 정보를 토대로 이를 통합하여 Management Table을 구축하고, LoRaWAN 망 내에서 IoT 서비스 기반의 Access Control과 인스턴스간 연결을 중계하고 Management Table에 동적으로 상태 변화를 반영하는 역할로 Controller를 구현하였다.

o LoRaWAN 인스턴스는 생성과 동시에 Controller에 연결할 인스턴스 정보를 요청하고 Controller는 이 table을 참조하여 이에 응답한다. LoRaWAN Network Server는 사전에 등록된 Gateway/Concentrator를 수용, 데이터를 수신하여 해당하는 Application EUI로 등록된 Application Server로 최종 데이터를 포워딩한다.

3.3. LoRaWAN Controller 운용 및 검증

o OpenStack Dashboard에서 LoRaWAN 시험 환경을 구축하고 Controller로부터 전달된 정보에 따라 End Device의 데이터를 수집한 Gateway/Concentrator는 Network Server로 연결, Network Server는 다시 Controller에 의해 지정받은 Application EUI에 해당 받은 Application Server로 데이터를 최종 전달하는 동작을 확인하였다.

4. LoRaWAN Application Server

4.1. LoRaWAN Application Server 설계 및 구현

o Application Server는 통상 web application을 생성하거나 실행할 수 있도록 환경을 제공하는 software framework로써의 의미와 서비스가 실행되고 client에 서비스를 제공하는 서버의 역할로 다양한 의미를 지닌다. LoRaWAN Application Server는 data consumer로써 수신되는 데이터를 decrypt하여 사용자에게 전달하는 역할을 수행한다. 다수의 Application Server는 electric metering, smoke alarm, 등등의 각 서비스별로 상이한 특정 데이터를 처리하도록 구분될 수 있으며, 사용자로부터의 downlink data 또한 encryption하여 Network Server로 전송하는 역할을 수행한다.

o 시험 환경 상 구현한 LoRaWAN Application Server의 기능은 여타 Application 과 같이 데이터를 수집하여 해당하는 사용자에게 최종 전달하기 위한 목적으로 존재한다. Application Server는 수집된 데이터를 decryption하여 사용자에게 데이터를 전달한다. 추가적으로 Application Server에서는 WAS(Web Application Server)인 Apache Tomcat과 연동을 통해 Application EUI와 Gateway EUI로 구분된 End Device의 데이터와 Concentrator의 location 정보를 사용자에게 browser를 통해 Visibility를 제공하도록 구현하였다.

4.2. LoRaWAN Application Server 기능

o 사용자는 Web Browser를 통해 Application EUI로 구분되는 device별 데이터를 Line Chart와 Table로 확인 할 수 있고, 현재 데이터의 전송에 사용되는 Concentrator의 GPS 정보를 확인할 수 있다. Application Server는 Application EUI로 식별 및 해당되는 데이터만을 Network Server로부터 전달받기 때문에 기존의 Application EUI 데이터를 추가적으로 받도록 증설이 될 수 있으며, 새로운 EUI로 데이터를 받도록 생성도 가능하다.

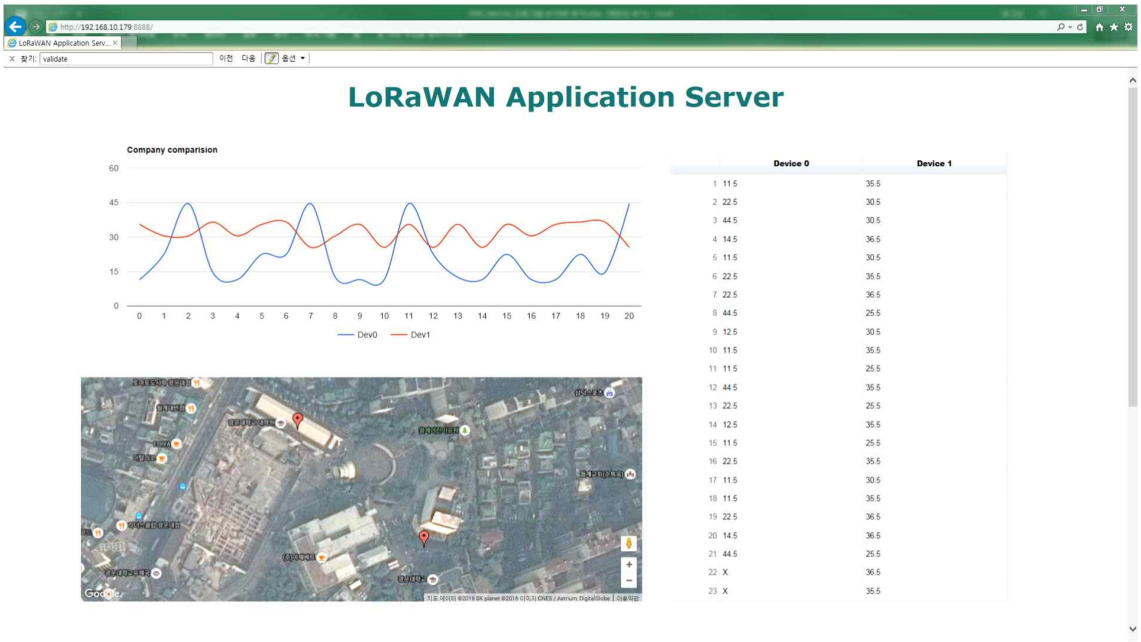

그림 15 LoRaWAN Application Server를 통한 수신 데이터 시각화

4.3. LoRaWAN Application Server 운용 및 검증

o LoRaWAN Network Server에서 수신한 데이터의 Application EUI를 식별하여 해당하는 Application Server로의 데이터 포워딩을 수행함을 확인하였고, 데이터 및 사용한 Concentrator의 GPS 좌표를 시각화 하였다.

o 이를 기반으로 향후 다양한 Application을 접목하여 서비스 환경을 구성 할 수 있으며, OpenStack과 ONOS 통합 환경에서의 LoRaWAN 시험 환경으로 활용할 수 있다.

References

[1] Devstack, http://docs.openstack.org/developer/devstack/
[2] OpenStack, http://openstack.org
[3] Django, https://www.djangoproject.com/
[4] Horizon Quick Start, http://docs.openstack.org/developer/horizon/quickstart.html
[5] Horizon, http://docs.openstack.org/developer/horizon/topics/tutorial.html
[6] Customizing Horizon, http://docs.openstack.org/developer/horizon/topics/customizing.html
[7] LPWAN, https://en.wikipedia.org/wiki/LPWAN
[8] LoRaWAN Specification, https://www.lora-alliance.org/portals/0/specs/LoRaWAN%20Specification%201R0.pdf

K-ONE 기술 문서

- 이 문서의 기술적인 내용은 프로젝트의 진행과 함께 별도의 예고 없이 변경될 수 있습니다.

- 본 문서와 관련된 문의 사항은 아래의 정보를 참조하시길 바랍니다.
  (Homepage: http://opennetworking.kr/projects/k-one-collaboration-project/wiki, E-mail: k1@opennetworking.kr)

작성기관: K-ONE Consortium
작성년월: 2016/04/23