K-ONE 기술 문서 #22

ONOS 상 LISP SBI 지원 및 NETCONF를 통한 관리 방법의 설계 및 구현

Document No. K-ONE #22
Version 1.0
Date 2017-05-10
Author(s) 최준묵, 한윤선

K-ONE

ONK
OpenNetworkingKorea

■ 문서의 연혁

버전	날짜	작성자	내용
초안 - 0.1	2017. 03. 28	최준묵	초안 작성
0.5	2017. 04. 26	최준묵	NETCONF/YANG 내용 추가
0.8	2017. 04. 29	한윤선	설계 내용 추가
1.0	2017. 05. 10	최준묵	오탈자 수정

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

기존의 네트워크를 관리하는 방법으로는 SNMP(Simple Network Management Protocol)이 있었다. 그러나 SNMP는 네트워크 장비를 설정하기보다는 모니터링을 활용하는 데에 주로 사용되었다. 따라서 차세대 네트워크 관리 프로토콜에 대한 요구사항이 발생하였고 이를 충족 시킬 수 있는 새로운 네트워크 관리 프로토콜인 NETCONF(Network Configuration Protocol)가 등장하였다. NETCONF는 XML 형태로 잘 규격화된 메시지를 사용하여 장비와 통신하며 관리 정보를 모델링하기 위해 새로운 데이터 모델링 언어인 YANG를 사용한다.

식별자/위치자 분리 패러다임을 구현한 LISP을 지원하는 장비로는 OpenLISP이나 OOR(Open Overlay Router)이 있다. LISP이 정상적으로 동작하기 위해선 이러한 장비들을 적절히 설정해야 할 필요가 있다. 그러나 현재 LISP 지원 장비를 설정하기 위해서는 수동으로 CLI나 설정 파일 수정을 통해야 한다. 따라서 이를 SDN 컨트롤러 상에서 설정 할 수 있도록 하기 위해 ONOS Management Plane을 설계하고 개발하였다. ONOS Management Plane은 NETCONF/YANG을 사용하여 LISP 지원 장비에 직접 접근하지 않고도 설정을 수정 할 수 있게 해준다.

본 기술문서는 NETCONF와 YANG이 등장한 배경과 개념, 구조에 대해 기술하며 이를 통해 LISP 지원 장비를 ONOS 컨트롤러에서 관리 할 수 있도록 NETCONF/YANG을 사용하는 ONOS Management Plane에 대한 설계를 서술한다. 그리고 이러한 설계를 어떻게 구현했는지에 대해 기술한다.

Contents

K-ONE #22. ONOS 상 LISP SBI 지원 및 NETCONF를 통한 관리 방법의 설계 및 구현

그림 목차

K-ONE #22. ONOS 상 LISP SBI 지원 및 NETCONF를 통한 관리 방법의 설계 및 구현

1. 서론

기존의 네트워크를 관리하는 방법으로는 SNMP(Simple Network Management Protocol)이 있었다. SNMP를 사용하여 관리되는 네트워크는 Managed device, Agent, NMS(Network management station)으로 구성되며 MIB(Management information base)를 통해 정의되는 정보를 사용한다. 그러나 SNMP는 네트워크 장비를 설정하기보다는 모니터링을 활용하는 데에 주로 사용되었다. 따라서 차세대 네트워크 관리 프로토콜인 NETCONF(Network Configuration Protocol)이 등장하였다. NETCONF는 XML 형태로 잘 규격화된 메시지를 사용하여 장비와 통신하며 관리 정보를 모델링하기 위해 새로운 데이터 모델링 언어인 YANG를 사용한다.

식별자/위치자 분리 패러다임을 구현한 LISP을 지원하는 장비로는 OpenLISP이나 OOR(Open Overlay Router)이 있다. LISP이 정상적으로 동작하기 위해선 이러한 장비들을 적절히 설정해야 할 필요가 있다. 그러나 현재 LISP 지원 장비를 설정하기 위해서는 수동으로 CLI나 설정 파일 수정을 통해야 한다. 따라서 이를 SDN 컨트롤러 상에서 설정 할 수 있도록 하기 위해 ONOS Management Plane을 설계하고 개발하였다. ONOS Management Plane은 NETCONF/YANG을 사용하여 LISP 지원 장비에 직접 접근하지 않고도 설정을 수정 할 수 있게 해준다.

본 기술문서는 NETCONF와 YANG에 대해 기술하며 이를 통해 LISP 지원 장비를 ONOS 컨트롤러에서 관리 할 수 있도록 NETCONF/YANG을 사용하는 ONOS Management Plane에 대한 설계 및 구현을 서술한다.

2. NETCONF 프로토콜

본 절에서는 NETCONF(Network Configuration Protocol)이 등장한 배경과 용어, 구조와 명령어에 대해 설명한다.

2.1. NETCONF 프로토콜 개요

기존의 네트워크를 관리하는 방법으로는 SNMP(Simple Network Management Protocol)이 있었다. SNMP를 사용하여 관리되는 네트워크는 Managed device, Agent, NMS(Network management station)으로 구성되며 MIB(Management information base)를 통해 정의되는 정보를 사용한다[1]. 그러나 SNMP는 네트워크 장비를 설정하기보다는 모니터링을 활용하는 데에 주로 사용되었다. 따라서 네트워크 관리자들은 SNMP와 함께 CLI 스크립트를 통해 네트워크를 관리하였다[2]. 그러나 이러한 방식은 SNMP의 데이터 모델링인 MIB의 구조적 한계와 UDP에 기반한 SNMP의 안정성의 문제를 가지고 있다. 이에 IAB(Internet Architecture Board)와 IETF, 그리고 네트워크 오퍼레이터들이 새로운 네트워크 관리 체계에 대해 논의하였다. 그리고 이 논의 내용을 RFC 3535로 문서화하였다.

이 문서에는 다음과 같은 몇 가지 요구 사항들이 있다[3].

- Configuration data, operational data, statistics data 사이의 분명한 구분이 필요하다.
- Configuration data, operational data, statistics data를 개별적으로 얻고 비교할 수 있어야 한다.
- 네트워크 관리자가 개별적인 장치가 아닌 네트워크 전반에 대한 설정에 집중할 수 있어야 한다.
- 시간과 양단 사이의 설정에 대한 일관성 검증이 쉬워야 한다.

이러한 요구 사항을 반영하여 NETCONF의 초안이 RFC 4741 문서로 2006년에 도출되었으며 최근 2011년에 RFC 6241 문서로 개정된 NETCONF 프로토콜이 문서화 되었다.

2.2. NETCONF 용어 정리

- Configuration data

장치의 초기 상태에서 현재 상태로 전환하는데 요구되는 writable data들의 집합을 의미한다.

- Datastore

정보를 저장하고 접근 할 수 있는 개념적인 위치를 의미한다. 이러한 datastore는 파일이나 데이터베이스 등으로 구현 할 수 있다.

- Candidate configuration datastore

Candidate configuration data는 장치의 현재 configuration에 영향을 주지 않으면서 현재 동작 중인 configuration datastore에 commit 될 수 있는 configuration data를 의미한다. 따라서 Candidate configuration datastore는 이러한 data를 관리하는 공간이다.

- Candidate configuration capability

Candidate configuration capability는 현재 장치가 candidate configuration store를 지원함을 나타낸다.

- Capability

NETCONF의 기본 사양을 확장 할 수 있는 추가적인 기능을 의미한다.

- Remote procedure call(RPC)

별도의 원격 제어를 위한 코딩 없이 함수나 프로시저를 실행 할 수 있게 하는 프로세스 간의 통신 기술이다. NETCONF에서는 <rpc>와 <rpc-reply> 메시지를 주고받음으로써 실행된다.

- Session

서버와 클라이언트는 안전하고 접속 지향형 세션을 통해 메시지를 주고 받는다.

- Message

세션을 통해 주고 받는 프로토콜 요소를 의미하며 메시지들은 잘 정의된 XML 문서 형태이다.

- State data

Configuration data와 다른 추가적인 정보를 의미하며 읽기 전용의 상태 정보나 수집된 통계 정보 등이 state data에 해당된다.

2.3. NETCONF 프로토콜 구조

NETCONF는 클라이언트와 서버 사이에서 RPC에 기반한 방법으로 통신한다. NETCONF 세션은 네트워크 관리자나 네트워크 설정 프로그램과 네트워크 장치 사이의 논리적인 연결을 의미한다. NETCONF는 다음과 같이 개념적인 네 개의 레이어로 나뉠 수 있다[4].

- Secure Transport layer

NETCONF는 RPC에 기반한 통신 방식 사용한다. 클라이언트가 하나 이상의 RPC 요청 메시지를 보내면 이에 대해 서버는 RPC 응답 메시지를 보낸다. NETCONF는

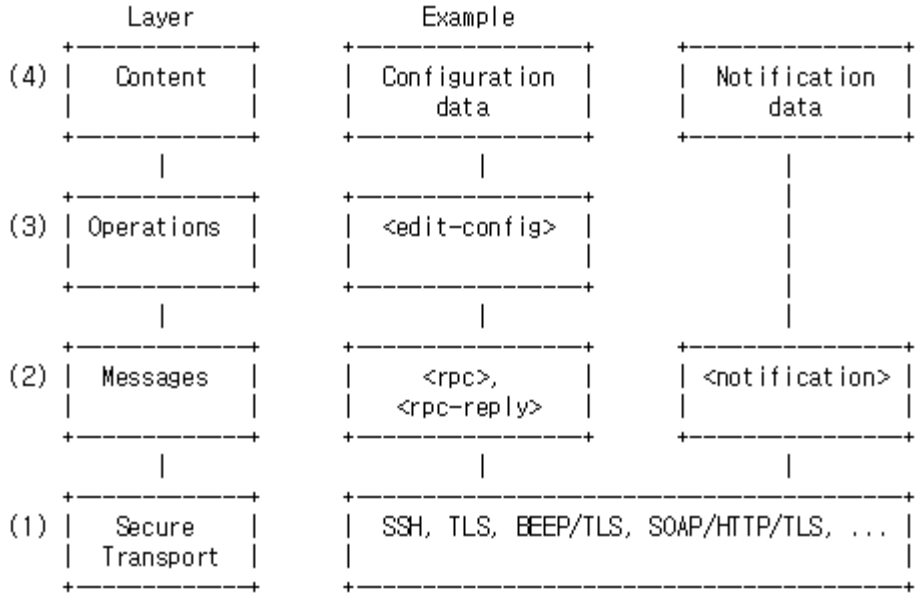

그림 1. NETCONF 프로토콜 레이어

어떠한 전송 프로토콜을 사용 할 수 있지만 NETCONF 프로토콜에 세션 타입을 명시 할 수 있는 방법을 제공해야만 한다. NETCONF는 접속 지향형 프로토콜로, 지속적인 연결을 요구한다. 이러한 연결은 순차적이고 신뢰성 있는 전송을 제공해야

한다. NETCONF 연결은 인증, 데이터 무결성, 기밀성, 응답 보호를 제공해야하며 NETCONF는 이러한 특성을 전송 프로토콜에 의존한다

- Message layer

Message layer는 전송 프로토콜과 무관한 인코딩을 위한 프레이밍 기법을 제공해야 한다. 프레이밍 기법은 RPC invocation, RPC result, Notification을 위해 사용된다. 이러한 NETCONF 메시지들은 XML 문서 규격을 준수해야한다. RPC invocation은 반드시 message-id 속성을 가져야 하며 해당 invocation에 응답하는 RPC result는 해당 속성을 가져야 한다. NETCONF 메시지는 요청이 처리 되기 전에 다른 요청이 도착 할 수 있으며 이러한 일련의 요청에 대한 응답은 반드시 요청을 받은 순서대로 해야 한다.

- Operation layer

NETCONF는 Configuration datastore에 대한 Retrieve, configure, copy, delete 명령어를 제공한다. 이러한 기본적인 기능은 NETCONF capability에 의해 확장 될 수 있다. 확장된 기능들은 세션 연결 때 서버와 클라이언트 사이의 capability exchange를 통해 알려진다.

- Content layer

Content는 NETCONF에 의해 정의되는 대상이 아니다. NETCONF가 사용하는 content는 XML을 통해 정의되며 대부분의 content는 네트워크 관리와 관련된다. Content로 사용 될 수 있는 모델로는 NETMOD working group에서 제안한 YANG data 모델이 있다.

2.4. NETCONF 명령어

- <get-config>

기능 - 특정 Configuration datastore에서 전부 또는 일부를 가져온다
변수 - source - 쿼리를 보낼 configuration datastore 이름, filter - configuration datastore에서 가져 올 부분을 명시

- <edit-config>

기능 - 전부 또는 일부 configuration을 목표 configuration datastore에 전달한다. Configuration은 파일이나 인라인으로 표현 할 수 있다. 만약 목표 configuration datastore가 없다면 새로 생성한다.
변수 - target - 목표 configuration datastore 이름, default-operation - <edit-config> 요청에 대해 기본적으로 적용될 명령어, test-option - 값을 적용하기 전 검증을 위해 수행될 옵션, error-option - <edit-config> 명령어를 수행 할 때 에러가 발생한 경우 취할 행동, config - 적용 될 configuration data

- <copy-config>

기능 - 원본 configuration datastore를 기반으로 새로운 configuration datastore를 생성하거나 기존의 configuration datastore를 대체
변수 - target - 목표 configuration datastore로 사용 될 이름, source - <copy-config> 명령어의 원본 configuration datastore 이름

- <delete-config>

기능 - Configuration datastore 삭제
변수 - target - 목표 configuration datastore 이름

- <lock>

기능 - <lock> 명령어는 목표 configuration datestore에 대한 lock을 획득 할 수 있게 해준다. Lock을 획득하게 되면 클라이언트는 다른 클라이언트에 대한 상호작용을 배제하고 configuration datastore에 대한 작업이 가능해진다. Lock은 lock이 release 되거나 세션이 종료 될 때까지 지속된다.
변수 - target - lock을 획득 할 configuration datastore 이름

- <unlock>

기능 - <unlock> 명령어는 획득한 configuration lock을 반환 할 때 사용된다.
변수 – target – lock을 반환 할 configuration datastore 이름

- <get>

…….기능 - <get> 명령어는 configuration과 device의 state information을 가져온다.
변수 – filter – system configuration과 state information에서 어떠한 부분을 가져올지 명시

- <close-session>

기능 – NETCONF 세션을 정상 종료한다

- <kill-session>

기능 – NETCONF 세션을 강제 종료한다

2.5. NETCONF 명령어 예시

그림 2는 <copy-config> 명령어를 나타내며 그림 3은 그에 대한 응답을 나타낸다. message-id 항목은 <rpc>와 그에 응답하는 <rpc-reply>가 같아야 한다. <copy-config>의 변수인 target과 source 항목이 XML 형태로 포함 되어 있다. 만약 추가적인 속성이 <rpc> 메시지에 추가된다면 이는 <rpc-reply>에 수정 되지 않고 그대로 반환 되어야 한다. 이는 어떠한 “xmlns" 속성도 포함 할 수 있다.

```
<rpc message-id="101"
        xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
      <copy-config>
        <target>
          <running/>
        </target>
        <source>
          <url>https://user:password@example.com/cfg/new.txt</url>
        </source>
      </copy-config>
</rpc>
```

그림 2. <copy-config> 명령어

```
<rpc-reply message-id="101"
    xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  <ok/>
</rpc-reply>
```

그림 3. <copy-config> 에 대한 응답

3. YANG(Yet Another Next Generation)

본 절에서는 YANG(Yet Another Next Generation)이 등장한 배경과 용어, 구조에 대해 설명한다.

3.1. YANG 개요

여러 네트워크 관리 프로토콜들은 저마다 데이터 모델링 언어를 가지고 있다. SNMP는 Structure of Management Information(SMI)라는 데이터 모델링 언어를 사용한다. YANG은 차세대 네트워크 관리 프로토콜인 NETCONF 프로토콜에서 사용되는 데이터를 모델링하기 위해 사용되는 언어이다. YANG의 기본 타입 시스템과 구조는 SMIng를 계승하였으나 SMIng과 달리 네트워크 관리 프로토콜로부터 독립되지 않고 데이터 모델이 XML로 직렬화 될 수 있어야 한다는 NETCONF 프로토콜의 개념을 채용했다[5]. YANG의 표준은 2008년 IETF의 NETMOD working group에 의해 제정되었다. 그리고 YANG 1.0 기술사양서가 2010년 10월에 RFC 6020 문서로 공개되었다.

YANG은 다음과 같은 특징을 가지고 있다[6].

- 사람이 읽고 배우기 쉬운 표현법
- 계층적인 configuration data model
- 재사용 가능한 그룹과 타입
- Configuration 검증을 위한 정규적인 constraint
- Module과 submodule을 통한 모듈성

3.2. YANG 용어[7]

- Module

Module은 하나의 데이터 모델을 정의하며 YANG에서 정의의 기본적인 단위가 된다. Module은 module-header, revision, definition statement를 가진다. Module header statement는 module를 기술하며 정보를 제공한다. Revision statement는 module의 수정 내역에 대한 정보를 제공하며 definition statement는 데이터 모델이 정의된 module의 body를 말한다.

- Submodule

Submodule은 module에 정의를 추가하는 부분 module이다. 이러한 submodule은 여러 module에 의해 참조 될 수 없다.

- Node

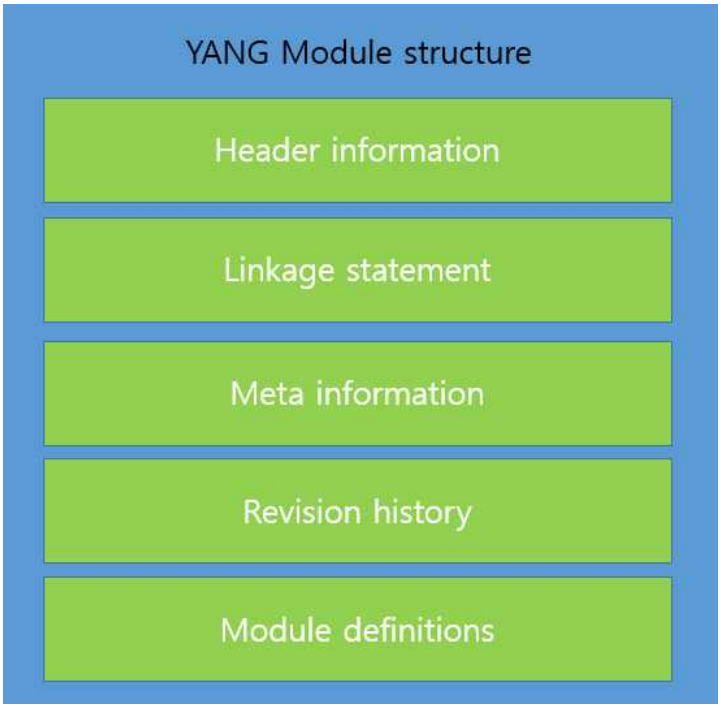

그림 4. YANG module 구조

YANG은 데이터 모델링을 위해 4가지 노드를 사용하며 leaf node는 정수나 문자열 같은 간단한 데이터를 포함한다. 이 노드는 단 한 가지의 값만 가지며 자식 노드를 가지지 않는다. Leaf-List node는 특정 데이터 타입만을 가진 leaf node들의 배열이다. Container node는 subtree에서 관련된 노드들을 묶기 위해 사용되는 노드다. List node는 여러 개의 키와 임의의 타입과 임의의 자식 노드를 가질 수 있다.

- Statement

Module의 여러 가지 속성을 기술하는 구문으로 YANG에는 module, submodule, typedef, tyhpe, container, leaf, leaf-list, list, choice 등의 statement가 있다.

- Constraint

Constraint는 계층에 존재하는 노드의 값에 대해 적용 할 수 있는 제한을 기술한다. Constraint는 configuration data, state data, notification content, RPC input parameter, RPC output parameter 등에 적용 될 수 있다.

- Built-in type

YANG은 다른 프로그래밍 언어처럼 언어 내에 정의된 built-in type을 가지고 있으며 이를 통해 추가적인 타입을 정의할 수 있다.

3.3. YANG 구조

YANG module은 다음 그림 4과 같은 구조로 되어 있다.

- Header information

Header information은 <yang-version statement>, <namespace statement>, <prefix statement> 등으로 구성 되어 있다. <yang-version statement>는 module을 정의하는 데에 사용된 YANG의 버전을 명시한다. <namespace statement>는 namespace의 URI로 되어 있으며 XML namespace를 정의한다. <prefix statement>는 module과 module의 namespace에 접근하는데 사용되는 접두사를 명시한다.

- Linkage statement

Linkage statement는 <import statement>와 <include statement>로 되어 있다. <import statement>는 다른 module이나 submodule에 정의된 grouping, typedef, extension 등의 정의를 사용 할 수 있게 해준다. <include statement>는 submodule의 부모 노드나 다른 submodule의 부모 노드가 submodule을 사용 할 수 있게 해준다.

- Meta information

Meta information은 <organization statement>, <contact statement>, <description statement>, <reference statement> 등으로 구성 되어 있다.

- Revision history

Revision history는 <revision statement>로 구성 되어 있다.

- Module definition

3.4. YANG 예시

다음 그림 5은 YANG을 사용하여 데이터를 모델링한 예제를 보여준다. Module example-sports는 "http://example.com/example-sport"로 URI 형태의 namespace를 정의하고 있다. 그리고 prefix로 sport 단어를 사용하며 ietf-yang-types라는 타입 라이브러리 모듈을 import하고 있다. 이 모듈은 sports라는 컨테이너를 가지고 있으며 이 컨테이너는 person들의 list와 player들의 list로 구성 되어 있다.

```
module example-sports {

  namespace "http://example.com/example-sports";
  prefix sports;

  import ietf-yang-types { prefix yang; }

  typedef season {
    type string;
    description
      "The name of a sports season, including the type and the year, e.g,
       'Champions League 2014/2015'.";
  }

  container sports {
    config true;

    list person {
      key name;
      leaf name { type string; }
      leaf birthday { type yang:date-and-time; mandatory true; }
    }

    list team {
      key name;
      leaf name { type string; }
      list player {
        key "name season";
        unique number;
        leaf name { type leafref { path "/sports/person/name"; }  }
        leaf season { type season; }
        leaf number { type uint16; mandatory true; }
        leaf scores { type uint16; default 0; }
      }
    }
  }
}
```

그림 5. YANG 예시

4. ONOS-LISP Management Plane

4.1. Management plane 설계

기존의 LISP 프로토콜은 장비의 설정와 관련된 부분이 별도로 존재하지 않아 LISP 지원 장비를 제어하기 위해서는 OOR이나 OpenLISP 장비에 직접 접근하여 설정을 변경해야 했다. 이러한 문제를 해결하기 위해 NETCONF/YANG을 사용하여 장비에 대한 직접적인 접근 없이 ONOS 컨트롤러 상에서 이를 제어 할 수 있는 ONOS Management plane을 디자인하였다.

그림 6는 ONOS Management plane 구조를 나타낸다. LISP 어플리케이션들은 Management plane의 Manager를 CLI나 RESTful API를 통해 접근 할 수 있다. 이를 통해 어플리케이션들은 LISP 지원 장비에 대한 명령을 내릴 수 있으며 Manager는 이를 NETCONF 명령어로 변환하여 LISP 지원 장비에 명령어를 전달한다.

Management plane은 다음과 같은 명령어를 지원한다.

- Connect to OORdevice through NETCONF

형태) lisp-connet {username} {password} {address} {port}
예시) lisp-connet foo bar 192.168.0.1 830

- Get map configured map resolvers of a device

형태) lisp-get-map-resolver {deviceId}
예시) lisp-get-map-resolver netconf:192.168.10.1:830

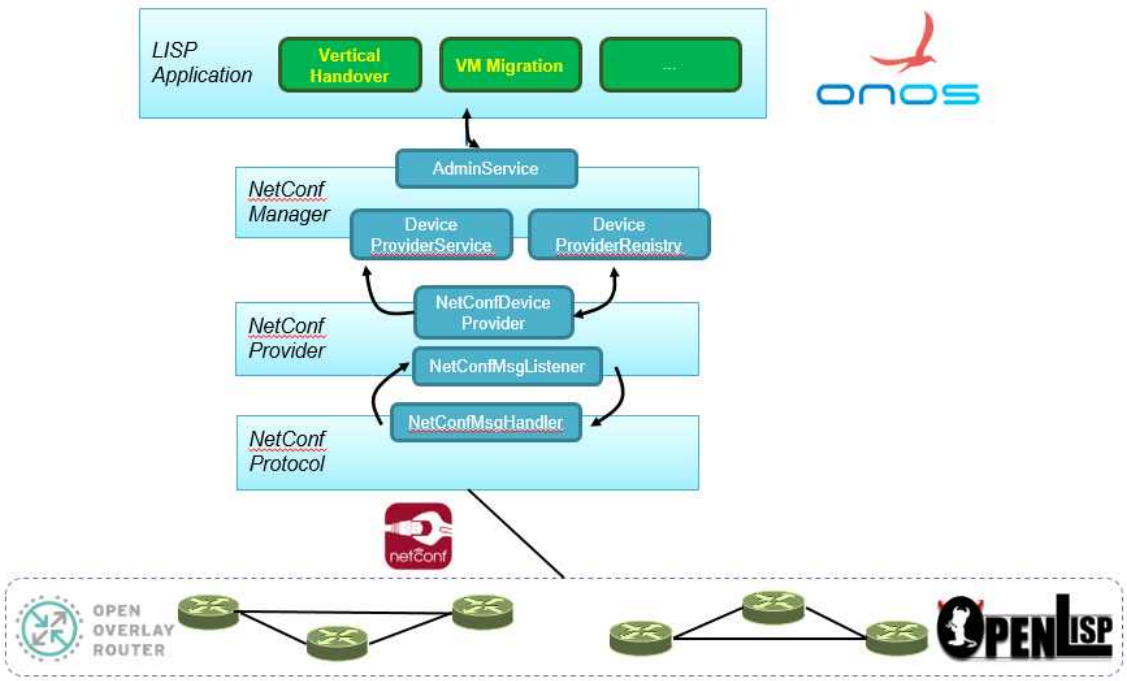

그림 6. ONOS Management plane 구조

- Add map resolver to the device

형태) lisp-add-map-resolver {deviceId} {resolver-address}
예시) lisp-add-map-resolver netconf:192.168.10.1:830 10.10.10.10

- Remove map resolver of the device

형태) lisp-remove-map-resolver {deviceId} {resolver-address}
예시) lisp-remove-map-resolver netconf:192.168.10.1:830 10.10.10.10

- Add local eid DB

형태) lisp-add-local-eid {deviceId} {EID} {netmask} {RLOC} {Priority} {Weight}
예시) lisp-add-local-eid netconf:192.168.10.1:830 10.10.10.10 32 192.168.10.1 1 1

- Remove local eid DB

형태) lisp-remove-local-eid {deviceId} {EID} {netmask} {RLOC} {Priority} {Weight}
예시) lisp-remove-local-eid netconf:192.168.10.1:830 10.10.10.10 32 192.168.10.1 1 1

- Get configured local DB of a device

형태) lisp-get-local-db {deviceId}
예시) lisp-get-local-db netconf:192.168.10.1:830

4.2. Management plane 구현

LISP Management plane은 다음 그림 7과 같이 구현 되어 있다. LispConfigService. 파일은 Management plane의 인터페이스를 정의하는 파일이다. cli와 rest 폴더는 Management plane의 CLI와 RESTful API를 구현하기 위한 코드를 담고 있으며 이들은 LispconfigService의 함수를 호출하는 형태로 구현 되어 있다.

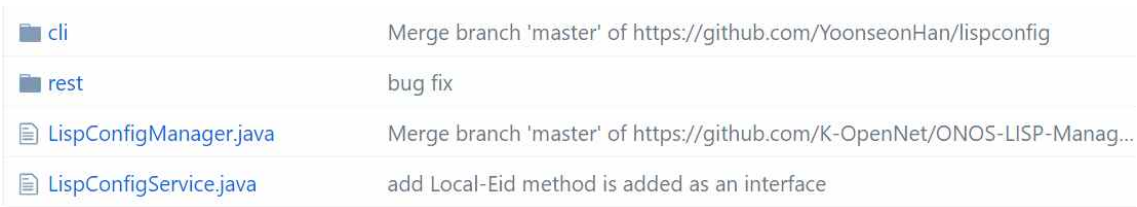

그림 7. Management plane 구조

- LispConfigManager

LispConfigManager 파일은 LispConfigService 인터페이스를 구현하는 파일로 LISP Management plane의 핵심적인 기능을 구현하는 파일이다. LispConfigManage 파일은 다음과 같은 함수를 구현한다.

- getConfig

deviceId 변수를 받아 ONOS NETCONF 컨트롤러로부터 현재 설정 정보를 받아온다.

- addItrMapResolver

전역 변수로 deviceId와 해당 deviceId가 가지고 있는 Mapresolver들의 리스트에 대한 매핑을 가지고 있다. addItrMapResolver는 input으로 들어온 deviceId를 key로 Mapresolver 리스트를 매핑으로부터 찾는다. 만약 매핑이 존재하지 않는다면 새로 매핑을 생성하고 input으로 들어온 Mapresolver의 주소를 추가한다. 그리고 updateItrMapResolver 함수를 호출한다.

- removeItrMapResolver

input으로 들어온 deviceId로 Mapresolver 리스트를 찾아 삭제한다. 그리고 updateItrMapResolver 함수를 호출한다.

- addEtrEidDataBase

전역 변수로 deviceId와 해당 deviceId가 가지고 있는 LispMapRecord들의 리스트에 대한 매핑을 가지고 있다.

- removeEtrEidDataBase

input으로 들어온 deviceId로 LispMapRecord 리스트를 찾아 삭제한다. 그리고 updateEtrEidDatabase 함수를 호출한다.

- connectDevice

input으로 들어온 address와 port를 사용하여 registerDevice 함수를 호출하고 configureDevice 함수를 호출하여 두 함수가 제대로 수행되었으면 true를 반환한다.

- updateItrMapResolver

deviceId가 가지고 있는 Mapresolver 리스트를 string으로 변환하고 포맷에 맞춘 뒤 copyConfig 함수로 해당 device에 설정을 반영한다.

- updateEtrEidDatabase

deviceId가 가지고 있는 LispMapRecord 리스트를 serializeMapRecord 함수를 통해 string으로 변환하고 포맷에 맞춘 뒤 copyConfig 함수로 해당 device에 설정을 반영한다.

- serializeMapRecord

LispMapRecord를 정해진 포맷에 따라 string으로 변환한다.

5. 결론

기존의 네트워크를 관리하는 방식이 점차 규모도 방대해지고 고도화 되는 네트워크를 관리하는 데에 한계점을 보임에 따라 이를 해결하기 위해 차세대 네트워크 관리 프로토콜인 NETCONF와 이에 사용되는 데이터 모델링 언어인 YANG이 확산되고 있다.

본 기술문서는 NETCONF와 YANG에 대해 기술하며 이를 통해 LISP 지원 장비를 ONOS 컨트롤러에서 관리 할 수 있도록 NETCONF/YANG을 사용하는 ONOS Management Plane에 대한 설계 및 구현을 서술하였다. 현재 개발된 ONOS LISP Management Plane은 추가적인 명령어 지원과 더 많은 LISP 지원 장비에 대한 지원, 그리고 궁극적으로는 ONOS에 구현된 LISP Subsystem과 통합하여 하나의 subsystem으로 구현하고자 한다. 이를 위해 ONOS LISP Subsystem과 통합하는 과정을 진행하고 있다.

References

[1] NETCONF, https://en.wikipedia.org/wiki/NETCONF

[2] 이재미, “NETCONF-YANG의 이유있는 탄생!”, http://www.ciscokrblog.com/694

[3] J. Schoenwaelder., "Overview of the 2002 IAB Network Management Workshop," IETF RFC 3535, May, 2003

[4] R. Enns *et al.*, "Network Configuration Protocol (NETCONF)," IETF RFC 6241, June, 2011

[5] YANG, https://en.wikipedia.org/wiki/YANG

[6] “What is YANG?", http://www.tail-f.com/education/what-is-yang/

[7] M. Bjorklund., "YANG - A Data Modeling Language for the Network Configuration Protocol (NETCONF)," IETF RFC 6020, Oct, 2010

K-ONE 기술 문서

작성기관: K-ONE Consortium
작성년월: 2017/04