

Data Modeling Method of NETCONF Protocol's Content Layer Applying VTD-XML

Lee Yang Min[†] · Jae Kee Lee^{**}

ABSTRACT

It is appropriate to use the NETCONF to monitor and manage today's complex networks that are composed of variety links and heterogeneous equipment. Since the first standard of the NETCONF are launched, there have been several revisions, but disadvantages of each layer capabilities is still present and the most typical disadvantage is XML document processing efficiency of the Content layer. In this paper, we perform data modeling by constructing a sub-tree based on the dependencies between Content layer data, and suggest method of extract efficiently data from XML by applying the extended VTD-XML technique for the XPath query. We performs experiment to compare NETCONF in proposed method to NETCONF in previous studies and NETCONF standard. we validate superiority of improved NETCONF in the paper. As experimental results, we verify that improved NETCONF is better than the other two NETCONF each 4% and 10% in terms of query processing rate, and faster than each 3.9 seconds and 10.4 seconds in terms of query processing speed.

Keywords : NETCONF, Network Management, Subtree, Filtering, VTD-XML, XPath

VTD-XML을 적용한 NETCONF 프로토콜 Content 계층의 데이터 모델링 기법

이 양 민[†] · 이 재 기^{**}

요 약

이질적인 링크와 다양한 장비로 구성된 오늘날의 복잡한 네트워크를 모니터링하고 관리하기 위해서 NETCONF를 사용하는 것이 적합하다. NETCONF의 첫 표준이 배포된 이래로 여러 번의 개정이 있었으나 계층별로 기능의 단점은 여전히 존재하고 있는데, 가장 대표적인 것이 Content 계층에서의 XML 문서 처리 효율성이다. 본 논문에서는 Content 계층에 데이터 간 종속성에 기반한 서브트리를 구성하여 데이터 모델링을 수행하고, XPath 질의에 대해 확장된 VTD-XML 기술을 적용하여 XML로부터 데이터를 효율적으로 추출하는 방법을 제안한다. 제안 기법이 적용된 NETCONF와 선행 연구의 NETCONF 및 표준 NETCONF를 비교하는 실험을 수행하고 논문에서 개선한 NETCONF의 우수성을 검증한다. 실험 결과로 본 논문에서 개선한 NETCONF가 다른 두 가지 NETCONF에 비해 질의 처리율에서는 각각 4%와 10% 우수하며, 질의 처리 속도에 있어서는 각각 3.9초와 10.4초 빠름을 확인하였다.

키워드 : NETCONF, 네트워크 관리, 서브트리, 필터링, VTD-XML, XPath

1. 서 론

이질적인 링크와 다양한 장비로 구성된 오늘날의 네트워크를 모니터링하고 관리하기 위한 업무에는 XML 문서 기

반 장비 관리 모델을 사용하는 NETCONF를 사용하는 것이 적합하다. NETCONF는 SNMP와 같이 기존의 관리 프로토콜의 기능적인 한계점을 극복할 수 있고 다양한 장비에 대해 표준적인 방법으로 중앙 집중적 관리가 가능하다[1, 2]. 2006년도에 IETF에 의해 NETCONF의 첫 번째 표준이 배포된 이래로 기능의 꾸준한 업데이트와 다양한 개정이 있었고, 많은 연구자들에 의해 NETCONF가 가지는 여러 가지 문제점도 해결되었다. 그러나 계층별로 단점이 남아있는데, 가장 대표적인 것이 Content 계층에서 XML 문서 처리의 효율성에 대한 것이다. NETCONF의 최상위 계층인 Content

* 이 논문은 2014년도 정부(교육과학기술부)의 재원으로 한국연구재단의 기초연구사업 지원을 받아 수행된 것임.

† 정 회 원 : 동아대학교 컴퓨터공학과 정보기술연구소 조교수

** 정 회 원 : 동아대학교 컴퓨터공학과 교수

Manuscript Received: August 21, 2015

First Revision: September 25, 2015

Accepted: October 2, 2015

* Corresponding Author : Jae Kee Lee(jklee@dau.ac.kr)

계층에서 데이터 모델링은 현재 YANG을 통해 XML 문서 형태로 표현하는 것이 사실상의 표준이다. 그러나 만약 XML 문서의 크기가 숫자가 많으면 XML 문서에서 필요한 정보를 추출하거나 저장하는 작업의 처리 속도는 상대적으로 느리다. 따라서 XML 문서의 교환을 기본으로 하는 NETCONF에서 XML 데이터의 추출 및 처리 속도와 데이터의 크기는 NETCONF를 이용하는 네트워크 관리 시스템의 성능에 영향을 줄 수 있다.

본 논문에서는 Content 계층에 데이터 간 종속성을 중심으로 서브트리를 구성하여 데이터를 모델링하는 기법을 적용하고[3], XPath 질의에 대해 확장된 VTD-XML(Virtual Token Descriptor for XML)[4, 5]라 불리는 파싱 기술을 적용하여 XML로부터 데이터를 효율적으로 추출하고 모델링하는 방법을 제안한다. 두 가지 기법을 적용하여 Content 계층의 처리 효율을 증가시키고, NETCONF를 이용하는 네트워크 관리 시스템의 성능도 증가될 수 있도록 한다. 제안 기법이 적용된 NETCONF와 데이터 종속성 기반의 모델링을 사용하는 선행 연구의 NETCONF, 그리고 표준 NETCONF를 함께 비교하는 실험을 하였다. 그리고 실험 결과를 분석하여 이 논문에서 제안한 기법의 타당성을 검증할 것이다.

논문의 구성은 2절에서 관련 연구에 대해 간략하게 언급하고, 3절에서 NETCONF의 Content 계층에 적용한 기법과 제안 시스템의 구성을 상세하게 설명하며, 4절에서는 표준 NETCONF와 데이터 종속성 기반의 모델링을 적용한 NETCONF[1]의 성능과 본 논문에서 제안한 기법을 적용한 개선된 NETCONF의 비교 실험과 분석 결과를 기술한다. 마지막 5절에서는 최종 결과와 향후 과제를 포함한 결론을 제시함으로써 논문을 맺도록 한다.

2. 관련 연구

2.1 NETCONF와 관련된 다양한 관련 연구

NETCONF와 관련된 연구는 매우 많다. 연구의 종류를 분류하면 크게 두 가지인데, NETCONF 전체를 고려하는 관리 시스템 개발 분야와 프로토콜의 각 계층을 개선하는 연구 분야로 구분할 수 있다.

첫째, NETCONF 관리 시스템 연구 분야에서 가장 대표적인 것은, 네트워크 관리 업무를 위해 Yencap라고 불리는 NETCONF 관리 시스템(Suite)을 들 수 있다[6]. 또한 NETCONF와 호환 가능한 IP 공유 장비를 위해 XCMS(XML-based Configuration Management System)라는 시스템도 개발되었다[7]. NETCONF에 기반을 둔 네트워크 관리 구조를 개발한 연구도 있다. 이 연구의 경우 NETCONF 기반의 네트워크 관리 시스템 구조와 그것의 동작 요소를 기술하고 있다[8].

둘째, NETCONF의 각 계층과 관련된 연구이다. Transport 계층과 관련된 연구는 요즘 연구 분야에서는 이루어지지 않고 있다. 이 계층에서 여러 전송 프로토콜이 경쟁을 하였지

만 현재는 안정적인 장비의 설정 변경과 보안을 이유로 SSH를 기반으로 하는 것이 일반화되었다. RPC 계층에서는 주로 RPC 계층에서 처리 속도를 증가시키는 데 초점을 맞추고 있다[9, 10]. 처리 속도를 증가시키는 다른 방법으로는 NETCONF의 장비 관리를 위한 오버헤드를 감소시키는 방법이 있다[10, 11]. Operation 계층의 개선에서는 <lock> 연산을 수정하거나 장비 설정과 관련된 메모리를 부분적으로 잠글 수 있게 개선하는 연구들이 존재한다[1, 12-15]. Content 계층과 관련된 연구에서 가장 큰 문제는 데이터 모델링 표준이 없다는 것이다. 이런 문제점을 지적하는 연구[9]가 있지만 문제점을 해결할 수 있는 방법을 제시하지는 못하고 있다. 강제된 표준이 없다고 하더라도 XML을 이용하여 데이터 모델링을 하는 것이 사실상의 표준으로 받아들여지고 있다. 다만 XML을 이용하여 데이터를 모델링할 때 생성되는 구조가 XML 문서이기는 해도 어떤 구조(예. 트리 구조)로 생성할 것인가와 이를 어떤 형태로 처리할 것인지에 대해서는 여전히 다양한 연구들이 진행되고 있다[16, 17].

2.2 XML 처리 기술

XML이 인터넷에서 정보를 표현하고 전달하는 용도로 사용되기 시작한 것은 꽤 오래되었고 사용되는 영역 또한 크게 확장되었다. 이와 같은 영역 중에 NETCONF에서 장비 설정을 구성하여 저장하는 목적이나 설정 변경 명령을 전송하는 것에 목적을 둔 네트워크 관리 영역이 포함된다[4, 5, 7].

NETCONF에서 XPath 기능은 <filter> 원소 내에서 XPath 표현식의 사용을 NETCONF 피어(Peer)가 지원할 수 있다는 것이다. XPath는 XML 문서 내에서 정보를 찾기 위한 언어로 XML 문서 내의 원소와 속성 사이를 탐색하기 위해 사용된다. XPath는 XML 문서 내에서 노드 또는 노드 집합을 선택하기 위한 경로 표현식을 이용한다. 이런 경로 표현식은 전통적인 컴퓨터 파일 시스템에서 작업할 때 흔히 보는 표현식과 매우 유사하다. 경로 표현식은 특정 노드 또는 특정값을 포함한 노드를 발견하기 위해 사용될 수 있는 Predicate(술어부)를 포함할 수 있다. XPath는 100개의 내장 함수를 가지고 있다.

NETCONF에서 XML 서브트리 필터링은 <get> 또는 <get-config> 연산에 대해 서버의 데이터 모델로부터 특정 XML 서브트리를 에이전트가 선택하도록 해주는 기법이다. 에이전트에서 서브트리 필터는 하나도 없거나 또는 그 이상의 원소와 원소들의 속성으로 구성된다. XPath와는 달리 서브트리 필터링을 사용하는 에이전트는 프로세싱 동안에 어떤 특정 의미를 가지는 데이터-모델의 활용이 필요하지 않다. 서버 내에서 지원되는 XML 파싱(parsing) 데이터를 이용하여 획득되는 XML 정보에 따라 데이터 모델이 생성된다[3].

XML을 처리하는 방법과 관련된 연구에서 사용되는 서브트리와 XPath 기법은 각각의 장점과 단점이 존재하는데, XPath는 표현 방법의 유연성과 자료의 추출 방법이 직관적이지만 처리 속도가 오래 걸린다. 서브트리 방법은 처리 속도 면에서는 XPath보다 탁월하지만 사용자의 관점에서 보

면 이해하기가 어렵고 Operation 계층과의 연동을 생각하면 적용하기에 힘든 것이 사실이다[4, 5].

3. 제안 기법과 시스템 구성

이 논문에서 제안하려는 방법은 크게 두 가지로 요약할 수 있다. 한 가지는 Content 계층에서 장비 설정 데이터를 XML로 생성할 때 서브트리를 기반으로 하는 저장 방법을 사용한다. 그리고 장비의 설정 데이터 종속성을 XPath에 의한 상대적 경로 간의 관계로 만들어진 테이블로 표현하여 모델링한다. 다른 한 가지는 오픈 소스로 공개되어있는 VTD-XML의 확장판을 사용하여 앞의 기법을 고속으로 처리함과 동시에 XML 문서로부터 적합한 데이터를 고속으로 추출하는 것이다.

3.1 제안 시스템 개요

논문에서 적용하려는 VTD-XML과 XPath를 이용한 데이터 추출 및 Content 계층의 데이터 모델링은 전부 NETCONF 에이전트 측에 포함된다. 매니저로부터 전달되는 연산을 포함한 XML 문서는 XML 후보 문서를 저장하는 큐에 들어와서 후보 문서를 체크하는 모듈로 전달된다. 선행 연구[1]에서 제안한 규칙을 통해 우선순위를 확인하고 긴급 XML 문서 또는 일반 XML 문서로 분류된다. 어떤 종류의 XML 문서에 상관없이 VTD-XML을 통해 바이너리 배열로 표현되고 XML의 문서의 모든 원소는 그 위치와 내용이 저장된다. 그리고 배열의 내용을 가지고 XPath 질의를 통해 필요한 데이터를 추출한다.

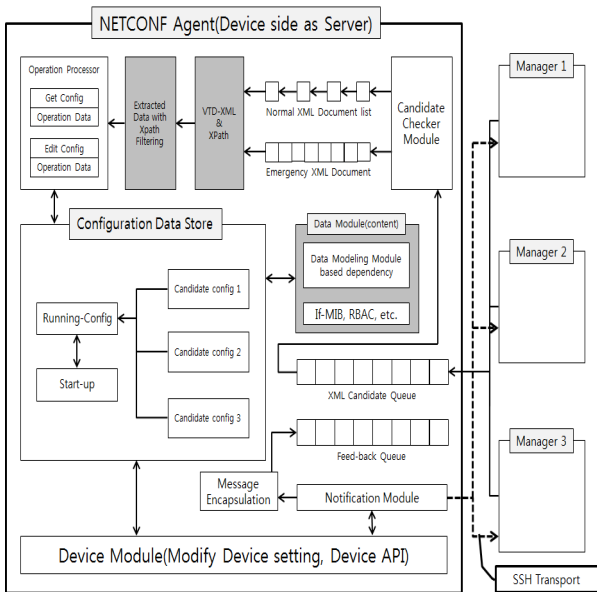


Fig. 1. Structure Drawing of Proposed System

Operation 계층의 연산이 수행된 후 장비 설정 데이터의 저장소와 데이터 모듈(Data Module)은 주기적으로 통신하면

서 장비 설정 데이터의 종속성에 기반한 설정 데이터를 모델링하고 저장한다. 데이터를 모델링할 때에도 VTD-XML에 의해 생성된 배열 내의 XML 원소 위치의 주소를 기반으로 한다. 특히 XPath로 표현 가능한 상대 경로를 활용하여 장비 데이터 간의 종속성을 표현하고 종속성을 가진 데이터들을 하나의 서브트리로 구성하는 테이블을 생성하는데 이것은 3.3절에 자세히 기술하였다. 이와 같은 기법들을 적용한 전체 시스템 구조도는 Fig. 1에 나타나있다.

3.2 VTD-XML와 XPath를 적용한 데이터 추출

본 논문에서는 XML 문서 형태에서 적합한 데이터를 추출하기 위해 확장된 VTD-XML을 적용한다. VTD-XML 모델에 기반한 XML 파싱은 “가상 토큰 서술자”라 불리는 “비-추출” XML 기술에 초점을 맞추고 있다. “비-추출”은 본래 XML 문서는 바이너리 형태로 메모리 내로 읽어지고 모든 원소의 위치는 바이트 배열로 분석되어서 그 정보가 기록된다는 것을 뜻한다(Fig. 2).

순회 연산은 바이트 배열로 분석된 정보 기록을 이용하여 이루어진다. VTD-XML은 XML 문서를 파싱하고 각 이벤트에 대해서 64bit 바이너리 포맷의 VTD 레코드(토큰)를 생성한다. VTD-XML은 DOM과 비교하면 더 높은 성능을 내면서 더 적은 자원을 필요로 한다. 따라서 VTD-XML을 사용하면 XML 문서의 빠른 분석과 처리가 가능하다. 여기에 XPath를 필요한 데이터를 추출하는 목적으로만 사용하여 XPath의 표현 유연성의 장점을 취하고 XPath가 일반적인 서브트리 방식보다 느린 단점을 보완할 수 있다[4].

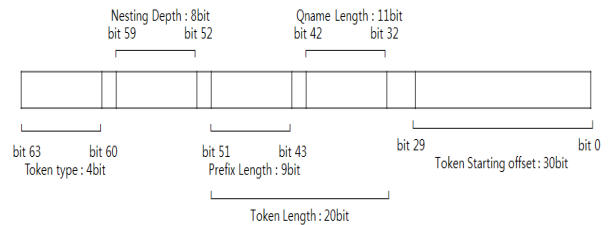


Fig. 2. VTD Record's Byte Array

XML 파서는 256GB 이상의 문서를 처리할 수 있도록 XPath 질의를 사용하며 이런 능력은 다수의 네트워크 관리자가 매우 많은 수의 장비 설정 명령을 빈번하게 만들어낼 때도 유용하다. XPath의 데이터 추출 과정은 Fig. 3과 같다.

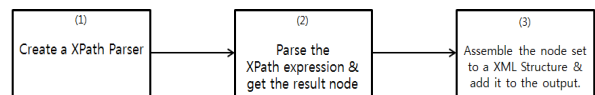


Fig. 3. Process of XPath Filtering

3.1절에서 서브트리 기반으로 하여 XML 문서를 저장하고 장비 설정 데이터의 종속성을 이용하여 모델링하였기 때문에 XPath를 이용하여 관련 있는 데이터 아이템을 묶을 수 있고 낭비적인 객체의 생성을 회피할 수 있다. XPath 표

현식을 기반으로 작성된 코드는 이해하기도 쉽고 디버거도 단순하며 쉽게 이해될 수 있고 쓰는 것과 디버그가 단순하다. 결과적으로 VTD-XML은 트리 구조를 기반으로 하면서 최종적으로는 서브트리 루트의 포인터로 구성되는 데이터 구조를 고속으로 생성하는 데 적합하다. VTD-XML은 세 개의 컴포넌트로 구성되고 최상위 레벨에 나타나는 VTD-XML을 위한 자바 API가 있다. 세 개의 컴포넌트는 VTD generator, VTD navigator, Autopilot이다.

3.3 장비 설정 정보의 종속성을 이용한 Content 계층의 데이터 모델링 기법

1) 서브트리 기반의 XPath 데이터 저장 방법[3]

장비의 설정 정보는 최종적으로는 메모리에 저장되지만 실제 CPU 점유나 시간 소모를 하는 것은 설정 데이터인 XML 문서를 파싱하고 필요한 데이터를 추출하거나 저장하는 작업이다. Fig. 4는 NETCONF로 <get-config> 연산을 수행한 후에 응답을 받는 예시를 나타낸 것이다. 구조적으로 보면 <interface> 원소 내에 이름, IP 주소, MAC 주소 등이 트리 형태로 포함되어있는 것을 알 수 있다. 또한 실제 속성값은 다르지만 <interface> 원소가 두 번 반복되고 있는 것도 확인이 가능하다. 이 연구에서는 서브트리 기반 XPath 처리를 통한 데이터 저장법을 이용해서 네트워크 장비 설정 데이터 종속성을 표현하고 모델링하는 방법을 제안한다. 모델링은 우선 Fig. 4를 Fig 5와 같은 형태로 만드는 것에서 시작한다.

```
<?xml version="1.0" encoding="UTF-8"?>
<rpc-reply message-id="101" xmlns="urn:ietf:params:netconf:base:1.0">
  <data>
    <interface xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces1">
      <name> eth1 </name>
      <ipv4-address> 192.168.5.10 </ipv4-address>
      <macaddr>aa:bb:cc:dd:ee:ff</macaddr>
      <subnet> <name> eth.1.1 </name> </subnet>
    </interface>
    <interface xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces2">
      <name> eth2 </name>
      <ipv4-address> 192.168.5.20 </ipv4-address>
      <macaddr>kk:ll:mm:nn:oo:pp</macaddr>
      <subnet> <name> eth.2.1 </name> </subnet>
    </interface>
  </data>
</rpc-reply>]]]]>
```

Fig. 4. XML Document of Reply

현재 XML 데이터 저장의 최신 기법은 XML 노드 레이블링(labeling)이다. XML 원소에 레이블링을 한 예시는 Fig. 6에 있는 (2:25, 2)와 같은 표기이다. 이것은 Fig. 5에서 첫 번째 interface를 나타내는데, 콜론으로 구분되는 앞의 두 숫자는 (시작:끝)을 알리는 범위이고 뒤의 숫자는 트리 내에서 그 노드의 깊이를 뜻한다.

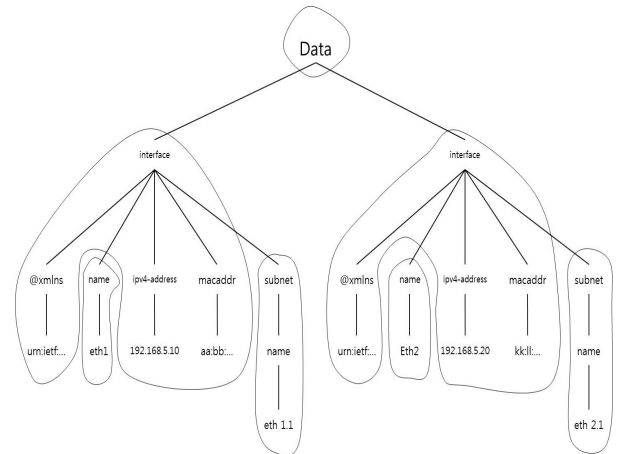


Fig. 5. Tree of XML Document and Its Subtree (Mark with Free-Line)

레이블은 XPath 처리를 진행할 때 구조적 조인을 가능하게 한다. 즉, n_1, n_2 라는 노드가 트리 내에 있다면 그 노드들의 구조적 관계인 조상-후손 또는 부모-자식 관계는 레이블로 결정된다. 또한 n_1 의 범위가 n_2 의 범위를 포함한다면 n_1 은 n_2 의 조상이 된다. 조상-후손 쌍인 관계에서 레벨이 1 차이가 있으면 노드 하나는 다른 노드의 부모 노드이다. 이와 같은 레이블을 통해 Fig. 5에 있는 것과 같은 데이터를 저장하게 되는데 저장된 아이템은 각각의 노드들로 구성된다. 노드는 원소의 태그 이름 또는 속성 이름(접두어 @)이다. XML 트리의 서브트리는 XML 저장소의 한 저장 단위로 간주할 수 있다. 예를 들면, Fig. 5의 XML 트리에서 자유선(free-line)으로 표기된 부분은 XML 트리가 서브트리로 분할될 수 있음을 보여준다. 각 서브트리는 하나의 저장 단위로 저장 가능하며, 서브트리-기반 XML 저장에서는 전체 XML 트리가 분할 없이 전체로 저장될 수 있고, 전통적인 노드-기반 저장에서는 노드별로 완전히 분리되어 저장된다[3].

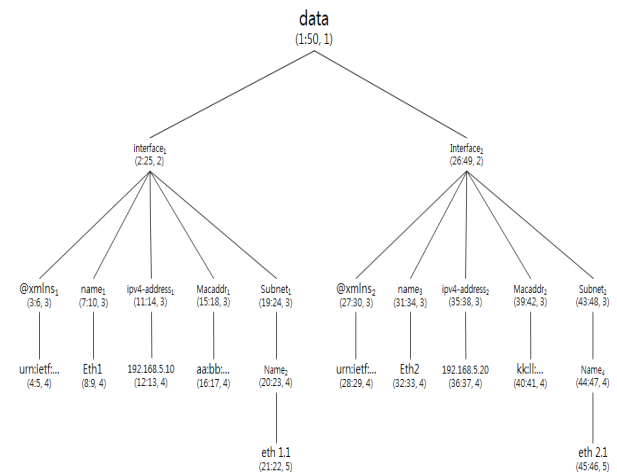


Fig. 6. Tree of XML Document with Labeling

XML 데이터를 서브트리에 저장하고 XPath 질의를 서브트리에서 추출을 위한 파싱 없이 처리하기 위해 XML path schema와 XML fragmentation schema 스키마가 필요하다. path schema는 Fig. 7에 나타난 것처럼 트리로 표현될 수 있는데 Fig. 5의 XML 트리로부터 만들어진 것이다. 각 노드 N 의 옆에 접두어 #을 붙여 표기한 숫자는 노드 N 의 path ID로, 루트로부터 N 까지의 path type으로서 할당된다. 예를 들면, path type인 /data/interface/ipv4-address는 Path ID가 #5로 할당된다. XPFS 트리에는 두 가지 종류의 노드가 있는데 하나는 단일 원으로 표현된 것이고 다른 하나는 이중 원이다. 이중 원은 저장될 서브트리의 루트를 나타내고 단일 원은 서브트리 t 에 소속된 자손 노드를 나타낸다. XPFS에서의 서브트리 정보와 Fig. 6의 레이블링 정보를 포함하여 저장되는 방식을 나타내면 Table 1과 같이 표현된다.

<Table 1>에는 7개의 서브트리가 저장되어있다(no 열은 단순한 참조 번호이므로 저장 내용과는 무관). 서브트리 (1), (2), (5)는 내부에 다른 서브트리가 있다는 의미의 기호 "\$"를 포함한다. 본 논문에서 참조한 [3]은 XML 문서가 특정 DTD를 따른다는 것을 가정하고 있는데, XML 트리를 서브트리로 분할하는 데 있어서 기본 XML 분할 규칙이 '+, *, ?'로 표기되는 것은 다른 서브트리의 루트가 될 수 있는 반복 또는 선택 원소이다.

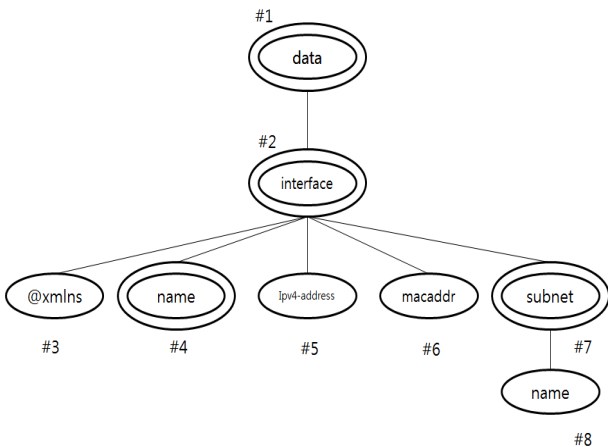


Fig. 7. XPFS(XML Path Fragmentation Schema) Tree for 'Fig. 5'

Fig. 6의 interface1과 interface2는 반복 원소인데 두 원소는 같은 부모인 data 노드를 가지며, 따라서 path type은 /data/interface로 동일하다. 기본 XML 분할 규칙하에서 두 원소는 서브트리의 루트이다. 선택 원소에 대해서는, path type이 /data/interface/name에서 태그 이름이 'name'인 XML 노드가 선택적이라면 그들의 각 인스턴스 $name_1$ 과 $name_3$ 은 서브트리의 루트이다. 이런 방법을 통해 Table 1과 같이 저장하는 방법의 이점은 데이터 저장 공간을 줄임과 동시에 파싱 없는 XPath 질의를 부분적으로 처리할 수 있다는 점인데 이에 관한 내용은 [3]에 보다 상세히 나타나 있다.

본 논문에서는 이런 XML의 서브트리 기반 저장법을 활용하여 장비의 설정 정보를 처음 만들고 저장할 때 데이터 사이의 종속성과 서브트리 기반 저장방식을 접목하도록 하였다.

Table 1. Storage Base on Subtree

no	name	path ID	begin	end	value
(1)	data	#1	1	50	<data>\$\$</data>
(2)	interface	#2	2	25	<interface xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces1"> <ipv4-address>192.158.5.10</ipv4-address> <macaddr>aa:bb:cc:dd:ee:ff</macaddr>\$\$</interface>
(3)	name	#4	7	10	<name>eth1</name>
(4)	subnet	#7	19	24	<subnet><name>eth1.1</name></subnet>
(5)	interface	#2	26	49	<interface xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces2"> <ipv4-address>192.158.5.20</ipv4-address> <macaddr>kk:ll:mm:nn:oo:pp</macaddr>\$\$</interface>
(6)	name	#4	31	34	<name>eth2</name>
(7)	subnet	#7	43	48	<subnet><name>eth2.1</name></subnet>

2) XPath 처리를 통한 데이터 저장과 장비 설정 데이터의 종속성을 통합한 데이터 모델링

앞에서 기술한 XML 데이터의 저장법을 데이터 종속성을 활용한 기법과 접목하면 초기에 네트워크 장비의 설정 정보의 데이터를 모델링할 때 관련 있는 데이터들을 하나의 블록으로 다룰 수 있다. 예를 들어 NETCONF에서 인터페이스와 그 인터페이스상의 서브 넷(Subnet)에 관련된 작업을 수행하려고 할 때, XML 문서로 표현된 특정 명령어에서 XPath의 질의는 XPath 표현식으로 나타내면 다음과 같은 형태가 될 수 있다.

```

(1) /data/interface[@xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces1"]//name
(2) /data/interface[@xmlns="urn:ietf:params:xml:ns:yang:ietf-interfaces2"]//name
    
```

이와 같은 표현식에 대해 특정 인터페이스의 이름, 그리고 서브 넷의 인터페이스 이름을 변경하거나 검색해서 반환한다고 가정하자. 본 논문에서 제시하는 XML 저장 구조를 사용하여 데이터를 모델링한다고 가정하면 이런 질의에 대해서 Fig. 8과 같은 형태의 논리적인 데이터 구조가 생성된다.

(1)의 질의에 대해서는 Fig. 8의 ①과 같은 사각형 내의 서브트리를 하나의 데이터 블록으로 하여 서브트리의 루트를 기준으로 접근할 수 있다. 마찬가지로 (2)의 질의에 대해서 서브트리 구조는 동일하지만 저장 범위가 다른 ②사각형 내의 서브트리를 하나의 데이터 블록으로 다룰 수가 있다. 두 가지 질의에 대해서 인터페이스의 이름과 서브 넷을 사용하는 인터페이스의 이름을 검색할 수 있다. 경우에 따라서 장비의 인터페이스의 모든 이름을 반환하라는 질의에 대

해서는 ①의 점선 사각형을 사용하여 데이터를 추출할 수 있다.

이와 같은 데이터 저장구조는 장비의 데이터 종속성에 의존하여 데이터 블록을 만들고 여러 명의 관리자가 동시에 하나의 장비에 접근할 때에도 각각의 사각형 하나를 개별적으로 잠그면(lock 연산) 부분 잠금 연산을 수행할 수 있다. 그리고 Fig. 8에 구분한 것과 같이 데이터 구조의 논리적 부분과 물리적 부분이 완전히 분리되어있고 네트워크 관리자의 관점에서 볼 때 물리적 부분은 추상화가 되어있으므로 XML 문서의 서브트리 저장구조만을 보고 장비 설정 구성이 가능하다.

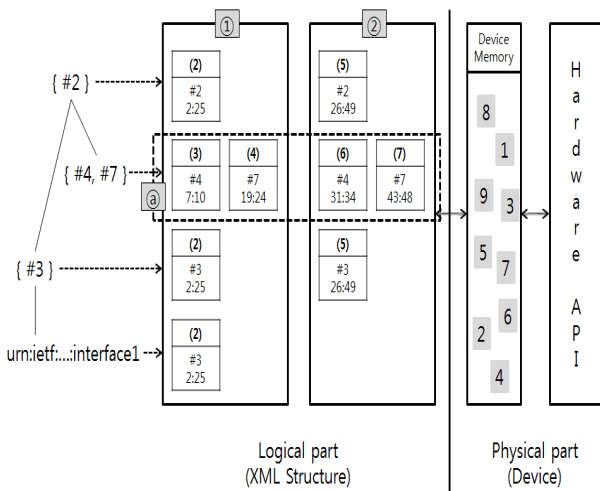


Fig. 8. Stored Data Structure for NETCONF Query

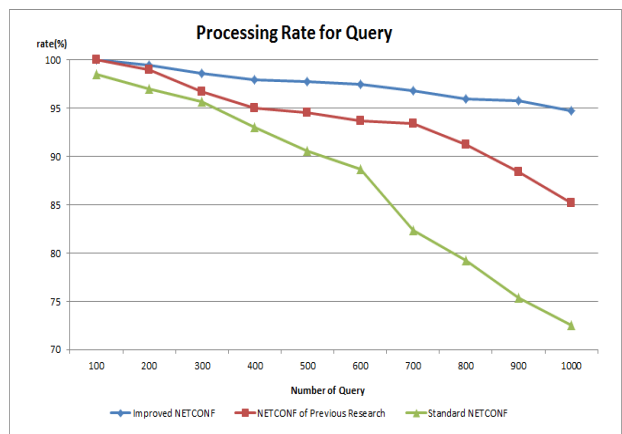
4. 실험 환경 및 실험 결과

4.1 실험 환경

본 논문에서 적용한 데이터 모델링 기법을 포함한 NETCONF와 1절 및 2.1절에서 언급한 데이터 종속성을 고려한 데이터 모델링 기법을 적용한 선행 연구[1]의 NETCONF, 그리고 표준 NETCONF 간의 비교를 수행하였다. 적용한 데이터 모델링 기법을 위해서 VTD-XML은 오픈 소스를 활용하여 64bit JVM에서 기동하였고, 바이너리 배열로부터 장비 설정 데이터 종속성에 의한 모델링을 위해서 C언어를 사용하여 선행 연구에서 개발하였던 NETCONF GUI 프로그램의 에이전트 부분에 탑재하였다. 가상의 스위치를 활용하였기에 실제 장비에서의 테스트는 본 논문에서 수행하지 않았다. 실험 환경은 가상 스위치를 기동하도록 설정한 인텔 i7 프로세서와 메모리 4G로 구성된 에이전트용 컴퓨터와 같은 설정으로 구성된 매니저 기동용 컴퓨터로 구성된다. 사용된 운영체제는 Windows 7/64bit이다. 네트워크 매니저의 수는 10명으로 제한하였고, 세 종류의 프로토콜에 대한 프로세싱 시간은 윈도우 내장 시스템 함수를 이용하여 측정하였다.

4.2 장비 구성 정보 변경 질의의 처리율 비교

이 실험은 여러 개의 NETCONF 매니저가 있다고 가정할 때, 각 NETCONF 매니저가 장비에 대한 구성 정보를 변경하거나 읽어오는 명령의 횟수에 대한 실제 처리율을 비교하는 것이다. 즉 장비의 설정 정보를 서브트리 기반으로 구성하는 데이터 모델링 기법이 다른 모델링 방법과 부분 잠금(partial lock)에서 어느 정도의 처리율과 처리 속도를 보이는지 확인하는 것이다. 실험은 명령어의 수를 100개에서 1000개까지 늘려가면서 명령 수행 성공(OK Message) 메시지를 받은 횟수를 측정하였고, 명령어의 종류는 <get-config>와 <edit-config>로 제한하였다. 각 명령어의 개수마다 10회의 실험을 반복하였다.



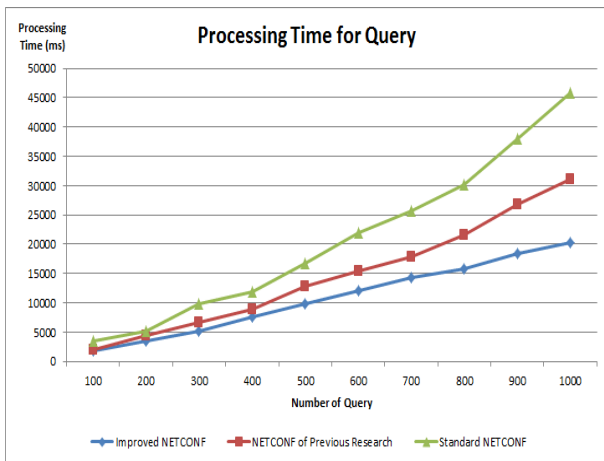
실험 1회당 XML 문서의 수: 100개에서 1000개까지 증가
 1개의 XML 문서 크기: 3Kbyte 미만
 실험 횟수: 각 명령어 수마다 10회의 반복 실험
 CPU 점유율: 2% 미만(세 방식 모두 동일)

Fig. 9. Processing Rate for Query

Fig. 9에 나타나있는 결과를 보면 본 논문의 기법은 명령어가 1000개에 이르는 시점에서도 약 95%의 처리 성공률을 보여주며, 평균적으로는 97%의 명령어 처리율을 보여준다. 데이터 종속성에 기반하는 데이터 모델링 기법을 사용하여 NETCONF도 평균 93% 이상의 처리율을 보인다. 표준 NETCONF는 명령어가 700개 이상인 시점부터 처리율이 다소 하락하여 평균적으로 87%를 보여주는데 10개의 매니저에 대한 1개의 에이전트 응답에 있어서 나쁜 성능은 아니다. 다만 실험에 사용된 명령어가 두 가지로 제한되지 않고 더 많은 종류가 사용되어 장비 설정 데이터에 대한 부분 잠금이 더 많이 요구되는 실험을 수행한다면 표준 NETCONF의 경우에는 더 많은 성능 하락이 예상된다. 더욱이 이 실험에서 매니저 사이의 우선순위는 구분하지 않았기 때문에 설정 정보의 후보 문서가 폐기(discard)되는 확률도 상대적으로 낮아서 전체적으로 고른 처리율을 보인다. 그러나 본 논문에서 두 가지 기법을 추가한 NETCONF가 나머지 NETCONF보다 평균에서 각각 4% 및 10% 정도 높은 처리율을 보이는 것은 확인할 수 있었다.

4.3 질의에 대한 시스템 처리 속도 비교

이 실험은 질의의 숫자 증가에 따른 NETCONF 에이전트의 반응 속도를 비교하는 실험으로 매니저에서 질의를 보내고 에이전트에서 처리한 후 응답을 받을 때까지의 시간을 측정하였다. 이 시간에 중요한 부분은 XPath 프로세싱 시간으로서, XML 문서의 형태로 저장되는 장비의 설정 데이터에 대해 XPath 표현식이 처리되는 시간을 의미한다. 본 논문에서 제안한 기법을 적용한다면 장비의 설정 데이터는 XML 문서를 테이블 형태의 주소로 표현한 Table 1과 같은 상태로 저장되어 있을 것이다. 이와 같은 데이터를 만드는 시간을 포함한다면 전체적인 처리 속도는 달라지겠지만 이 실험에서는 1차로 데이터가 모델링되어있는 것을 가정하고 이후 명령어에 대한 응답이 도달하는 시간만을 측정하였다. 실험에 사용된 정보들은 다음의 사각 박스에 정리하였다.



단일 <edit-config> 처리 시간(3Kbyte 기준): 평균 약 0.021초
 단일 <get-config> 처리 시간(3Kbyte 기준): 평균 약 0.023초
 매니저 수: 10개 우선순위 무시, 에이전트 수: 1개
 질의의 수: 100개에서 1000개까지 증가
 CPU 점유율: 2% 미만(세 방식 모두 동일)

Fig. 10. Processing Time for Query

질의의 수가 100개 미만일 경우에는 단일 명령어 처리 평균값에 질의의 수를 곱하는 결과가 나타난다. 즉 Fig. 10에 보인 것과 같이 본 논문에서 개선한 NETCONF와 나머지 두 종류의 NETCONF가 명령어 200개까지는 단일 질의 처리 속도인 20ms에 질의의 수를 곱한 것보다 조금 더 많은 시간을 소모하고 있다. 질의가 300개 이상 시점으로부터 표준 NETCONF는 처리 시간이 크게 상승하기 시작하여 질의가 600개가 될 시점에서는 단일 질의의 평균 처리 속도보다 1.5배 이상 더 걸리는 0.037초, 즉 37ms(milli second) 이상을 소모하게 된다. 한편 선행 연구의 NETCONF도 VTD-XML을 적용하지 않았고, 또한 데이터 모델링 기법을 서브트리 기반한 테이블 형태의 저장 방식을 사용하지 않아서 질의의 숫자가 늘어날 때마다 질의에 대한 데이터를 추출하는 시간이 증가한다. 특히 질의의 수가 500개를 지나는 시점에서 단

일 질의의 당 약 26ms의 시간을 소모하고 1000개가 되는 시점에서는 3.1ms를 소모한다. 본 논문에서 제시하는 기법으로 개선된 NETCONF의 경우에는 질의가 1000개가 되는 시점에서도 24ms 정도로 단일 질의를 처리하는 시간과 큰 차이가 없다. 오히려 질의 수가 적을 때는 큰 차이가 없지만 질의 수가 늘어날수록 장비 설정 정보를 서브트리 기반으로 저장하여 모델링한 데이터의 효율성이 높아지고 추가적으로 VTD-XML의 데이터 추출 성능이 높아지기 때문인데, 구체적으로는 장비의 설정 정보를 확인할 때 서브트리의 루트까지를 확인하면 종속 관계에 있는 데이터를 직접 찾을 수 있으며, 부분 잠금 또한 서브트리를 기준으로 빠르게 진행되기 때문에 성능의 하락이 거의 없다. 최종적으로 100개에서 1000개까지의 질의를 처리하였을 때 누적 평균 시간은 본 논문에서 개선한 NETCONF가 선행 연구의 NETCONF보다는 3.9초 빠르고 표준 NETCONF보다는 10.4초 빠르다.

다만 앞서 언급한 바와 같이 본 논문에서 제시하려는 개선된 NETCONF의 경우 1차적으로 장비 설정 데이터를 모델링하는 시간이 요구되는데 이 시간은 실험 결과에 나타나지 않았는데, 데이터 모델링을 위한 시간 소모와 모델링을 위한 CPU 점유율과 같은 부분은 NETCONF의 최종 목표인 사람의 개입을 거의 최소화하는 네트워크 관리에 대해서 상대적으로 적은 오버헤드라고 생각된다.

5. 결론 및 향후 과제

본 논문에서는 이질적인 장비로 구성되는 현대의 네트워크를 중앙 관리하기 위해 적합한 NETCONF 프로토콜의 단점을 개선할 수 있는 두 가지 기법의 적용을 제시하였다. 2006년에 처음 표준이 발표된 NETCONF는 계층별로 꾸준한 개정과 다양한 연구들이 존재하여 왔고 많은 문제점도 해결되었지만, Content 계층에서 XML 문서 처리를 효율적으로 하는 방법은 아직도 연구가 진행되고 있다. 현재 Content 계층에서의 데이터 모델은 YANG을 통해 XML 문서 형태로 표현하는 것이 표준화되었으나 XML 문서의 크기가 숫자, 또는 명령어를 전송할 수 있는 NETCONF 매니저가 많은 경우에 XML 문서에서 필요한 정보를 추출하거나 저장하는 작업의 처리 속도는 상대적으로 느리다.

본 논문에서는 이런 단점을 개선하고자 데이터 간 종속성 중심으로 서브트리를 구성하여 데이터 모델링을 하는 기법을 Content 계층의 데이터 모델링 기법으로 적용하고, XPath 질의에 대해 확장된 VTD-XML을 적용한다. 이를 통해 XML로부터 데이터를 효율적으로 추출하고 모델링 하는 방법을 제안한다. 제안 기법을 적용한 개선된 NETCONF와 데이터 종속성에 기반한 데이터 모델링만을 적용한 NETCONF, 그리고 표준 NETCONF를 비교하는 실험을 수행하였다. 실험은 질의의 개수를 증가시키면서 처리를 성공하는 비율과 질의가 처리되는 속도를 주요 인자로 하였다. 결과로서 본 논문에서 제시하는 기법을 적용한 개선된 NETCONF가 질의 처리율에서는 다른 두 기법에 비해 평균적으로 4%와 10%

가 높다는 것을 확인하였으며, 질의를 처리하는 누적 평균에서는 다른 두 기법에 대해 각각 약 3.9초와 10.4초 빠름을 확인할 수 있었다.

향후 연구과제로는 다양한 인자인 CPU 점유율, 메모리 사용률, 다수의 매니저와 다수의 에이전트 같은 환경에서 실험을 수행하여 본 논문에서 제시하지 못한 개선 기법의 장단점을 더욱 구체적으로 확인하고, 실제 장비에 적용하는 것 또한 다음 연구 내용으로 고려하고 있다.

References

[1] Y. M. Lee and J. K. Lee, "Improving and Optimizing the Operation Layer Algorithm of NETCONF Protocol," *International Conference on Advanced Information Networking and Applications*, DOI: 10.1109/WAINA.2014.79, pp.449-455, 2014.

[2] Y. M. Lee, M. Y. Cha, and J. K. Lee, "Development of Update Methods for Configuration Data of NETCONF Protocol considering Multiple Network Administrators," *Journal of Korean society for Internet Information*, Vol.14, Issue 5, 2013.

[3] K. H. Shin, and H. C. Kang, "Subtree-based XML Storage and XPath Processing," *The KSII Transactions on Internet and Information System*, Vol.4, No.5, pp.877-894, 2010.

[4] C. Subhashini and A. Arya, "A Framework For Extracting Information From Web Using VTD-XML's XPath," *International Journal on Computer Science and Engineering*, Vol.4, No.3, pp.463-468, 2012.

[5] X. Lan, J. Su, and J. Cai, "VTD-XML-Based Design and Implementation of GML Parsing Project," *Information Engineering and Computer Science*, DOI: 10.1109/ICIECS.2009.5363857, pp.1-5. 2009.

[6] M. J. Choi, et al., "XML-based Configuration Management for IP Network Devices," *IEEE Communications Magazine*, Vol.41, No.7, pp.84-91, 2004.

[7] Yang [Internet], http://www.netconfcentral.org/static/slides/yangtutorial/yang_getting_s24tarted2.html

[8] Y. Chang and D. Xiao, "Design and Implementation of NETCONF-Based Network Management System," *FGCN '08. Second International Conference on*, DOI: 10.1109/FGCN.2008.121, pp.256-259, 2008.

[9] J. Huang, B. Zhang, G. Li, X. Gao, and Yan Li, "Challenges to the New Network Management Protocol-NETCONF," *Education Technology and Computer Science, ETCS '09. First International Workshop on*, pp.832-836, 2009.

[10] S. M. Yoo, "Performance Improvement Methods for NETCONF-Based Configuration Management," *Computer Science*, pp.242-252, 2006.

[11] J. Schonwalder and M. Bjorklund, "Network configuration management using NETCONF and YANG," *IEEE Communications Magazine*, Vol.48, No.9, pp.166-173, 2010.

[12] J. Yu and I. Al, "An Empirical Study of the NETCONF Protocol," *IEEE Computer Society Magazine*, pp.253-258. 2010.

[13] M. S. Lee, "Improved Performance of network configuration management and System base on NETCONF," *Korea Communication Science Society Journal*, Vol.48, No.9, pp.787-790, 2008.

[14] E. Apostolos Nikolaidis, et al., "Management Traffic in Emerging Remote Configuration Mechanisms for Residential Gateways and Home Devices," *IEEE Communications Magazine*, Vol.43, Issue 5, pp.154-162, 2005.

[15] Y. M. Lee and J. K. Lee, "Development Renewal Method for Device Configuration Data of NETCONF Protocol considered Multiple Network Managers," *IEEK Summer Conference*, Paper CD, 2013

[16] Y. Gao. B. Zhang, G. Li, and Y. Li. "The comparison and analysis of tree data model and table-like data model based on NETCONF," *Informatics in Control, Automation and Robotics(CAR), International Asia Conference*, Vol.2, pp. 75-78, 2010.

[17] M. Golling, R. Koch, P. Hillmann, R. Hofstede, and F. Tietze, "YANG2UML: Bijective transformation and simplification of YANG to UML," *Network and Service Management (CNSM), 10th International Conference*, DOI: 10.1109/CNSM.2014.7014179. pp.300-303, 2014.



이 양 민

e-mail : yangwenry@dau.ac.kr

2000년 동아대학교 컴퓨터공학부(학사)
2002년 동아대학교 컴퓨터공학과(석사)
2006년 동아대학교 컴퓨터공학과(박사)
2003년~2014년 동아대학교 컴퓨터공학
시간강사

2015년~현 재 동아대학교 컴퓨터공학과 정보기술연구소 조교수
관심분야 : IoT, WSN, Cloud Computing, NETCONF, SDN, etc



이 재 기

e-mail : jklee@dau.ac.kr

1974년 영남대학교 전자공학과(공학사)
1983년 영남대학교 전자계산학과
(공학석사)
1990년 동경대학 전자정보공학과
(공학박사)

1984년~1990년 한국전자통신연구원 연구원

1990년~현 재 동아대학교 컴퓨터공학과 교수

관심분야 : IoT, SDN, Ubiquitous Computing, Sensor Network, MANET, Distributed Monitoring System, etc.