

NETCONF 기반의 시스템 및 네트워크 구성관리 성능 향상

정회원 이명숙*, 유선미**, 종신회원 주홍택*, 후원기***

Performance Improvement of Systems and Network Configuration Management Based on NETCONF

Myung-Suk Lee*, Sun-Mi Yoo** *Regular Members*
Hong-Taek Ju*, James W. Hong*** *Lifelong Members*

요 약

IETF의 NETCONF 워킹그룹에서는 구성관리의 상호운용성을 보장하기 위하여 구성관리표준 프로토콜인 NETCONF에 대한 표준을 제정하고 있다. 이러한 구성관리에서는 상호운용성뿐만 아니라 성능도 구성관리에서 중요한 요소 중의 하나이지만 현재까지는 성능에 대한 논의가 많이 이루어지고 있지 않은 실정이다. 이러한 점을 고려하여 본 논문에서는 NETCONF 프로토콜의 성능을 측정하고 성능을 향상시킬 수 있는 방안을 제시한다. 또한 제시된 방안에 대한 성능검증 결과를 제시한다.

Key Words : Configuration Management, NETCONF, Performance Evaluation, XML, RPC

ABSTRACT

IETF's NETCONF WG has taken efforts in standardizing configuration management protocol, which allows high interoperability of configuration management. In addition to interoperability, high performance is also required in configuration management, but many researches have often discarded the performance aspect of it. In order to fill that void, this paper proposes methods to improve performance with layers of NETCONF. It demonstrates and compares the performance evaluations to verify the efficiency of the proposed methods.

1. 서 론

네트워크 구성관리는 네트워크를 구성하는 각 장치의 동작을 규정하는 운영 값을 설정하고 설정된 값을 수집하여 분석하는 작업이다. 예를 들면 라우터의 라우팅 파라미터를 설정하거나 침입탐지시스템이나 파이어월의 보안 설정 값을 설정하고 모니터링하는 작업이다. 구성관리는 지역적으로 분산되어 있는 네트워크 장치들을 중앙에서 관리를 해야 하

므로 원격 관리가 필수적이다.

원격에 있는 장치에 대한 구성관리를 위해서 IETF(Internet Engineering Task Force)^[1]에서 NETCONF^[2] 표준안을 제시하였다. NETCONF 표준은 현재 네트워크가 다수의 회사에서 제작한 다양한 종류의 장치가 서로 연결되어 있는 점을 고려하였다. 따라서 이 표준을 적용하게 되면 구성관리를 효과적으로 수행할 수 있다.

구성관리는 다수의 장치를 대상으로 하기 때문에

※ 이 논문은 2007년 정부(교육과학기술부)의 재원으로 한국학술진흥재단의 지원을 받아 수행된 연구임(KRF-013-D00087)

* 계명대학교 컴퓨터공학과 컴퓨터네트워크 연구실(mslee@kmu.ac.kr), ** 삼성전자 연구원, *** 포항공과대학교 컴퓨터공학과

논문번호 : KICS2008-07-326, 접수일자 : 2008년 7월 24일, 최종논문접수일자 : 2008년 9월 1일

표준에 의한 상호연동성 뿐만 아니라 구성관리에 대한 효율성도 매우 중요하다. 예를 들면 많은 수의 장치에 대한 설정 값을 모니터링 해야 하는 경우에 효율적으로 설정 값을 모니터링하지 못하면 적절한 결과를 얻을 수 없다. 또한 구성관리에서 장치의 운영 설정 값을 변경하는 동안에 네트워크가 불안정해 질 수 있기 때문에 신속한 구성관리 완료가 필요하다.

NETCONF 표준은 기능적인 측면에서는 충분히 논의되어 표준으로 만들어졌으나 효율적인 측면에서의 기술적 검토가 매우 부족한 상태이다. NETCONF 표준에 대하여 성능을 측정하여 분석한 결과도 없고 NETCONF 표준을 효율적으로 사용하기 위한 방안도 없다. 본 논문에서는 NETCONF 표준의 성능을 향상시켜 구성관리의 효율성을 높일 수 있는 방안을 제시한다. NETCONF 표준에서는 OSI나 인터넷 모델과 같이 구성관리 프로토콜을 기능별로 세분화하여 계층(Layer)별로 나누어 놓았다. 각 계층별로 성능을 향상시킬 수 있는 방안을 제시하고 성능을 측정하여 제시한 방법에 대한 효과를 검증한다. 제시된 방안에는 표준을 준수하면서 성능을 향상시킬 수 있는 방법과 표준에는 제시되어 있지 않지만 성능을 향상시킬 수 있는 방법이 모두 포함된다.

본 논문의 구성은 다음과 같다. 서론에 이어 II 장에서는 관련 연구를 살펴보고, III장에서는 구현 및 성능측정 환경을 설명한다. IV장, V장, VI장에서는 NETCONF의 3개 계층인 Application 프로토콜 계층, RPC 계층, Operation 계층의 성능향상에 대하여 각각 설명한다. 마지막 VII장은 결론 및 향후 연구방향에 대하여 설명한다.

II. 관련연구

이번 장에서는 관련연구로서 NETCONF 프로토콜을 설명하고 네트워크관리 프로토콜의 성능에 관한 기존의 연구를 살펴본다.

2.1 NETCONF 프로토콜

NETCONF는 매니저와 에이전트 간의 간소화된 RPC 메커니즘 기반의 통신 기능을 제공한다. 에이전트와 매니저 사이에 전송되는 메시지를 구조화된 XML 형식으로 정의하고 있다. 이러한 구조화된 XML 메시지를 통하여 에이전트로부터 구성 정보를 가져오고, 수정하는 구성관리 서비스를 RPC(Remote

Procedure Call) 방법으로 제공하게 된다. 즉, 요청 메시지에 수행하고 싶은 관리동작 이름을 XML 태그 안에 명시하여 보내면 이 메시지를 받은 쪽에서는 자신의 XML 파서를 통하여 관리동작 이름을 추출하고 그 관리동작을 실행시킨 후 수행 결과를 응답 메시지에 담아서 보내게 된다.

전송되는 메시지를 4가지 계층으로 나누어 정의함으로써 다양한 환경의 구성관리 요구 사항을 충족시키고 있다. 표 1은 NETCONF의 4계층에 대한 정의와 각 계층에 포함되는 주요 XML 태그 및 예를 보여준다.

표1에서 Application 프로토콜 계층은 구성관리를 위한 전송기능을 담당하고 RPC 계층은 구성관리의 요청과 응답기능을, Operation 계층은 수행해야 할 관리동작 기능을 담당한다. Content 계층은 구성관리 데이터 자체이며 이 계층은 표준화의 대상이 아니다.

그림 1은 'edit-config' 관리동작의 요청을 표현하는 NETCONF 프로토콜 메시지의 예로써 NETCONF에서 정의하고 있는 4가지 계층 중 Application 프로토콜

표 1. NETCONF의 계층적 구조

계층	내용(태그 또는 예)
Application 프로토콜	메시지들을 전달하는 전송 프로토콜의 사용방법 (BEEP, SSH, HTTP 등)
RPC	RPC 관리동작에 대한 요청과 응답을 나타냄 (<rpc>, <rpc-reply>, <rpc-error> 등)
Operation	NETCONF 에서 정의한 RPC 를 제공할 관리동작들을 나타냄 (<get-config>, <edit-config>, <copy-config>, 등)
Content	XML 형태의 관리 구성 정보 데이터 (보안설정, 라우팅 정보 등)

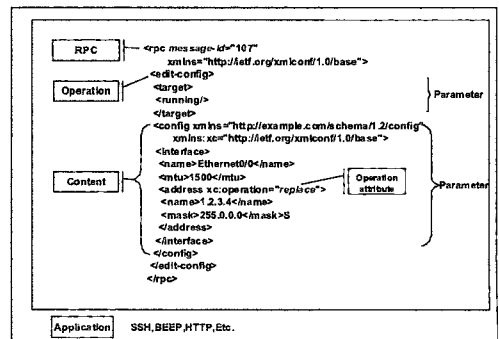


그림 1. NETCONF 프로토콜의 예

계층을 제외한 나머지 세 개의 계층과 메시지와의 관계를 보여 준다. 이 메시지는 실제 동작하는 네트워크 장치의 구성 정보인 <running>의 지정된 인터페이스 카드와 NETMASK의 구성 정보 값을 수정하는 예이다.

NETCONF는 계층으로 나누어 정의하고 있으며, 전송 프로토콜인 Application 프로토콜에 대해서는 중립적이다. 따라서 어떠한 전송 프로토콜도 지원하는 것이 기본 개념이다. 하지만 현재는 지원 프로토콜로써 SSH^[4], BEEP^[5] 그리고 SOAP over HTTP^[3]에 대해서만 고려한다. Application 프로토콜 계층은 매니저와 에이전트의 연결(connection)관리를 수행한다. 매니저와 에이전트가 관리수행을 위해서 복수의 Application 프로토콜을 사용하여 연결이 가능하다. 복수의 연결을 묶어서 세션(session)이라 부르며 한 세션에서 논리적으로 연결된 관리행위가 수행된다.

NETCONF 프로토콜은 RPC기반의 통신 모델을 사용한다. Application 프로토콜에 종속되지 않는 요청과 응답의 구조를 제공하기 위해 <rpc>, <rpc-reply>를 사용한다. 또한 매니저가 요청한 메시지에 대해 에이전트가 제어할 수 있는 방법을 <rpc> 태그 안에 요청 메시지로 명시한다. 이는 'message-id'의 정의를 통해 제공하게 되는데, 메시지 번호를 사용하여 이전에 요청한 메시지를 제어 할 수 있다. 즉, 'message-id'는 매니저와 에이전트의 접속에 의해 세션이 이루어진 후, 해당 세션을 통해 주고 받는 메시지를 관리하기 위한 번호라고 할 수 있다.

NETCONF는 SNMP에서 제공하고 있는 'get', 'set', 'trap'에 대응할 수 있는 'get-config', 'edit-config', 'notify' 관리동작 뿐만 아니라, 트랜잭션(transaction) 처리 즉, 관리동작 수행 중에 에러가 나면 수행 전의 상태로 돌아가는 'rollback', 또는 에러가 난 곳에서 멈추고 에러 메시지를 보내는 'stop-on-error' 등이 정의된다. 또한 구성 정보의 일관성을 유지하기 위해서 여러 매니저가 동시에 장치의 구성 정보를 수정하고자 할 때, 수정되고 있는 구성 정보에 대해서 잠금(lock)을 걸 수 있는 관리동작이 정의된다. SNMP에서 제공하지 않는 관리동작들이 새롭게 정의 되었다.

현재 관리 구성 정보인 content는 관리되어야 할 네트워크 장치마다 그 내용이 다르게 정의되기 때문에, 구성 정보 내용은 네트워크 장치 업체에 의존적이게 된다. 따라서 구성 정보를 정의하는 언어와 그 내용에 대한 표준화 작업은 별도로 이루어져야 한다.

2.2 네트워크관리 성능연구

네트워크 관리 분야에서 수행되었던, 성능에 관련된 연구들로 먼저, 네덜란드의 Twente 대학에서 수행되었던 웹 서비스와 SNMP의 성능을 비교한 연구^[11]를 살펴본 후, 구성관리 분야에서 수행되었던 성능 연구^[10]를 소개한다. 두 개의 연구는 NETCONF가 XML에 기반을 두고 있으므로 밀접한 관련이 있다.

네덜란드에 있는 Twente 대학교의 Aiko Pras와 그의 연구원들은 웹 서비스에 관련된 여러 연구를 해왔다. 여러 연구들 중에서, 그들은 SNMP와 웹 서비스를 이용한 네트워크 관리의 성능을 비교 분석하는 연구도 수행하였다^[11]. 해당 연구는 성능 비교를 위해서, 대역폭 사용량, CPU 시간, 메모리 사용량 그리고 수행시간을 측정하였다. 또한, 그들은 성능 테스트에 사용하기 위해 웹 서비스를 기반으로 하는 네트워크 관리 시스템을 프로토타입 형태로 구현하였다. 구현한 시스템과 SNMP를 사용하는 관리 시스템을 이용하여 비교하는 테스트를 수행하였다. 해당 연구의 결과는 SNMP와 웹 서비스를 이용하여 네트워크 관리시 대역폭 부분에서 확인한 차이를 보여준다. 또한, 그들은 SNMP가 아주 적은 수의 객체의 값을 읽는 경우에는 웹 서비스보다 더 좋은 성능을 보여준다고 결론을 내렸다. 하지만, 많은 수의 객체들의 값을 읽을 때는 웹 서비스가 SNMP보다 두드러지게 좋은 성능을 보여준다는 결론을 내렸다. 그들은 해당 연구를 통해서 웹 서비스를 이용한 네트워크 관리 방법은 큰 규모의 네트워크에 훨씬 더 적절하다고 제시하였다.

XML 기술은 IETF, UPnP 그리고 DSL 포럼들의 표준화 및 프로토콜로 많이 사용되어 왔다^[12]. 해당 XML 기술은 XML 데이터를 사용하므로서 긴 길이의 데이터를 지니며, 전송 프로토콜로는 SOAP RPC를 많이 사용한다. 많은 장비들을 관리해야 하는 구성관리 환경에서는 패킷의 크기가 문제가 될 수 있다. Apostolos는, 패킷의 크기를 줄이기 위해 해결책으로 Lempel-Ziv 압축 알고리즘 사용을 제시하였다^[10]. Lempel-Ziv 압축 알고리즘^[13]을 사용함으로써 얻는 효과를 실험을 통해서 증명하였다. 실험은 홈 네트워크 장비들을 관리하는 환경에서 수행되었다. 압축 알고리즘의 사용으로 줄어든 패킷의 크기가 줄어드는 것을 비교 실험으로 증명하였으며, 수행 시간을 측정하였다. 수행 시간은 네트워크의 속도를 변화시키면서 측정하였다. 해당 연구에서는 XML기술을 사용하는 구성관리에 압축방식은 큰 효

과를 제공한다고 결론을 내렸다. 또한, 압축과정이 수행시간에 많은 지연을 줄 수 있을 것 같지만 실제로는 큰 영향을 주지 않으며, 수행시간은 네트워크의 속도에 오히려 많은 영향을 받는다고 하였다.

III. 구현 및 성능측정 환경

NETCONF 기반의 구성관리 성능을 측정하기 위한 환경으로 이전 연구에서 개발한 XCMS (XML-based Configuration Management System)를 활용하였다⁶⁾. XCMS는 그림 2와 같이 NETCONF 매니저와 NETCONF 에이전트로 구성되어 있다.

이들 간의 전송계층의 연결은 표준에서 제시하는 SOAP over HTTP, BEEP 그리고 SSH를 지원한다. 실험을 위한 물리적 네트워크 연결은 100Mbps Ethernet으로 직접 연결되어 있다. 성능에 사용한 장치는 리눅스 시스템이며, 구성 정보는 리눅스 시스템 정보 중 네트워크 정보인 interface 정보를 사용하였다. 모든 실험에서의 응답 시간을 좀 더 정확하게 측정하기 위해 그림 3에 나온 그래프처럼 최소 1000번씩 측정을 하여 측정값들의 평균값을 이용하였다.

다음 장에서 제시하는 각 방법들은 XCMS를 기반으로 구현되었다⁹⁾. NETCONF 매니저는 NETCONF 에이전트와 통신하기 위해서 메시지를 전송한 후, 서로간의 통신이 가능해지면, 서로의 ca-

pability 내용을 담은 Hello 메시지를 전송하면서 NETCONF 통신을 시작한다. 매니저는 NETCONF 요청 메시지를 전송하면서 구성관리를 수행한다. 우리는 NETCONF 매니저가 요청 메시지를 전송하는 부분과 응답 메시지를 전송 받는 부분들에 대해서 응답 시간과 네트워크 사용량을 측정하였다.

측정결과, 각각의 전송프로토콜에서 iface의 개수를 증가시킬수록 응답시간과 네트워크 사용량은 조금씩 증가하였으나, Zlib 압축기술을 사용한 데이터 크기별 응답시간과 네트워크 사용량은 압축기술을 사용하지 않은 경우보다 데이터 크기가 커질수록 큰 차이를 보였다. 또한 subtree와 XPath에 대한 필터링의 수행시간과 메모리 사용량을 측정한 결과 합병이 필요한 메시지에서는 XPath를 사용하는 경우가 수행시간이 더 적게 걸렸고, 합병이 필요없는 메시지에서는 subtree를 사용하는 경우에 수행 시간이 더 적게 걸리는 차이를 보여주었다. 메모리 사용량에 있어서는 subtree는 작업을 수행함에 따라서 차이가 생기지만 XPath는 작업의 복잡성 및 내용에 관계없이 항상 동일한 메모리를 사용하는 것을 확인할 수 있었다.

IV. Application 프로토콜 계층 성능

NETCONF 표준에서 application 프로토콜 계층은 NETCONF 매니저와 에이전트 사이의 메시지 전송기능을 담당한다. Application 프로토콜 계층에서 사용하는 전송 프로토콜로 SOAP over HTTP, SSH 그리고 BEEP이 있다. 3가지 전송 프로토콜을 사용할 수 있지만 각각에 대한 성능적인 차이점을 성능측정으로 검증한다.

전송 프로토콜들의 성능을 측정하기 위해서 다양한 크기의 구성 정보를 읽는 작업을 수행하였다. NETCONF 매니저는 해당 작업을 수행하기 위해서, 설정 값을 읽어 오기 위한 'get-config' 명령을 수행한다. 에이전트로부터 받는 응답정보의 양을 변화시키기 위해서 물리적 연결정보인 interface의 구성 정보 중 네트워크 연결 수를 나타내는 iface의 개수를 2개씩 증가시켜 수행하였다. 그림 3은 3개 전송 프로토콜에 대해 수행한 실험 결과로 응답 시간의 변화를 나타낸다. 그림 4는 동일한 실험에 대하여 네트워크 사용량을 나타낸다.

그림 3의 그래프에 따르면, 예상외로 SOAP 프로토콜이 2000~2500msec 정도의 가장 적은 응답 시간이 소요되는 것을 확인할 수가 있다. 그림 4의 그래

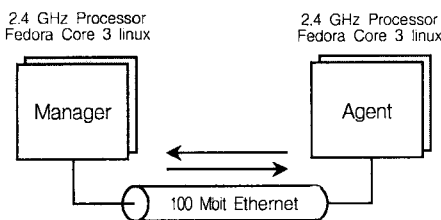


그림 2. 구성도

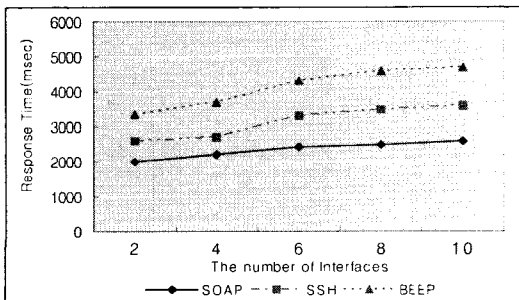


그림 3. 전송 프로토콜별 응답시간

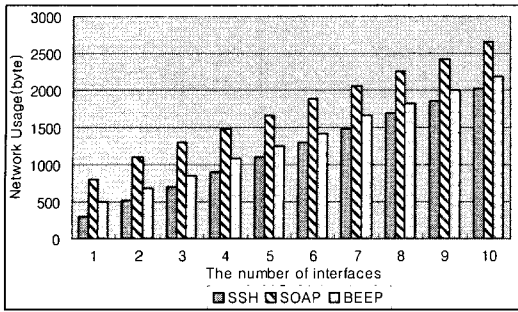


그림 4. 전송 프로토콜별 네트워크 사용량

프에서 보여주는 것처럼 SOAP을 통해 전송되는 메시지는 다른 전송프로토콜에 비해 400~600byte 정도의 큰 메시지를 전송한다. 보통 SOAP 메시지가 크기 때문에 긴 응답 시간을 나타낼 것으로 예상되지만 결과는 그렇지 않다. 이러한 결과는 각 전송 프로토콜의 통신방식이 다르기 때문에 나타난 결과이다. SOAP 프로토콜은 WSDL^[8]을 이용한 RPC 방식으로 관리동작을 바로 호출하는 방식을 사용한다. 반면에, SSH는 포트를 포워딩 하는 방식 및 인증 방식을 수행하면서 중간 수행 시간에서 오버헤드가 발생한다. BEEP은 peer-to-peer 방식으로 통신이 된다. 클라이언트도 메시지를 전송할 수 있으며, 비 동기식 메시지 통신이 가능하다. 매니저와 에이전트가 서로의 메시지 순서를 확인하는 메시지 전송이 필요하다. 이러한 점이 메시지 전송 횟수를 증가시키고 응답시간도 길어진다. BEEP는 메시지의 크기가 커질수록 응답시간이 다른 프로토콜들에 비해서 많이 길어지는 현상을 보여준다.

그림 4의 네트워크 사용량에서 SSH를 제외한 두 프로토콜은 XML 기술들로, 부가적으로 꼭 붙어야 하는 내용이 많기 때문에 사용량이 많은 순서로 SOAP, BEEP, SSH 순서이다. 특히, SOAP over HTTP는 크기가 작은 데이터를 전송 시 오히려 부가적으로 붙는 데이터가 전송하려는 데이터의 크기보다 더 클 수 있다. 네트워크 사용량과 응답시간의 관계는 앞 문단에서 설명하였다.

전송 메시지들이 네트워크 트래픽에 많은 영향을 준다면 효율적인 구성관리라고 할 수가 없다. XML을 기반으로 동작하는 NETCONF의 application 프로토콜 계층에서는 트래픽의 양을 줄일 수 있는 방안 제시가 필요하다. XML 메시지의 크기를 줄일 수 있는 방법으로는 binary XML, 압축 방법등 여러가지가 존재한다. II장 관련연구에서 살펴보았던 구성관리 성능에 관한 연구 결과^[10]와 비교하기 위해서

NETCONF가 메시지를 전송할 때의 트래픽양을 줄이는 방법으로 압축 방법을 사용하였다. XML 메시지들의 특징은 데이터 값의 앞 뒤로 붙는 긴 태그들과 반복적으로 사용되는 단어들이다. 이 특징을 이용해서 XML 메시지를 압축하는 기술 중 하나인 Zlib을 이용한다. 각 프로토콜의 payload 부분에 반복적으로 사용되는 태그들은 압축의 효과가 크다.

그림 5와 그림 6은 압축을 수행하지 않고 데이터를 전송하는 경우와 압축 수행 후 데이터를 전송하는 경우의 네트워크 사용량과 응답시간을 측정한 도표이다. 압축의 효과를 알아보기 위해서 다양한 크기의 데이터에 대해서 수행하였으며, 전송 프로토콜로는 XML 기술 중 대표적인 SOAP over HTTP를 사용하였다.

그림 5에서 보면, 5000bytes 이하 작은 크기의 데이터를 압축 할 경우에 눈에 띄는 큰 효과는 없다. 하지만, 데이터의 크기가 10000bytes 이상이 될수록 XML 메시지의 중복되는 데이터가 많아지게 되어 압축효과가 매우 크다. 또한, 우리는 압축을 수행함으로써 얻을 수 있는 오버헤드부분을 측정하였다. 그림 6의 결과에 따르면 데이터의 크기가 2000bytes 이하인 경우에는 압축에서 발생하는 오버헤드로 오히려 수행 시간이 길다. 하지만, 데이터의 크기가

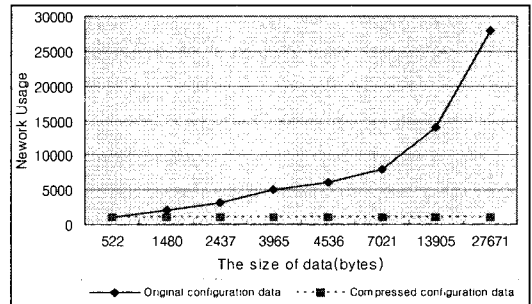


그림 5. 압축에 대한 데이터 크기 변화

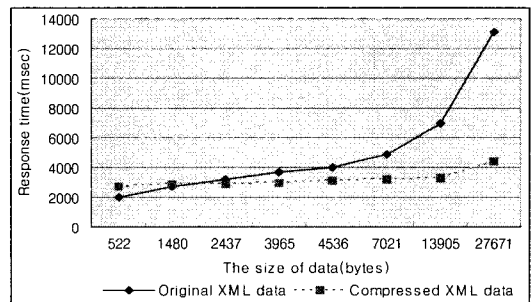


그림 6. 압축에 대한 응답시간의 변화

10000bytes 이상에서 수행 결과는 다르다. 구성관리 에 관한 다른 연구 결과와 동일하게 압축 방법은 데이터의 크기가 클수록 네트워크 사용량과 응답 시간 모두에서 효과가 증가하는 것을 확인 할 수 있다. 따라서 2000bytes 이하의 매우 적은 데이터를 사용할 때에는 압축 방식이 오히려 단점으로 작용할 수 있지만, 많은 구성 정보를 관리할 때에는 압축 방법이 성능을 많이 향상 시킨다.

V. RPC 계층 성능

NETCONF의 RPC 계층은 NETCONF 매니저로부터의 구성관리 수행 요청 메시지와 NETCONF 에이전트로부터의 수행결과인 응답 메시지임을 나타내는 요청과 응답역할을 수행한다. RPC 계층에서는 XML 태그인 <rpc>로 수행 관리동작을 감싸서 NETCONF의 요청 메시지를 나타내고, <rpc-reply>로 결과 값을 감싸서 NETCONF의 응답 메시지를 나타낸다. 현재 NETCONF에서 제시하고 있는 <rpc> 요소 아래에는 단 한 개의 관리동작만이 들어간다. 즉, NETCONF 매니저가 한 개의 관리동작을 지닌 rpc 메시지를 생성하여 전송하고, NETCONF 에이전트는 전송 받은 메시지의 관리동작을 수행한 후에, 결과 값인 rpc-reply 메시지를 생성하여 전송한다. 매니저는 그 뒤에 새로운 관리동작을 수행할 메시지를 생성한 후 전송한다. 즉, 매니저는 에이전트의 수행 결과가 완료되어 응답 메시지가 올 때까지 그 뒤의 작업을 수행 못하고 기다려야 한다. NETCONF에서는 최근에 해당 방식을 RPC 계층에서의 비 효율성으로 지적하고, 'pipelining' 이라는 새로운 방법을 제시하였다. 하지만, 현재까지 NETCONF에서는 단지 'pipelining' 방법을 사용할 수 있다'라는 내용만을 제시할 뿐, 해당 방법에 대한 지침서 및 성능, 해당 방법을 사용함으로써 얻을 수 있는 효과에 대한 결과들을 제시하고 있지 않다. 이번 장에서는 RPC 계층에서의 효율성을 검증하고, 좀 더 효율적으로 수행할 수 있는 방안들을 제시한다.

NETCONF에서 제시하고 있는 'pipelining' 방식은 기존의 'pipelining' 방식과 유사하다. NETCONF 에이전트가 관리동작을 수행하는중에, NETCONF 매니저로부터 메시지를 전송 받을 수 있다. 하지만, NETCONF의 특성 상, 관리동작은 들어오는 순서대로 수행되어야 한다. 즉, NETCONF 에이전트는 이전 관리동작이 완료되면, 해당 관리동작이 완료되기 전에 전송 받은 메시지들 중 가장 먼저 받은 메시지를

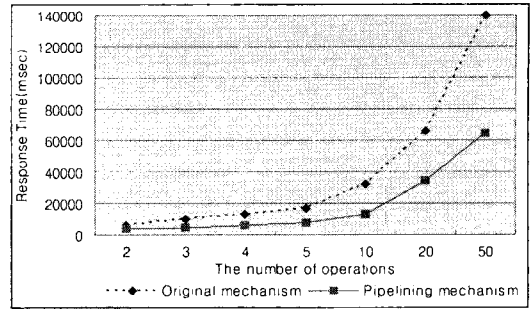


그림 7. pipelining의 응답시간의 변화

처리한다.

'pipelining' 방식의 효율성을 측정하기 위해서, 우리는 명령어의 개수를 다르게 하여 응답 시간을 측정하였다. 또한, 'pipelining' 방식을 사용하지 않은 경우에는 매니저가 응답 메시지를 받는 즉시, 요청 메시지를 전송하는 방식으로 수행하여 측정하였다. 즉, 관리자가 메시지를 입력하고 생성하는 시스템 이외의 요소는 배제하였다. 전송 프로토콜은 SSH를 사용하였다.

그림 7에서 관리동작의 개수가 10이상 증가 할수록 'pipelining'의 응답 시간이 줄어드는 것을 확인할 수 있다. 그 이유는 이전 관리동작이 수행되는 동안에 다음 요청 메시지가 전송되므로, NETCONF 에이전트는 새로운 요청 메시지를 기다리고 있지 않아도 된다. 또한, NETCONF 매니저는 응답 메시지를 전송 받은 뒤에 새로운 요청 메시지를 전송하는 공백을 줄일 수 있으므로 위와 같은 결과가 나온다. 따라서 관리동작의 개수가 많아질수록 'pipelining'을 사용한 경우와 사용하지 않은 경우의 응답 시간의 값 차이는 더욱 커지는 것이다. 하지만, 'pipelining'을 사용함으로써 얻을 수 있는 위험성은 매니저가 메시지를 확인하기 전에 새로운 메시지를 전송해야 하므로 잘못된 수행을 하는 경우가 있을 수 있다. 이러한 경우를 위해서 NETCONF 매니저는 'discard-change' 관리동작이나 'roll-back' 기능을 제공해야 된다.

VI. Operation 계층 성능

NETCONF에서의 operation 계층은 실제적인 구성 정보 수행 내용들을 정의한다. Operation 계층은 구성 정보 관리에 필요한 'get'과 'set'의 역할을 지닌 여러 관리동작들을 제공하고 있다. 'set' 기능은 관리 대상인 네트워크 장비들의 구성 정보를 변경하므로써 네트워크 장비들의 상태에 직접적인 영향을 끼친다. 따

라서, 'set' 기능을 수행하는 관리동작들의 성능은 구성관리에 있어서 중요한 요소이다. NETCONF 프로토콜에서는 'edit-config'와 'copy-config' 두 관리동작이 구성 정보를 설정하는 기능을 수행한다.

NETCONF는 구성 정보의 특정 부분에 대해서만 관리동작을 수행하기 위하여 필터링 방식을 제시하고 있다. 특히, 구성 정보 값을 읽어 오는 'get'과 'get-config' 관리동작은 구성 정보에 대해서 두 종류의 필터링 방식을 제공한다. 해당 필터링 방식들 중 하나는 NETCONF에서 기본으로 제시하고 있는 'subtree filtering' 방식이고, 나머지 하나는 XML의 기술인 'XPath'^[7] 방식이다. 두 방식은 서로 비슷하면서도 많은 차이점을 보인다. 먼저, 'subtree filtering'은 NETCONF에서만 사용되고 있는 방식이다. 즉, NETCONF에서 만든 필터링 방식이다. 'subtree filtering'의 규칙은 파악하기가 많이 어려우며, 의미가 정확하지 않은 부분들이 많다. 또한, 정확한 사용방법의 예시가 많이 부족하여서 구현시 어려움이 있으며, 각 구현물마다 방식이 다소 차이가 있을 수 있다. 반면에, 'XPath'는 일반적인 'XPath'의 표현법을 그대로 사용한다. 'XPath'의 표현을 사용하기 때문에 사용하는 방법은 간단하지만 해당 표현 방법이 다소 복잡하다. 이러한 기능적 장·단점 이외에 성능적 측면에서의 수치적 증명이 필요하다. 왜냐하면, NETCONF에서는 'XPath'가 수행 시간 및 메모리 측면에서 다소 부담이 되는 특징을 지녔다며 'XPath'대신에 새로운 'subtree filtering'의 규칙들을 만들어서 제시하였다. 'XPath'는 수행 대상 메시지를 파싱하기 위해서 해당 내용을 DOM에 올려서 파싱 작업을 수행한다.

우리는 두 필터링 방식의 성능적 차이점을 알아 보기 위해서 실험을 수행하였다. 'get-config' 오픈레이션을 사용하였으며 두 경우로 나누어서 실험을 수행하였다. 첫 번째 실험은 한 번의 파싱 작업이 아닌, 여러 경우의 파싱 작업을 수행한 후, 해당 결과를 합치는 방식이다. 즉, 합병작업을 수행할 때의 두 필터링 방식의 응답시간과 메모리 사용량을 측정하였다. 두 번째 실험은 간단한 파싱 작업을 수행할 때의 두 필터링 방식에 대해서 동일하게 응답시간과 메모리 사용량을 측정하였다.

표 2는 필터링의 성능측정에 사용한 메시지를 보여주고 있다. 합병이 필요한 메시지는 eth0이면서 1000Mbps인 모든 네트워크 설정에 요청한 관리동작을 수행하라는 의미이다. 합병이 불필요한 메시지는 이름을 가지고 있는 모든 네트워크 설정을 의미한다. 각 요청 메시지의 수행 결과 메시지는 두 필

표 2. 필터링에 사용한 메시지.

	필터	Subtree Filtering
합병이 필요한 메시지	subtree	<pre><filter type="subtree"> <interfaces> <iface> <name>eth0</name> </iface> <iface> <mtu>1000</mtu> </iface> </interfaces> </filter></pre>
	XPath	<pre><filter type="xpath"> //name[.='eth0']//mtu[.='1000'] </filter></pre>
합병이 필요 없는 메시지	subtree	<pre><filter type="subtree"> <interfaces> <iface> <name/> </iface> </interfaces> </filter></pre>
	XPath	<pre><filter type="xpath"> //name </filter></pre>

표 3. 필터링의 수행시간과 메모리 사용량.

	필터	수행시간(ms)	메모리 사용량(KB)
합병이 필요한 메시지	subtree	2.085	1816
	XPath	1.901	1828
합병이 필요없는 메시지	subtree	1.716	1812
	XPath	1.954	1828

터링 방식에서 동일하다. 표 3에 나온 결과 값을 살펴보면, 두 실험의 성능 차이가 서로 다르다. 먼저, 첫 번째 실험인 합병기능이 필요한 경우에는 'XPath'를 사용하는 경우가 수행 시간이 더 적게 걸리는 것을 확인할 수 있다. 이는, 모든 구성 정보를 DOM에 올려놓고 파싱을 수행하는 'XPath'에서는 merge작업과 단순한 작업에 있어서의 차이가 적을 것이다. 하지만, 요청 메시지의 내용과 구성 정보를 일대일로 매칭하면서 수행하는 'subtree filtering'은 여러 작업을 합쳐야 하는 작업에 대해서는 좀 더 많은 수행 시간을 요구한다. 반면에, 간단한 수행을 요구한 두 번째 실험에서는 'subtree filtering'이 더 적은 시간이 걸리는 것을 확인할 수 있다. 이는 단순한 작업 임에도 DOM에 모든 메시지를 올리고 수행 하는 'XPath'의 특징 때문이다. 마

지막으로, 메모리 사용량에 있어서는, subtree는 작업을 수행함에 따라서 차이가 생기지만, 'XPath'는 작업의 복잡성 및 내용에 관계없이 항상 동일한 메모리를 사용한다. 또한, 모든 데이터를 메모리에 올리고 사용하는 'XPath'가 더 많은 양을 사용하는 것을 확인할 수 있다.

VII. 결론 및 향후 연구

시스템 및 네트워크의 설정 값을 중앙에서 관리하는 구성관리기가 급속히 확대되고 있다. 다양한 회사의 다양한 제품이 혼재하는 현재의 시스템 및 네트워크 구성을 감안하면 표준화된 구성관리 방안이 필요하다. NETCONF는 IETF에서 제정한 표준 구성관리 프로토콜이다.

우리는 NETCONF의 성능을 향상시키기 위한 몇 가지 방안을 제시하였고 성능측정을 통하여 제시한 방법을 검증하였다. 또한 NETCONF 프로토콜 사용에 있어서 선택 가능한 몇 가지 선택사항 중에서 어떤 방법이 성능측면에서 어떤 특성을 가지고 있느냐에 대한 대답을 제시하였다.

Scalability 측면에서는 Application 프로토콜 계층은 3개의 전송 프로토콜 중 SOAP이 다른 프로토콜에 비해 우수하였고, Zib를 이용한 압축을 함으로써 확장성이 좋을 수 있었다. RPC 계층에서는 Pipelining을 사용한 경우가 그렇지 않은 경우보다 확장성이 좋았고, Operation 계층에서는 두 필터링의 수행시간과 메모리 사용량을 측정된 결과 확장성은 비슷한 결과를 보였다.

본 논문에서 제시하는 성능결과를 기반으로 향후 NETCONF의 표준화에 기여하고자 한다. 또한 XCMS이외에 다른 구현에서도 동일한 결과가 도출되는지 검증하는 작업을 해외 연구자들과 연구교류를 통하여 확인하는 것도 향후 연구방향이다.

참 고 문 헌

[1] IETF, "Network Configuration," <http://www.ietf.org/html.charters/netconf-charter.html>.

[2] R. Enns, Ed, "NETCONF Configuration Protocol", RFC 4741, Dec. 2006.

[3] T. Goddard, "Using the Network Configuration Protocol (NETCONF) Over the Simple Object Access Protocol (SOAP)," draft-ietf-netconf-soap-06,

<http://www.ietf.org/internet-drafts/draft-ietf-netconf-soap-06.txt>, September 16, 2005.

[4] M. Wasserman, T. Goddard, "Using the NETCONF Configuration Protocol over Secure Shell (SSH)," <http://www.ietf.org/internet-drafts/draft-ietf-netconf-ssh-05.txt>, Oct. 2005.

[5] E. Lear, K. Crozier, "Using the NETCONF Protocol over Blocks Extensible Exchange Protocol," <http://www.ietf.org/internet-drafts/draft-ietf-netconf-beep-07.txt>, Sept. 2005.

[6] Hyoun-Mi Choi, Mi-Jung Choi, James W. Hong, "Design and Implementation of XML-based Configuration Management System for Distributed Systems," Proc. of the IEEE/IFIP NOMS 2004, Seoul, Korea, April 2004, pp. 831-844.

[7] W3C, "XML Path Language (XPath) Version 2.0," W3C Working Draft, November 2005.

[8] W3C, "Web Services Description Language (WSDL) Version 1.2" July 2002.

[9] Sun-Mi Yoo, Hong-Taek Ju, James Won-Ki Hong, "Web Services Based Configuration Management for IP Network Devices," Proc. of the IEEE/IFIP MMNS 2005, LNCS 3754, Barcelona, Spain, Oct., 2005, pp.254-265.

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

[11] A. Pras, T. Dreviers, R. v.d. Meent and D. Quartel, "Comparing the Performance of SNMP and Web Services-Based Management," IEEE eTNSM, Vol.1, No.2, Dec. 2004, pp.1-11.

[12] Mi-Jung Choi et al, "XML-based Configuration Management for IP Network Devices," IEEE Communications Magazine, Vol. 41, No. 7, July 2004. pp.84-91.

[13] J. Adiego, G. Navarro, and P. de la Fuente, "Lempel-Ziv Compression of Structured Text," IEEE Proc. Data Compression Conf.

이 명 숙 (Myung-Suk Lee)

정회원



2001년 2월 계명대학교 컴퓨터공학 학사
2003년 2월 계명대학교 컴퓨터공학 석사
2003년~현재 계명대학교 컴퓨터공학 박사과정
<관심분야> XML 기반의 네트워크 관리, 네트워크 모니터링

유 선 미 (Sun-Mi Yoo)

정회원



2004년 숭실대학교, 컴퓨터학 학사
2006년~포항공과대학교, 컴퓨터공학과 석사
<관심분야> XML 기반의 네트워크 관리, 에이전트 기술, 정책 기반의 네트워크 관리

주 흥 택 (Hong-Taek Ju)

종신회원

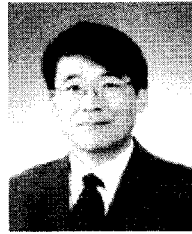


1989년 8월 한국과학기술원 전산학 학사
1991년 8월 포항공과대학교 컴퓨터공학 석사
2002년 2월 포항공과대학교 컴퓨터공학 박사
1991년 9월~1997년 2월 대우통신, 종합연구소, 선임연구원

2002년 9월~현재 계명대학교 컴퓨터공학과 조교수
<관심분야> Web-based Network Management, Network Monitoring, Mobile Device Management

홍 원 기 (James W. Hong)

종신회원



1983년 Univ. of Western Ontario, BSc in Computer Science
1985년 Univ. of Western Ontario, MS in Computer Science
1985년~1986년 Univ. of Western Ontario, Lecturer
1986년~1991년 Univ. of Wasterloo, PhD in Computer Science

1991년~1992년 Univ. of Wasterloo, Post-Doc fellow
1992년~1995년 Univ. of Western Ontario, 연구교수
1995년~현재 포항공과대학교 컴퓨터공학과 부교수
2003년~현재 한국통신학회 통신망운용관리연구회 위원장
2003년~현재 IEEE ComSoc CNOM Vice Chair
2004년~현재 IEEE ComSoc Director of On-Line Content
<관심분야> 네트워크 트래픽 모니터링, 네트워크 및 시스템 관리, Network Security