논문 13-38B-04-05

한국통신학회논문지 '13-04 Vol.38B No.04
http://dx.doi.org/10.7840/kics.2013.38B.4.266

소프트웨어정의네트워크 기반의 서비스 오버레이 네트워킹을 위한 네트워크 정책 제어기

조 진 용◆, 이 소 연*, 공 정 욱**, 김 종 원°

A Centralized Network Policy Controller for SDN-Based Service Overlay Networking

Jinyong Jo◆, Soyeon Lee*, JongUk Kong**, JongWon Kim°

요 약

본 논문은 인터넷과 같은 다중제공자(multi-provider) 네트워크 환경 하에서 패킷 플로우를 효과적으로 제어하기 위한 SDN(Software Defined Networking) 기반의 정책 제어기를 소개한다. 제안된 정책 제어기는 네트워크 가시성 정보를 이용해 가상링크 및 가상포트 등을 직관적으로 제어함으로써 효과적인 서비스 오버레이 네트워킹(service overlay networking) 환경을 실현한다. 또한, 논리적으로 구분된 다수의 주문형 가상망을 신속히 구성하고 동적으로 관리함으로써 응용에 최적화된 네트워킹 환경을 사용자에게 제공한다. 본 논문에서는 정책 제어기의 구조 및 특징을 소개한 후, 멀티캐스트를 위한 두 가지 서비스 응용을 예시한다. 또한, 해당 응용들을 이용한 네트워크 서비스의 구성 시간을 성능 평가함으로써 정책 제어기의 적용 가능성을 확인한다.

Key Words : Policy controller, software defined network, service overlay networking

ABSTRACT

In this paper, to manage the efficient control of IP packet flows crossing multi-provider networks such as Internet, we propose a SDN(Software Defined Networking)-based policy controller. The proposed policy controller leverages the visibility of underlying network and manages both virtual links and ports to inter-connect networking elements. The controller is capable of quickly composing multiple on-demand virtual networks and dynamically managing the composed networks, thus it can provide more flexible and optimized overlay networking environment to end-user applications. More specifically, we first look into the proposed structure and features of policy controller. With two kinds of service applications, we then verify the applicability of the proposed controller by evaluating its service composition time.

Ⅰ. 서 론

종단간 원칙(end-to-end principle)[1]은 높은 확장성과 다양성을 보장함으로써 인터넷의 폭발적인 성장을 가져왔다. 네트워크의 기능 및 성능이 체감 품질에 영향을 주는 종단간 응용(P2P, 영상 등)들이 지속적으로 증가하고 있기 때문에 네트워킹의 유연성 확보가 요구된다. 하지만, 패킷 포워딩 중심의

◆ 주저자 : 국가슈퍼컴퓨팅연구소, 한국과학기술정보연구원, jiny92@kisti.re.kr, 정회원
° 교신저자 : Networked Computing Systems 연구실, 광주과학기술원, jongwon@nm.gist.ac.kr, 종신회원
* 국가슈퍼컴퓨팅연구소, 한국과학기술정보연구원, soyeon.lee@kisti.re.kr, 정회원
** 국가슈퍼컴퓨팅연구소, 한국과학기술정보연구원, kju@kisti.re.kr, 정회원
논문번호 : KICS2012-01-069, 접수일자 : 2013년 1월 31일, 최종논문접수일자 : 2013년 3월 22일

핵심망 설계는 네트워크의 유연성을 저해하고 있으며, 실제로 인터넷의 낮은 유연성은 IP 멀티캐스트 및 IPv6 등 신규 서비스들의 망 전개에 진입장벽으로 작용하는 실정이다.

최근 컴퓨팅 분야에서 성숙한 프로그래밍 방법론을 네트워크 분야로 확장하여 네트워크의 유연성을 제공할 수 있는 새로운 네트워킹 패러다임으로 SDN(Software Defined Networking)[2]이 대두되고 있다. SDN은 제어평면을 논리적으로 중앙 집중화하고 프로그램 가능화함으로써 네트워크 관리와 제어의 유연성을 담보한다. 데이터평면을 직접 제어하기 때문에 기관망 및 데이터센터망 등 자원제어가 자유로운 단일제공자(single-provider) 네트워크에서 효과적으로 적용될 수 있다. 하지만 인터넷 등 다중 제공자(multi-provider) 네트워크에서는 타 영역(domain)의 네트워크 자원에 대한 직접적 제어가 어렵기 때문에 종단간 연결성을 요구하는 응용들에게 유연한 네트워킹을 제공하는 데 한계가 있다.

본 논문은 서비스 오버레이 네트워킹(service overlay networking)[1]을 효율적으로 제어하기 위한 SDN 기반의 정책 제어기(policy controller)를 소개한다. 정책 제어기는 사용자요구에 기반을 두어 네트워크 정책(network policy)을 구성하고 구성된 정책을 SDN이 가능한 통신망 요소(network element)들에게 시행하는 PNS(programmable network substrate)의 제어 평면이다. 미들박스(middlebox) 형태의 서비스 게이트웨이(service gateway)[3]들은 PNS의 데이터 평면으로써 정책 제어기가 시행한 서비스 정책을 실시한다. 제안된 정책 제어기는 인터넷에 분산 배치된 서비스 게이트웨이들을 제어함으로써 교차영역(inter-domain) 네트워킹 환경 하에서 동적 오버레이를 구성할 수 있으며 유연성을 요구하는 종단 응용들에게 가상링크(virtual link)를 통해 종단간 연결성을 제공한다.

제안된 정책 제어기는 OpenFlow[4] 기반의 중앙 제어기인 NOX[5], Beacon[11], Floodlight[12] 등과 상이한 추상화 수준(abstraction level)을 갖는다. 서비스 추상화 계층을 추가적으로 제공함으로써 사용자 친화적(user-friendly)인 네트워크 서비스가 가능하다. 구체적으로, 제안된 정책 제어기는 패킷 플로우의 정합 및 처리규칙에 근거해 가상망을 제어하며 최선형(best-effort) 또는 부하 제어형(load-balancing) 네트워크 서비스를 제공할 수 있다[2].

제안된 정책 제어기가 갖는 기술적 기여점은 다음과 같다. 첫째, 교차영역 네트워킹 환경하에서 비용-효율적인 플로우 제어가 가능하기 때문에 네트워킹의 유연성을 높이고 응용에 최적화된 가상망을 구성할 수 있다. 둘째, NAT(Network Address Translator) 등 주소 변환을 필요로 하는 미들박스 기기들의 실증 환경을 인터넷 상에 구성할 수 있다. 마지막으로, 사용자 주문형(on-demand) 가상망 구성이 가능하며 링크와 노드 정보 등 가상망 구성을 위한 네트워크 가시성(visibility)을 제공함으로써 다양한 서비스 응용들을 시험 검증 할 수 있다.

본 논문의 2장에서는 SDN기반의 가상망 구성 환경을 총람하며, 3장에서는 정책 제어기의 각 구성 요소들의 세부 구현 내용을 살펴본다. 4장에서는 정책 제어기의 관리 응용과 서비스 응용을 예시하고, 5장에서 성능 평가를 수행한다. 마지막으로, 6장에서 결론을 맺는다.

Ⅱ. SDN기반의 가상망 구성 환경

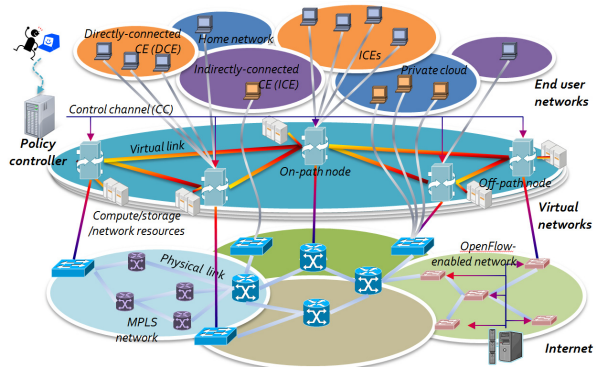


그림 1. 제안된 정책 제어기의 응용 환경
Fig. 1. Application environment of proposed policy controller

본 장에서는 정책 제어기가 적용될 SDN 환경에 대해서 OpenFlow 기반의 SDN 환경과 차별화되는 특징들을 중심으로 살펴본다. 그림 1은 제안된 정책

1) 본 논문에서 서비스 오버레이 네트워킹(service overlay networking)은 "on top of IP" 환경하에서 제공되는 중첩망 서비스를 의미한다. 또한 가상망(virtual network) 또는 논리망(logical network)은 중첩망과 동일한 의미로 사용한다.

2) 본 논문에서 네트워크 서비스(network service)는 서비스제공을 위해 정의된 네트워킹 프로토콜을 뜻하며 서비스 응용(service application)은 프로토콜을 구동하는 소프트웨어를 의미한다.

제어기가 적용될 SDN 기반의 가상망 구성 환경을 보여준다

가상망 구성환경은 구조적으로 OpenFlow기반 SDN 환경과 유사하며 정책제어기, 제어채널(control channel), 서비스 게이트웨이(이하, 노드)를 포함한다. 노드들은 가상링크를 통해 상호 연결되며 정책 제어기는 제어 채널을 이용해 노드들을 제어한다. 노드들은 IP 패킷의 프로세싱 및 포워딩을 담당하며 미들박스 형식의 네트워킹 모델(i.e. on-path 또는 off-path 노드)을 갖는다.

정책 제어기는 미들박스 형식의 노드들을 가상링크를 통해 상호 연결시킴으로써 교차영역 네트워킹을 가능하게 한다. 제안된 가상망 구성 환경은 응용계층 오버레이네트워크와 유사한 특징을 갖기 때문에 기저망(underlay network)의 구조 변경을 요구하지 않으며 기저망에서 제공하는 네트워킹 기능(e.g. 라우팅 등)을 활용할 수 있다. 결론적으로, 제안된 정책 제어기는 비용-효율적인 플로우 제어를 통해 종단 응용들에게 유연한 인터넷 환경을 제공하고자 한다.

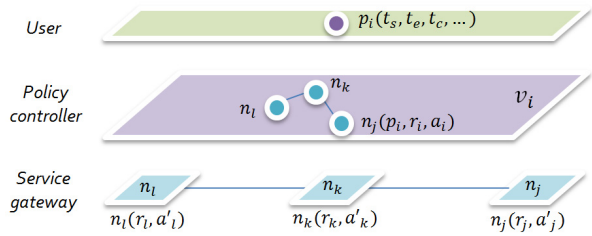


그림 2. 서비스 요청, 구성, 실행 과정의 개념도
Fig. 2. Conceptual view of service request, composition, and enforcement

정책 제어기가 적용될 가상망 구성 환경은 부분적으로 감결합된(partially decoupled) 네트워킹 정책을 적용받기 때문에 중앙 집중화된 제어기의 처리 부하를 노드들에게 분산시킬 수 있다. 즉, 제어평면의 기능이 정책 제어기와 노드에 분산되어 있다. 그림 2는 가상망 구성 환경에서 네트워크 서비스의 구성 및 실행(enforcement)를 보여주는 개념도이다. 정책 제어기는 사용자가 요청한 네트워크 서비스(p_i)를 분석하고 이에 대응하는 가상망 서비스을 구성한다. 또한, 가상망(v_i)에 포함된 노드들(n_j, n_k, n_l)에 대해 플로우 정합규칙(r)과 처리규칙(a)을 정의한다. 노드들은 제어기로부터 할당받은 처리규칙을 자신의 네트워킹 상태(유휴 자원 정보 등)에 적응시킨 후 정합되는 패킷에 대해 적응된 처리규칙(a')을 실행한다.

2.1. 네트워크 서비스 모델

그림 3은 정책 제어기가 제공하는 네트워크 서비스 모델을 보여준다. 정책 제어기는 사용자에게 제공될 네트워크 서비스를 결정한다. 즉, 사용자 요구를 충족시킬 수 있는 서비스 응용을 선택하고 노드들에게 가용 자원을 할당한다. 다양한 서비스 응용들(예를 들어, 라우팅 프로토콜 A, B, C 등)을 탑재할 수 있기 때문에 정책 제어기는 정책 결정을 통해 적합한 서비스 응용을 선택해야 한다. 개별 서비스 응용들에 대한 구체적인 기술은 본 논문의 영역을 벗어난다. 때문에 본 논문에서는 정책 제어기를 통해 제공될 수 있는 서비스 모델에 한정해 기술한다.

제안된 정책 제어기는 종단 사용자들에게 콘텐츠 전달, 응용 최적화, 서비스 가능화(enabling) 등의 네트워킹 기능을 제공할 수 있다. 콘텐츠 전달(content delivery) 모델은 노드 또는 노드와 연결된 컴퓨팅 기기(CE: Computing Element)의 메모리나 스토리지 자원을 활용하는 서비스이다. WAN 가속기[13]나 HTTP 프록시[14] 등이 콘텐츠 전달 모델에 속한다. 자원 최적화(resource optimization) 모델은 경로 최적화, 근접 네트워킹(proximity networking) 등 네트워크 자원의 효율적 사용을 지원하는 서비스이다. 서비스 연결(service inter-connection) 모델은 홈투홈(home-to-home)[15] 네트워킹, 멀티캐스트 브리징(bridging)[16] 등 서비스 도메인 간 연결 제어를 위한 서비스이다.

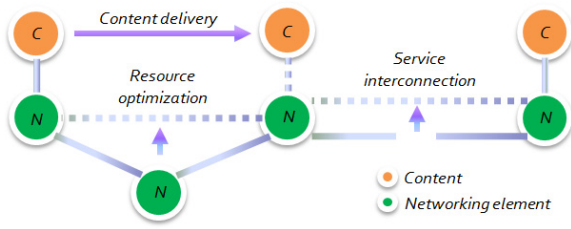


그림 3. 네트워크 서비스 모델
Fig. 3. Network service models

2.2. 네트워크 추상화

제안된 정책 제어기는 네트워크 서비스 추상화(network service abstraction) 및 네트워크 뷰 추상화(network view abstraction) 계층을 제공한다. 네트워크 서비스 추상화 계층이 서비스 UNI(Service User-Network Interface)를 제공하기 때문에 사용자

와 정책 제어기는 서비스 UNI를 이용해 상호 연동된다. 제어기에 탑재된 서비스 응용들은 뷰 추상화 계층에서 제공되는 네트워크 가시성 정보를 이용해 네트워크 서비스를 구성한다. 노드들은 네트워크 플로우 및 제어 추상화(network flow and control abstraction) 계층을 제공하며 정책 제어기에 의해 구성된 네트워크 서비스를 실현한다.

네트워크 플로우 및 제어 추상화 계층은 플로우 엔트리(flow entry)에 의해 구체화된다. 플로우 엔트리는 패킷 플로우의 정합규칙과 처리규칙을 제시한다. 정합규칙은 TCP/IP 5-tuple을 이용해 정의되며 개별 플로우 또는 플로우 집합(flow aggregation)을 명시할 수 있다. 처리규칙은 패킷 헤더 추가 및 삭제, 헤더 정보의 변경 등 저수준 명령과 DPI(Deep Packet Inspection)[3] 등 고수준 명령을 포함할 수 있다.

네트워크 뷰 추상화 계층은 정책 제어기 내부의 서비스 응용들에게 네트워크 가시성 정보를 제공한다. 네트워크 가시성은 노드 및 노드의 에지맵(edge map) 상태, 노드 간 가상링크의 상태 등을 포함한다. 정책 제어기는 플로우 설정요청 패킷이나 HeartBeat 패킷의 수신을 통해 노드의 상태 정보를 획득한다. 또한, 가상링크의 상태를 측정하기 위해 노드 간 액티브 모니터링(active monitoring)을 지시한다. 서비스 응용은 제어기가 수집한 가시성 정보를 바탕으로 네트워크 서비스를 구성한다.

네트워크 서비스 추상화 계층은 서비스 UNI에 의해 구체화된다. 서비스 UNI는 사유 프로토콜(proprietary protocol)인 SDP(Service Description Protocol)를 정의한다. 사용자는 SDP를 이용해 네트워크 서비스의 종류, 서비스가 적용될 플로우 영역(Flowspace), 에지링크(edge link) 및 가상링크의 종류 등 정책 제어기가 서비스 구성을 위해 필요한 네트워킹 정보를 제공한다.

2.3. 가상링크 및 에지링크

정책 제어기가 제공하는 처리규칙은 패킷 포워딩 모델과 패킷 프로세싱 모델로 구분할 수 있다. 포워딩 모델은 입력된 IP 패킷의 MAC 포트 또는 가상 출력 포트(virtual output port)를 결정해 포워딩하며 프로세싱 모델은 IP 패킷의 헤더 정보를 추가, 삭제, 또는 변경한다. 가상망에 속한 임의의 한 노드는 타 노드가 제공하는 가상포트(e.g. IP 주소, 노드 식별자 등)를 통해 해당 노드에 접속할 수 있다.

가상링크는 노드들 간의 점대점(point-to-point) 또는 점대다(point-to-multipoint) 연결성을 제공한다. 정책 제어기는 노드 간 가상링크의 구성을 위해 IP 터널링 또는 네트워크 주소변환 방식을 사용한다. IP 터널링은 패킷의 캡슐화, 주소변환, 역캡슐화 과정을 통해 노드 간 연결성을 제공하는 방식이며 네트워크 주소변환 방식은 캡슐화 및 역캡슐화 과정을 거치지 않고 주소변환만을 이용해 연결성을 제공한다(그림 4 참조). 주소변환은 전송 및 응용계층 프로토콜의 내부 주소 변경을 포함한다. IP 터널링 방식과 네트워크 주소변환 방식은 각각 종단 간 네트워킹 정보의 유지가 가능하거나 IP 단편화(fragmentation)의 문제가 발생하지 않는다는 장점이 있다.

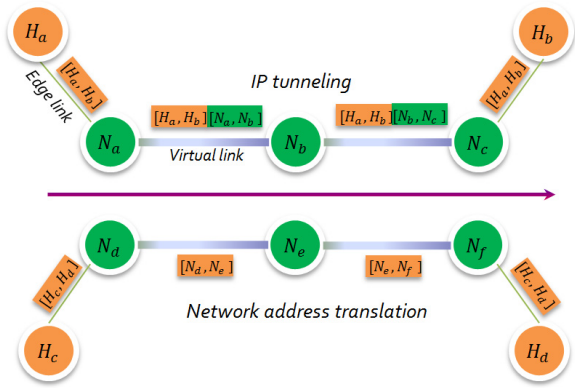


그림 4. IP 터널링과 네트워크 주소변환
Fig. 4. IP tunneling and network address translation

에지링크는 종단 CE와 노드 간에 물리링크 또는 가상링크를 의미한다. 노드는 CE와 물리적으로 직접 연결(directly-connected)되거나 논리적으로 간접 연결(indirectly-connected)된다. CE의 응용 소프트웨어나 커널 시스템의 변경을 요구하지 않기 때문에 종단 사용자는 투명한(transparent) 네트워킹이 가능하다. 서비스 확장성(extendability)을 높이기 위한 선택 사항으로 CE는 IP 터널링을 이용해 노드에 연결될 수 있다.

Ⅲ. 제어기 설계 및 구현

제안된 정책 제어기의 설계 목표는 인터넷 상에 주문형 가상망을 신속히 구성하는데 있다. 구성된 가상망은 종단 응용에 최적화된 네트워킹 환경을

3) SSDP(Simple Service Discovery Protocol) 등 응용 계층 프로토콜의 패킷 정보 필드에 제한적으로 DPI를 수행한다.

제공할 수 있다. 정책 제어기는 다음과 같은 설계 기준을 따른다. 첫째, 제어기가 제공하는 네트워킹 서비스에 대해서 사용자 접근성(accessibility) 및 활용성(usability)을 높인다. 둘째, 가상망을 사용자 요청별로 구성·유지함으로써 개별 가상망에 대한 서비스 관리성(manageability)을 높인다. 셋째, 가상망의 중앙 제어와 동적 관리를 통해 네트워킹의 유연성(flexibility)과 복원성(resiliency)을 높인다. 마지막으로, 개별 가상망에 적용되는 네트워킹 서비스를 차별화함으로써 서비스 다양성(diversity)을 추구한다.

3.1. 프로토타입

개별 구성 요소에 대한 설계는 OpenFlow 프로토콜, NOX 제어기, Ripcord 플랫폼[6] 등 기존 SDN 기술[17,18]들이 참조되었다. Ripcord는 데이터센터 네트워크를 제어하기 위한 NOX 기반의 네트워킹 플랫폼이다. 제안된 정책 제어기는 full-mesh 형태의 네트워크 토폴로지를 관리하고 가상링크와 가상포트에 기초해 네트워크 서비스를 구성한다는 점에서 타 기술과의 차별성을 갖는다.

제안된 정책 제어기가 기존 SDN 기술들과 차별화된 추상화 계층을 제공하기 때문에, 제어기 내부의 구성 요소 및 API(Application Programming Interface) 등이 재설계되어야 한다. 차별화된 추상화 계층은 2장에서 설명한 네트워킹 환경에 대한 추상화(i.e. 네트워크 뷰 추상화), 네트워크 플로우 및 제어의 추상화, 네트워크 서비스에 대한 추상화 등이다.

표 1. 네트워크 이벤트 및 제어 명령
Table 1. Network events and control commands

Events or control commands	Descriptions
NODE_JOIN, NODE_LEAVE	Node-join or leave events
STATS_REQUEST, STATS_REPLY	Request for flow and port statistics, and its reply
PACKET_OUT	Output command of a controller-assigned packet
SET_REQUEST, SET_FLOW	Flow setup request, Flow setup command
MON_REQUEST, MON_REPLY	Node-to-node probing request, and its reply
NODE_ALIVE	HeartBeat event

정책 제어기 및 노드는 네트워크 가시성 정보의 획득과 가상망의 제어를 위해 표 1과 같은 네트워크 이벤트 및 제어 명령을 사용한다. 이벤트는 노드가 제어기에게 보내는 메시지를 나타내며 명령은 제어기가 노드에게 보내는 메시지를 의미한다. OpenFlow 프로토콜의 이벤트 및 제어 명령과 유사하지만 이벤트나 제어 명령의 처리를 위한 내부 메커니즘은 가상망 환경에 맞춰져(customize) 있다. 메시징 프로토콜의 대표적인 특징은 노드들 간에 제어 메시지를 전달(노드-노드 메시징)하고 처리한다는 점이다. 예를 들어, 정책 제어기가 라우팅 경로상의 시작 노드에게 SET_FLOW, MON_REQUEST를 명령하면 해당 명령은 소스라우팅(source routing)을 통해 일련의 노드들에게 전달될 수 있다[4].

사용자나 관리자는 서비스 UNI 또는 제어기 CLI(Command Line Interface)를 통해 서비스 구성을 요청할 수 있다. 서비스 UNI는 서비스 요구자가 정책 제어기에게 가상망의 구성과 네트워크 서비스의 제공을 요청하는데 사용된다. 관리자는 제어기 CLI를 통해 특정 플로우에 대한 네트워크 서비스를 설정할 수 있다. SDN 환경에 응용 사용자의 서비스 요청을 수용함으로써 사용자 접근성과 가상망 환경의 활용성이 높아진다.

표 2. SDP 메시징을 위한 문자 집합
Table 2. Character set for SDP messaging

Service description	Descriptions
s=	User-defined name of a service
e=	IP addresses of CEs
p=	Join type to a network (dynamic, static)
c=	Communication types (point-to-multipoint communications)
t=	Types of edge links and virtual links
f=	Match rule of a flow

서비스 UNI는 사유 프로토콜인 SDP를 이용한다. 메시지 크기를 줄이고 신속한 프로토타이핑을 위해서 가독성이 높은 문자집합을 사용했으나, 호환성이나 확장성을 높이기 위해서 XML(eXtensible Markup Language) 등을 적용할 수 있다. 표 2는 SDP의 일부 형식을 보여준다. SDP를 이용해 추상

4) 정책 제어기의 선택 사항이다. 소스 라우팅을 위한 경로는 정책 제어기가 결정한다.

화된 서비스 요청은 정책 제어기에 의해 구체화된다. 즉, 정책 제어기는 사용자 요청을 기초로 서비스 제공에 필요한 구체적인 서비스 정책(가상망 토폴로지, 네트워크 서비스 등)을 결정한다.

3.2. 구성 요소

정책 제어기의 내부 구성은 관리 응용과 서비스 응용으로 구분된다. 그림 5는 정책 제어기의 내부 구조를 보여준다. 관리 응용의 각 구성요소가 갖는 기능은 다음과 같다.

요구 검증기(Verifier): 사용자 요청에 대한 수락 제어(admission control)를 수행한다. 수락 제어는 SDP 프로토콜의 준수 여부 및 서비스 제공을 위한 유휴 자원의 가용 여부 등을 기준으로 수행된다.

정책 관리기(Policy Manager): SDP 패킷 정보를 이용해 서비스 정책을 결정하고 결정된 정책을 PIB(Policy Information Base)에 저장한다. 또한, 노드가 신청한 플로우 설정요청에 대해 PIB에 저장된 서비스 정책을 적용하거나 개별 플로우에 대한 접근 제어(access control)를 수행한다.

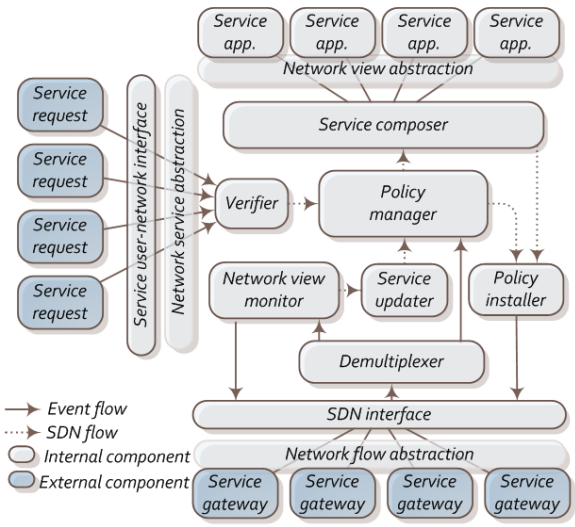


그림 5. 제안된 정책 제어기의 구조
Fig. 5. Architecture of proposed policy controller

서비스 구성기(Service composer): PIB에 저장된 서비스 정책을 읽어 서비스 응용을 구동한다. 개별 가상망은 논리적으로 독립된 네트워크 서비스와 토폴로지를 갖는다. 네트워크 토폴로지가 개별적으로 관리됨으로써 서비스 확장성을 높일 수 있다.

서비스 갱신기(Service updater): 노드 및 링크 오류가 발생할 경우, 개별 가상망의 네트워크 토폴로지를 갱신한다. 가상망 관리를 동적으로 수행함으로써 네트워킹의 복원성을 높인다.

네트워크 뷰 감시기(Network view monitor): 노드와 가상링크의 상태를 모니터링한다. 노드들은 가상링크를 통해 full-mesh 형태로 연결된다. 제어기와 노드 간 수직적 또는 노드와 노드 간 수평적 모니터링을 수행한다. Active 모니터링과 Passive 모니터링이 가능하다. 서비스 응용은 네트워크 뷰 감시기가 제공하는 네트워크 가시성 정보를 이용해 네트워크 서비스를 구성한다.

패킷 분류기(Demultiplexer): 노드는 플로우 설정요청 패킷, 노드/가상링크의 상태정보 패킷 등을 정책 제어기에게 송신한다. 패킷 분류기는 해당 패킷들을 해석하고 내부 구성 요소에게 전달한다.

정책 장치기(Policy installer): 네트워크 플로우 및 제어 추상화 계층에 호환되는 제어 메시지를 생성한다. 해당 제어 메시지를 수신한 노드는 플로우 테이블을 갱신하고 명령을 실행한다.

3.3. 가상망의 구성

정책 제어기는 서비스 정책이 등록된 플로우에 대해서 네트워크 서비스를 제공한다. 등록되지 않은 플로우에 대해서는 접근 제어를 수행한다. 관리자는 CLI를 통해 ACL(Access Control List)을 적용한다. 사용자는 SDP 메시지를 통해 네트워크 서비스를 요청하며 정책 제어기는 해당 요청에 대해 서비스 정책을 수립하고 결과를 PIB에 등록한다. 정책 제어기가 서비스 요구자의 주문형 서비스 요청을 수용함으로써 SDN 환경에 대한 사용자 접근성을 높이고 네트워크 서비스를 종단 응용에 특화시킬 수 있다.

수신된 패킷에 대한 플로우 정보가 없을 때, 노드는 정책 제어기에게 플로우 설정을 요청한다. 정책 제어기는 서비스를 구성하고 제어 메시지를 발생시켜 플로우 설정을 명령한다. 정책 제어기는 응용의 특성에 따라 반응형(reactive) 또는 순향형(proactive) 서비스 구성을 선택적으로 수행한다. 예를 들어, IP 멀티캐스트처럼 사용자들의 동적 참가가 예상되는 응용에 대해서는 반응형 서비스 구성이 수행된다. 즉, 노드의 명시적 플로우 설정요청에 의해 반응형 서비스 구성이 이루어진다.

순향형 서비스 구성은 노드의 설정요청 여부와 관계없이 서비스 구성이 수행된다. 순향형 서비스 구성이 요구되면, 정책 관리기는 사용자 요청에 대한 서비스 정책(플로우 정보, 가상망 토폴로지, 라우팅 프로토콜, 가상링크 형태, 서비스 응용 등)을 결

정한다. 선택된 서비스 응용들은 가상링크 및 노드들의 상태 정보 등 네트워크 뷰 감시기에 의해 유지되는 전체 공유 자원 정보(global network view)를 이용해 네트워크 서비스를 구성한다. 개별 가상망의 네트워크 가시성 정보(토폴로지, 라우팅 경로 등)는 독립적으로 관리된다. 독립적 가상망 관리는 신속한 망 구축 및 제거를 가능하게 함으로 서비스 관리성을 높인다. 네트워크 서비스를 실현하는 개별 플로우 엔트리는 정책 장치기를 통해 관련된 노드에 설치된다.

반응형 서비스 구성의 경우, 정책 관리기는 사용자 요청에 대해 서비스 정책을 결정하고 결정된 정책을 PIB에 등록한다. 플로우 설정요청 메시지를 수신하는 시점에 네트워크 서비스를 구성한다. 노드가 플로우 설정요청 메시지를 보내면, 정책 관리기는 해당 플로우에 대한 접근 제어를 우선 수행한다. 접근이 허락된 플로우에 대해서 정책 관리기는 서비스 정책의 PIB 등록 여부를 조사한다. 또한, 서비스 구성기는 등록된 정책을 이용해 서비스를 구성한다. 이후 과정은 순향형 서비스 구성과 동일하다.

네트워크 뷰 감시기는 노드의 유휴 자원, 가상링크의 상태 등 전체 공유 자원 정보를 모니터링한다. 이를 통해, 노드 및 링크 실패로 인한 연결성 단절 문제를 동적으로 해결한다. 노드의 상태 정보(네트워크 참가 및 탈퇴, 에지맵, 유휴 자원, 플로우 상태 등)는 Active 모니터링과 Passive 모니터링을 통해 얻는다. 네트워크 뷰 감시기는 가상링크의 상태 정보(연결성, 홉 카운트 등)를 얻기 위해 노드들 간 패킷 Probing을 제어함으로써 노드간 액티브 모니터링을 지원한다.

Ⅳ. 제어기의 관리 응용 및 서비스 응용의 구현

정책 제어기의 초기 프로토타입은 처리 속도를 고려해 C 코드를 이용해 구현했으며, 총 7,136 LOC(Lines of Code)를 갖는다(그림 5의 내부 구성요소 중 서비스 응용을 포함). 내부 구성요소는 네트워크 가시성 정보를 확보하기 위한 관리 응용(management application)과 가시성 정보를 활용해 서비스를 제공하는 서비스 응용(service application)으로 구분된다. 각 응용은 관리 및 서비스 특성에 따라 다양한 기능을 가질 수 있다.

4.1. 관리 응용

네트워크 뷰 모니터링

네트워크 가시성 정보는 NODE_JOIN, NODE_LEAVE, NODE_ALIVE, STATS_REPLY, MON_REPLY 이벤트를 통해 갱신된다. 노드가 NODE_JOIN을 통해 네트워크에 참가할 때는 정적으로 설정된 자신의 지리 정보 및 자신이 속한 AS(Autonomous System) 정보 등을 제어기에게 통보한다. NODE_JOIN을 통해 전달된 정보는 트래픽분리(traffic isolation)나 근접라우팅(proximity routing) 등의 서비스 응용에 활용될 수 있다.

CE와 물리적 연결을 갖지 않는 노드들은 활성 플로우(Active Flow)의 부재로 인해 SET_REQUEST 이벤트 발생이 제한적일 수 있다. 이는 가시성 정보와 물리 네트워크 간의 네트워크 일관성(Consistency)을 저해한다. 네트워크 가시성 정보를 최신 상태로 유지하기 위해서 각 노드들은 주기적으로 NODE_ALIVE 이벤트를 발생시킨다. NODE_ALIVE 메시지는 노드의 트래픽 통계와 테이블 통계(플로우 테이블, ARP(Address Resolution Protocol) 테이블, 가상포트 테이블 등)를 포함한다. 일정 시간동안 NODE_ALIVE 메시지를 받지 못할 경우, 묵시적 NODE_LEAVE 상태인 것으로 판단해 해당 노드와 관련된 가시성 정보를 삭제한다. 표 3은 NODE_ALIVE 메시지에 포함되는 통계 정보의 일부이다.

표 3. 가용 자원 및 트래픽 통계 정보의 획득
Fig. 3. Available resources and traffic statistics

Fields	Descriptions
FLOW_AVAILABLE, FLOW_CAPACITY	Available space in flow table and capacity of flow table
ARP_AVAILABLE, ARP_CAPACITY	Available space in ARP table and capacity of ARP table
VP_AVAILABLE, VP_CAPACITY	Available space in VP table and capacity of VP table
PORT_BYTES, PORT_PACKETS, PORT_DROPPED	Port statistics in bytes, in packets, and in dropped packets

STATS_REPLY 메시지는 개별 노드의 트래픽, 테이블, 플로우 통계 등 실시간 통계 정보를 확보하기 위해 사용된다. 노드는 정책 제어기가 요구한 STATS_REQUEST 메시지에 대한 응답으로 STATS_REPLY를 피드백(feedback)한다.

네트워크 뷰 감시기는 다양한 모니터링 정책을 수용할 수 있는 관리 응용으로써, MON_REQUEST 메

시지를 주기적으로 발생시켜 노드 간 가상링크의 상태를 모니터링한다. 감시기는 m개의 가용 노드 중 n $(2 \leq n \leq m)$개의 노드들을 선택한 후, 소스 라우팅이 가능하도록 MON_REQUEST 메시지를 구성한다. MON_REQUEST 메시지를 수신한 노드는 n개의 노드들이 순차적으로 MON_REQUEST 메시지를 수신할 수 있도록 소스 라우팅을 수행한다. 노드 간 도달 가능성(reachability) 및 홉 카운트와 같은 가상링크 정보가 MON_REQUEST 메시지를 통해 획득된다. 라우팅 경로 상의 각 노드 또는 최종 노드는 수집된 링크 정보를 정책 제어기에게 전달하기 위해서 MON_REPLY 메시지를 이용한다. 정책 제어기는 수집된 노드 및 링크 정보를 NIB(Network-view Information Base)에 저장해 관리한다.

가상망 토폴로지 관리

정책 제어기는 논리적으로 구분된 다수의 가상망들을 동적으로 관리한다. 서비스 갱신기는 구성된 가상망의 개별 네트워크 토폴로지를 관리함으로써 물리 네트워크(가상망 구성 환경)와 가상망 간에 일관(consistent)된 가시성 정보를 유지한다. 개별 가상망에 대한 정보는 VIB(Virtual-network Information Base)에 저장된다.

노드들이 가상링크를 통해 상호 연결되기 때문에, 관리 응용이 제공하는 원(raw) 네트워크 토폴로지는 full-mesh 형태로 간주된다. 서비스 응용은 원 토폴로지를 이용해 가상망 토폴로지(e.g. 트리(tree), 해쉬(hash) 기반 등등)를 구성하고 VIB에 저장한다. 원 네트워크 토폴로지에 상태 변화가 있을 경우, 서비스 갱신기는 기 구성된 개별 가상망의 토폴로지 정보를 수정한다. 변경된 서비스 정보는 정책 관리기 및 서비스 구성기를 통해 관련 노드들에게 반영된다. NODE_LEAVE와 같은 명시적 네트워크 탈퇴 요청이나 NODE_ALIVE의 타임아웃(timeout) 등 묵시적 네트워크 탈퇴에 의해 원 네트워크 토폴로지가 변경될 수 있다. 각 노드는 전용 CE이기 때문에 사용자 역동성(dynamicity) 또는 Churn으로 인한 네트워크 불안정성(instability)이 낮은 장점이 있다.

3.2. 서비스 응용

서비스 응용은 관리 응용이 제공하는 네트워크 가시성 정보를 바탕으로 네트워크 서비스를 구성한다. 정책 관리기는 사용자 요청을 수행할 서비스 응용을 선택한다. 본 절에서는 가상링크의 종류 및 서비스의 구성 방법 등에 차이가 있는 두 서비스 응용에 대해 구현 사례를 제시한다[5].

응용계층 멀티캐스트 서비스의 순향형 구성

본 응용은 노드(N)에 간접 연결된 CE(H)들에게 일대다 또는 다대다 통신 서비스를 제공하는데 목적이 있다. 간접 연결된 CE들을 가상망 구성 환경에 수용함으로써 서비스 접근성과 활용성을 높일 수 있다. 순향형 서비스 구성은 플로우 설정요청 및 응답으로 인해 발생되는 플로우 설정 시간(flow setup time)을 줄이므로, 에지링크의 링크 활용률(link utilization)을 높일 수 있다.

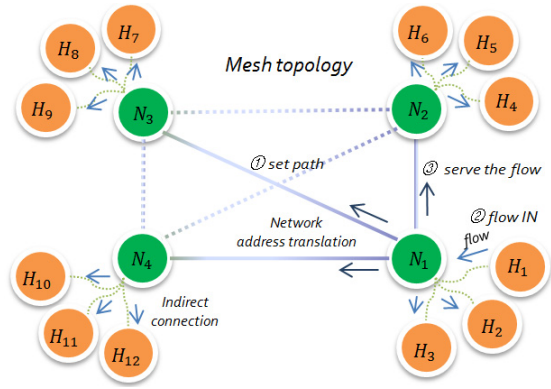


그림 6. 순향형 서비스 구성: 응용계층 멀티캐스트
Fig. 6. Proactive service composition: application-level multicast

사용자는 정책 제어기에게 CE들의 IP 주소 목록을 전달한다. 정책 제어기는 서비스 구성을 완료한 후 각 CE들이 접속할 노드 정보를 반환한다. 정보 교환을 위한 SDP 메시지를 이용한다. 사용자들은 Virtual Venue[7] 등 가상 공간을 이용해 CE들의 주소 목록을 교환할 수 있다.

그림 6은 순향형 서비스 구성을 통한 응용계층 멀티캐스트의 적용을 예시한다. 정책 제어기는 사용자가 요구한 네트워크 서비스를 선 시행하고 구성된 플로우 엔트리를 각 노드들에게 전달함으로써 노드들 간 가상링크를 설정한다(① set path). 한 CE가 연관된 플로우를 전송하면(② flow IN) 기 설정된 플로우 엔트리의 정보에 따라 해당 플로우를 처리하고 이웃 노드로 포워딩한다(③ serve the flow).

5) 정책 제어기의 구현 사례를 제시하는데 목적이 있다. 서비스 응용의 성능 최적화나 신규 서비스 응용의 제시 등은 본 논문의 영역에서 벗어난다.

가상망에 참가하는 노드들의 총 수가 n이고 노드 i의 차수(degree)가 d_i일 때, 노드 i는 이웃하는 노드들과 full-mesh 형태로 연결되기 때문에 $n-1$개의 가상링크를 갖는다 ($d_i < n$). 각 노드의 차수는 동일하며 가상링크가 연결되지 않은 초기 상태인 것으로 가정한다. 따라서 노드 i에 간접 연결될 수 있는 CE의 수는 $d_i-(n-1)$로 제한된다. 서비스에 참가할 CE의 총 수가 h이면, 정책 제어기는 다음을 만족시키는 α의 최소값을 선택한다. 즉, 정책 제어기는 m개의 가용 노드들 중 $n=\alpha$개의 노드들을 선택해 full-mesh 토폴로지를 구성한다.

$$\sum_{i=1}^{\alpha}(d_i-n+1) > h$$

정책 제어기는 SDP 메시지의 'e='과 'f='필드 값을 이용해 정합규칙을 구성하며 가상포트로 패킷을 출력하기 위한 플로우 처리규칙을 정의한다. 예를 들어, 송신자 n_s가 수신자 n_d로 전송하는 플로우 f에 대해서, 정책 제어기는 정합규칙 $m=[n_s, n_d, p_n, p_s, p_d]$을 구성한다. p_n은 IP 프로토콜 번호, p_s는 송신지 포트 번호, p_d는 목적지 포트 번호이다. 또한, 해당 플로우에 대한 처리규칙 $a=O_n(t[f])$를 정의한다. $t[f]$는 플로우 f에 속한 패킷 헤더의 필드값 변경을 의미하며 $O_n(\alpha)$은 변경된 패킷 플로우 α를 가상포트 n으로 출력하는 처리규칙이다. 플로우 f는 m에 의해 추상화 된다. 예를 들어, $O_{H_2}(t[f_{m_1}])$은 정합규칙 m_1에 해당되는 패킷 플로우가 수신되면 패킷 헤더의 필드값을 변경해 CE H_2로 전송하는 것을 의미한다.

표 4. 응용계층 멀티캐스트를 위한 정합규칙과 처리규칙들
Table 4. Match rules and actions set for application-level multicast

Node	Match rule	Actions
N_1	m_1 = $[H_1, N_1, p_n, *, p_d]$	$O_{H_2}(t[f_{m_1}]), ..., O_{N_4}(t[f_{m_1}])$
N_3	m_3 = $[N_1, N_3, p_n, *, p_d]$	$O_{H_7}(t[f_{m_3}]), ..., O_{H_9}(t[f_{m_3}])$

표 4는 그림 6의 N_1과 N_3에 대한 정합규칙 및 처리규칙을 예시한다.

서비스 구성이 완료되면 정책 제어기는 라우팅 경로 상의 노드들에게 SET_FLOW 명령을 전달한다. 명령을 수신한 각 노드들은 플로우 정합규칙과 처리규칙을 플로우 엔트리에 저장하고 정합되는 플로우에 대해서 처리규칙을 적용한다.

하이브리드 멀티캐스트 서비스의 반응형 구성

본 응용은 노드에 직접 연결된 CE 또는 노드가 속한 네트워크 도메인 간에 IP 멀티캐스트 서비스를 제공하는데 목적이 있다. CE는 IGMP(Internet Group Management Protocol)를 이용해 멀티캐스트 그룹에 참가하며 그룹 주소를 이용해 IP 멀티캐스트 패킷을 송신한다. 멀티캐스트 패킷의 노드 간 전달을 위해 IP 터널링이 이용된다. 서비스 도메인 간을 연결해 통신이 가능하게 하는 서비스 가능화 모델이다. 사용자는 IP 멀티캐스트 패킷을 송신하는 CE의 IP 주소를 SDP 메시지를 이용해 등록한다.

사용자는 SDP 메시지를 통해 IP 멀티캐스트 서비스를 요청하며 정책 제어기는 서비스 정책을 결정하고 PIB에 등록한다. 등록된 CE가 IP 멀티캐스트 패킷을 송신하고, 하나 이상의 CE가 IGMP 참가 메시지를 발생시키면 정책 제어기는 가상망을 구성한다. 무등록 CE가 송신한 IP 멀티캐스트 패킷은 정책 제어기가 삭제를 명령하고 노드에 의해 삭제된다.

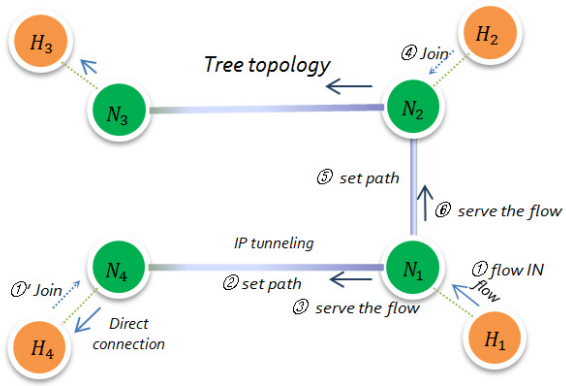


그림 7. 반응형 서비스 구성: 하이드리드 멀티캐스트
Fig. 7. Reactive service composition: hybrid multicast

그림 7은 반응형 서비스 구성을 통한 하이브리드 멀티캐스트의 적용을 예시한다. 패킷 플로우를 수신받은 노드(① flow IN)는 정책 제어기에게 플로우 엔트리의 설정을 요청한다. 정책 제어기는 멀티캐스트 그룹에 참가 요청한 CE가 존재할 경우(①' join), 플로우 엔트리를 구성해 노드들 간 가상링크를 설정하고(② set path) 해당 플로우를 관련 노드에게 포워딩한다(③ serve the flow). 이하, ④,⑤,⑥은 ①',②,③의 처리 과정과 일치한다.

노드는 CE가 송신한 IP 멀티캐스트 패킷의 일부를

캡슐화한 후, 정책 제어기에게 플로우 설정(SET_REQUEST 메시지 송신)을 요청한다. 개별 노드들이 IGMP 서비스를 제공하기 때문에, 정책 제어기의 간섭 없이 IGMP 메시지를 발생시킬 수 있다. 또한, 노드는 CE와의 멀티캐스트 연결 상태를 관리하기 위해 IGMP 테이블을 유지한다.

하나 이상의 송신자와 수신자가 가상망에 참가할 때, 네트워크 서비스가 구성된다(예를 들어, 그림 7의 CE H_4가 노드 N_4를 통해 그룹 참가를 요청하고 CE H_1이 N_1으로 멀티캐스트 패킷을 송신하는 시점). 서비스 응용으로 MST(Minimum Spanning Tree) 기반의 네트워크 토폴로지 관리기를 활용하며 노드들 사이의 가상링크를 구성하기 위해 IP 터널링이 이용된다.

표 5. 하이브리드 멀티캐스트를 위한 저합규칙과 처리규칙들
Table 5. Match rules and actions set for hybrid multicast

	Node	Match rule	Actions
②	N_1	m_1 = $[H_1, g, p_n, *, p_d]$	$O_{N_4}(c[f_{m_1}])$
	N_4	m_4 = $[N_1, N_4, p'_n, p'_s, p'$	$d[f_{m_4}]$
⑤	N_1	m_1 = $[H_1, g, p_n, *, p_d]$	$O_{N_2}(c[f_{m_1}])$, $O_{N_4}(c[f_{m_1}])$
	N_2	m_2 = $[N_1, N_2, p'_n, p'_s, p'$	$d[f_{m_2}]$

가상망 토폴로지는 그림 7의 H_2와 같이 CE가 신규 노드를 통해 멀티캐스트 그룹에 참가할 때 확장된다(i.e. H_2의 멀티캐스트 참가로 인한 N_2의 SET_REQUEST 이벤트 발생 시점). 그림 7의 표기 순서에 따른 정합규칙과 처리규칙은 표 5와 같다. 멀티캐스트 그룹 주소는 g이며, 가상링크 상의 프로토콜 번호는 p'_n이다. 정책 제어기는 개별 플로우 간의 충돌(Flowspace conflict)을 방지하기 위해 FAL(Flowspace Adaptation Layer) 헤더(e.g. $p'_s : p'_d$)를 설정한다. IP 터널을 구성하기 위해서, 플로우의 진입(ingress) 노드는 IP 패킷을 캡슐화($c[f_m]$)하며 출구(egress)노드는 캡슐화된 패킷을 역캡슐화($d[c[f_m]]$)한다. 예를 들어, $O_{N_4}(c[f_{m_1}])$는 정합규칙 m_1에 해당되는 패킷 플로우가 수신되면 해당 패킷들을 캡슐화를 하고 노드 N_4로 전송하는 것을 의미한다. 캡슐화를 위해 총 24바이트(i.e. 20바이트 IP 헤더와 4바이트 FAL 헤더)가 IP 패킷에 추가된다.

V. 평 가

본 절에서는 정책 제어기가 제공하는 두 서비스 응용을 이용해 정책 제어기의 서비스 처리 시간을 평가한다. 본 논문은 정책 제어기의 처리성능 평가에 초점을 둔다. SDN 환경에서 정책 제어기가 갖는 확장성(Scalability), 일관성(consistency) 등에 대한 평가는 추후 과제로 한다.

정책 제어기가 SET_REQUEST 이벤트를 수신하는(또는, 서비스 구성을 요청하는 SDP 패킷을 수신하는) 시점부터 서비스 구성을 마친 후 서비스에 참가하는 모든 노드들에게 SET_FLOW 명령을 송신 완료하는 시점까지의 경과 시간을 서비스 처리 시간으로 정의한다. 정책 제어기의 서비스 처리 시간은 노드의 플로우 설정 시간에 영향을 주기 때문에 최소화되어야 한다.

정책 제어기는 4GB RAM과 Intel Atom D525 1.8 GHz CPU를 갖는 Fedora Core 13 시스템 상에서 구동되었다. 네트워크 가시성 정보의 유지관리를 위해서 MySQL 5.1.56이 활용되었다.

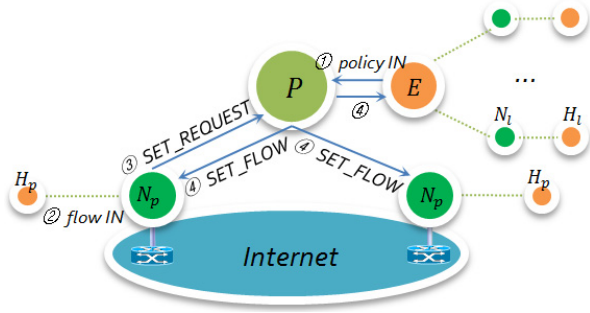


그림 8. 테스트베드; 순향성 서비스 구성의 적용을 예시
Fig. 8. Testbed; it illustrates a reactive service composition

실험 환경은 그림 8과 같다. 네트워크 플로우 및 제어 추상화를 위해서 2대의 물리적 노드(N_p)가 네트워크에 배치되었다. 각 노드는 정책 제어기(P)와 동일한 시스템 사양을 가지며 NetFPGA[8] 하드웨어를 이용해 IP 터널링 및 네트워크 주소변환을 수행한다. 물리적 노드들과 별개로 에뮬레이션 시스템(E)에 논리적 노드(N_l)들을 구현해 서비스 응용에 적용했다. 논리적 노드들이 사용하는 이벤트는 NODE_JOIN, NODE_ALIVE, SET_REQUEST로 제한된다. 마지막

으로, 각 노드들은 물리적 또는 논리적으로 연결된 CE(H_p 또는 H_l)를 가지며 각 기기(i.e. P, E, N_p)들은 1 Gbit/s 링크에 연결되었다.

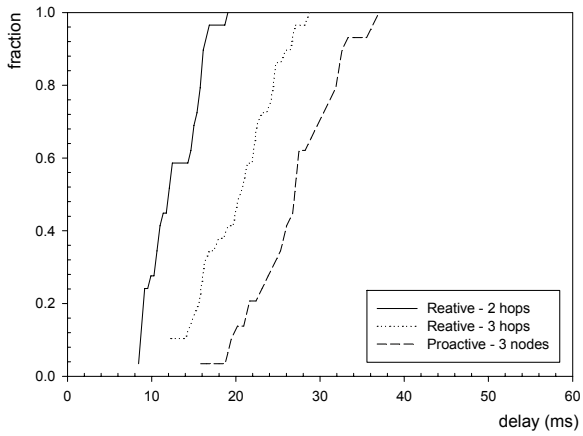


그림 9. 서비스 처리 시간의 누적분포함수
Fig. 9. CDF of service processing time

그림 9는 응용계층 멀티캐스트 서비스와 하이브리드 멀티캐스트 서비스에 대한 서비스 처리 시간의 누적분포함수(CDF)를 보여준다. 16개의 CE에 응용계층 멀티캐스트 서비스를 제공하기 위해서 3개의 노드가 사용되었다(그림 9의 Proactive - 3 nodes). 하이브리드 멀티캐스트 서비스의 경우, 3개의 CE를 위해 4개의 노드가 사용되었고 2개의 라우팅 경로를 구성했다. 한 개의 노드를 공유하기 때문에, 경로는 각각 2홉(Reactive - 2 hops)과 3홉(Reactive - 3 hops)을 갖는다. 오버레이 네트워크에서 홉 스트레치(hop stretch)의 증가는 패킷의 전달 지연을 유발[9]하기 때문에 적은 홉 수를 가정했다.

서비스 처리 시간은 네트워크 서비스를 구성하는데 사용된 노드의 수에 비례했다. 이는 데이터베이스의 처리 횟수 특히, 데이터베이스 쓰기 횟수에 영향을 받는 것으로 판단된다. 정책 제어기는 데이터베이스 시스템을 이용해 네트워크 가시성 정보를 유지하고 서비스 구성 결과 등을 저장하기 때문에 데이터베이스 처리(database transaction)를 필수적으로 수행해야 한다.

서비스 처리 시간은 정책 구성 시간과 서비스 시행 시간으로 구성된다. 표 6의 정책 구성 시간은 데이터베이스에서 기존 서비스 정책의 존재 여부를 파악하고 정책을 구성하는데 필요한 시간이다. 서비스 시행 시간은 선택된 서비스 응용의 구동, 결과의 데이터베이스 저장, 제어 메시지의 관련 노드 전송을 포함한다. 따라서 서비스 처리 시간 중 서비스 응용의 구동 결과를 데이터베이스에 기록하고 관련된 노드들에게 제어 메시지를 송신하는 후처리 과정이 제외된 시간이 정책 구성 시간이다. 즉, 정책 구성 시간은 데이터베이스 읽기(ACL 검사, PIB 검색 등)와 서비스 구성에 소요된 시간만 포함한다.

표 6. 평균 서비스 처리 시간 및 정책 구성 시간
Table 6. Average service processing time and policy composition time

ms	Reactive (2 hops)	Reactive (3 hops)	Proactive (3 nodes)
Service processing time	12.8	20.5	27.7
Policy composition time	4.7	5.4	-

2홉 라우팅과 3홉 라우팅을 수행하는 하이브리드 멀티캐스트 서비스의 서비스 처리 시간은 평균적으로 각각 12.8 ms와 20.5 ms로 측정됐다. 하지만, 정책 구성 시간은 4.7 ms와 5.4 ms로써, 후처리 과정에 각각 8.1 ms와 15.1 ms가 소요되었음을 확인할 수 있다[6]. 3개의 노드가 Full-mesh로 연결되는 응용계층 멀티캐스트 서비스의 서비스 처리 시간은 약 27.7 ms였다.

표 6의 3홉(Reactive - 3 hops) 라우팅과 3 노드 full-mesh(Proactive - 3 nodes) 구성을 위한 서비스는 동일한 노드 수를 갖는다. 하지만, 서비스 정책의 결정과 등록을 위한 추가적인 데이터베이스 처리로 인해 후자가 전자에 비해 약 7.2 ms의 지연이 증가했음을 알 수 있다.

반응형 서비스 구성 시, 정책 제어기의 서비스 처리 시간이 개별 노드의 플로우 설정 시간에 주는 영향이 크기 때문에 서비스 처리 시간은 최소화되어야 한다. 정책 제어기의 사용으로 인한 네트워킹의 유연성 증가를 고려할 때, 수십 ms의 서비스 처리 시간은 적절한 수준으로 판단된다. 참고로 퍼미션(permission) 검증, 경로 계산, 플로우 설정이 포함된 NOX 기반의 서비스 응용이 약 50 ms이내의 플로우 설정 지연을 보였다[10]. 후처리 과정 및 데이터베이스 최적화를 통해 서비스 처리 시간의 추가 단축이 가능할 것으로 판단된다. 노드에서의 패킷 처리 지연은 플로우에 속한 첫 번째 패킷에 대해서만 발생된다.

6) 순향형 서비스 구성은 PIB 등록을 위한 데이터베이스 쓰기 과정이 포함된다. 또한, 서비스 처리 시간이 플로우 설정 시간에 영향을 주지 않기 때문에 정책 구성 시간을 측정하지 않았다.

Ⅵ. 결 론

본 논문은 다중제공자 네트워크에서 패킷 플로우를 효과적으로 제어하기 위해 감결합된 네트워킹 정책을 논리적으로 중앙 집중화함으로써 네트워킹의 유연성을 확보하는 SDN 기반의 정책 제어기를 소개했다. 제안된 정책 제어기는 네트워크 가시성 정보를 이용해 가상링크 및 가상포트 등을 제어한다. 논리적으로 구분된 다수의 가상망을 신속히 구성·관리하기 때문에 종단 응용에 최적화된 네트워킹 환경을 제공할 수 있을 것으로 기대된다. 또한 제안된 정책 제어기가 적용될 SDN 환경은 미들박스 기기들의 기능을 통합하고 신규 인터넷 서비스를 시험 검증할 수 있는 비용-효율적인 네트워크 인프라로 활용될 수 있다.

References

[1] J. H. Saltzer, D. P. Reed, and D. Clark, “End-to-end arguments in system design,” *ACM Trans. on Comput. Syst.,* vol. 2, no. 4, pp. 277-288, Nov. 1984.

[2] N. McKeown, *Software-defined networking*, Retrieved Apr. 2009, from http://tiny-tera.standford.edu/~nickm/talks/infocom_brazil_2009_v1-1.pdf

[3] J. JO, S. Lee, K. Kong, and J. Kim, “A NetFPGA-based IP service gateway for the composition of service overlay networks,” *J. KIPS*, vol. 18, no. 6, pp. 413-422, Dec. 2011.

[4] N. McKeown, T. Aderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, “OpenFlow: enabling innovation in campus networks,” *ACM SIGCOMM comput. commun. review*, vol. 38, no. 2, pp. 69-74, Apr. 2008.

[5] N. Gude, T. Koponen, J. Pettit, B. Pfaff, M. Casado, N. McKeown, and S. Shenker, “NOX: towards an operating system for networks,” *ACM SIGCOMM comput. commun. review*, vol. 38, no. 3, pp. 105-110, Apr. 2008.

[6] B. Heller, D. Erickson, N. McKeown, R. Griffith, I. Ganichev, S. Whyte, K. Zarifis, D. Moon, S. Shenker, and S. Stuart, “Ripcord: a modular platform for data center networking,” *ACM SIGCOMM comput. commun. review*, vol. 40, no. 4, pp. 457-458, Oct. 2010.

[7] H. Ho, C. Yang, and C. Chang, “Building an e-learning platform by access grid and data grid technologies,” in *Proc. of EEE 2004*, pp. 452-455, Taipei, Taiwan, 2004

[8] NetFPGA, *The NetFPGA is:*, Retrieved Apr. 2013, from http://NetFPGA.org

[9] Y. Zhu, B. Li, and J. Guo, “Multicast with network coding in application-layer overlay networks,” *IEEE JSAC*, vol. 22, no. 1, pp. 107-120, 2004.

[10] Nicira Networks, Inc, “*Evolution of the Ethane architecture*,” Nicira Networks, Inc Technical Report AFRL-RI-RS-TR-2009-41, Feb. 2009.

[11] D. Erickson, *What is Beacon?*, Retrieved Apr. 2013, from http://www.beaconcontroller.net/

[12] Project Floodlight, *Open source software for building software-defined networks*, Retrieved Apr. 2013, from http://floodlight.openflowhub.org/

[13] S. Lhm, K. Park, and V. Pai, “Wide-area network acceleration for the developing world,” in *Proc. USENIX Annu. Tech. Conf.*, pp. 18, Boston, U.S.A., June 2010.

[14] A. Fox, S.D. Gribble, Y. Chawathe, and E.A. Brewer, “Adapting to network and client variation using infrastructural proxies: lessons and perspectives,” *IEEE Personal Commun.*, vol. 5, no. 4, pp. 10-19, 1998.

[15] T. Hwang, H. Park, and J. Chung, “Personal mobile a/v control point for home-to-home media streaming,” *IEEE Trans. Consum. Electron.*, vol. 54, no. 1, pp. 87-92, Feb. 2008.

[16] N. Kim and J. Kim, “UDP-tunneling based multicast connectivity solution for multi-party collaborative envrionments,” in *Proc. SPIE 6015*, pp. 275-284, Oct. 2005.

[17] N. Foster, A. Guha, M. Reiblatt, A. Story, M.J. Freedman, N.P. Katta, C. Monsanto, J. Reich, J. Rexford, C. Schelsinger, D. Walker, and R. Harrison, “Languages for software-defined networks,” *IEEE Commun. Mag.*, vol. 51, no. 2, pp. 128-134, Feb. 2013.

[18] M. Fernandez, "Evaluating OpenFlow controller paradigms," in *Proc. ICN 2013*, pp. 151-157, Seville, Spain, Jan. 2013.

[19] J. Oh, "Wireless Internet local broadcasting system using IP address translation," *J. KICS*, vol. 28, no. 3, pp. 217-223, Mar. 2003.

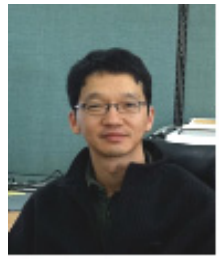
조 진 용 (Jinyong Jo)

1999년 2월 전남대학교 컴퓨터공학과 졸업
2002년 8월 광주과학기술원 정보통신공학과 석사
2003년 8월~현재 한국과학기술정보연구원
2007년 3월~현재 광주과학기술원 정보기전공학부 박사과정
<관심분야> 멀티미디어 시스템 및 서비스, 오버레이 네트워크, 소프트웨어-정의 네트워크

이 소 연 (Soyeon Lee)

2009년 2월 국민 대학교 전자공학과 졸업
2010년 3월~현재 한국과학기술정보연구원
<관심분야> 미래인터넷, 서비스 오버레이 네트워크

공 정 욱 (JongUk Kong)



1993년 2월 한국과학기술원 전기 및 전자공학과 졸업
1998년 2월 포항공과대학교 정보통신대학원 정보통신학과(석사)
2008년 충남대학교 정보통신공학과 박사수료
1993년~2001년 (주)데이콤 중앙연구소
2001년~2002년 (주)맥스웨이브
2002년~현재 한국과학기술정보연구원
<관심분야> 전자공학, 통신공학, 광통신 공학

김 종 원 (JongWon Kim)

1987년 2월 서울대학교 제어계측공학과 졸업
1989년 2월 서울대학교 제어계측공학과(석사)
1994년 2월 서울대학교 제어계측공학과(박사)
1994년~1999년 공주대학교 전자공학과 조교수
1998년~2001년 Univ. of Southern California, Dept. of Engineering - Systems 연구조교수
2001년~현재 광주과학기술원 정보통신공학부 교수
<관심분야> Networked Media Systems and Protocols focusing on "Dynamic Composition of Immersive Media-centric Services over the Wired/Wireless IP Convergence Networks"