SDN에 기반한 효율적인 네트워크 자원의 사용과 지연을 최소화하기 위한 서비스별 패킷 전송 알고리즘

손재혁○ 홍충선
경희대학교 컴퓨터공학과
{ sonjaehyeok, cshong } @khu.ac.kr

A Packet-Forwarding Algorithm Clasifed by Services for Eficient Network Resources and Minimizing Delay Based on SDN

JaeHyeok Son○ ChoongSeon Hong
Department of Computer Engineering, Kyung Hee University

요 약

미래 인터넷과 관련한 많은 연구들이 활발히 진행 중에 있는 지금, SDN(Software Defined Networking) 이라는 새로운 네트워킹 패러다임이 IT 사회의 큰 이슈로 대두되고 있는 추세이다. 따라서 본 연구에서는 SDN에서 패킷을 전송 할 때, 서비스 별로 패킷을 분류 및 전송 하도록 하여 네트워크 자원을 효율적으로 사용함과 더불어 패킷 전송 시에 발생 할 수 있는 지연을 최소화 하는 방안을 살펴보도록 할 것이다.

1. 서 론

현재 전 세계에 많은 사람들이 인터넷을 활용하여 다양한 활동 영역을 누리고 있다. 또한 IOT가 IT 산업에 새로운 이슈로 대두되어, 인간이 영위하는 모든 활동과 관련한 여러 분야들이 컴퓨터 네트워크와 밀접한 관련을 갖고 있다. 하지만 네트워크 사용자의 수가 증가하고 이에 따른 여러 문제점들이 생기고 있다. 예를 들어, 트래픽의 양이 기하급수적으로 증가하는 추세를 보이고 있으며, 트래픽들이 지나가는 네트워크 장비를 관리하는 데에 있어 많은 어려움을 겪는 것 또한 문제점이라 할 수 있다. 이러한 문제점들을 해결하기 위해 미래 인터넷에 관한 수많은 연구들이 활발히 진행 중에 있으며, 그 대표적인 예로 SDN(Software Defined Networking)을 들 수 있다. SDN의 기본적인 개념은 기존 Legacy 장비와는 다르게 Control Plane과 Data Plane이 따로 구분 되어 있어, 컨트롤러 소프트웨어에서 패킷 전송과 관련한 여러 결정 등을 하는 것이 가능하다는 것이다. 아래 그림 1[1]는 기존 네트워크 장비의 형태와 SDN 장비의 차이점을 보여준다.

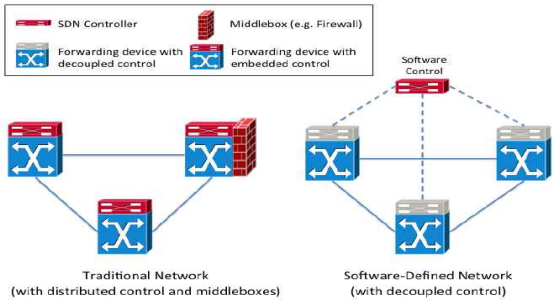

그림 1. 기존의 네트워크와 SDN의 비교

본 연구는 미래창조과학부 및 한국정보화진흥원의 미래네트워크연구시험망 구축운영 사업의 일환으로 수행하였음. [2014-Z349, 개방형 모바일 네트워크 플랫폼 국제확산 연구 및 실증시험] *Dr. CS Hong is the corresponding author

본 논문에서는 기존의 SDN이 동작하는 것과는 달리 서비스별로 우선순위를 정하고 서비스에 따라 패킷을 전송하여 네트워크 자원의 효율적인 사용과 더불어 패킷 전송 시 생기는 지연을 최소화 하는 방안을 제안한다.

2. 관련 연구

2.1. Procera

Procera는 프로그래밍 언어를 통해 여러 종류의 이벤트에 따라 컨트롤러에 네트워크 정책을 적용하고, 이러한 정책에 기반 하여 패킷 전송을 가능하게 하는 네트워크 컨트롤 프레임워크다[2].

2.2. 분산된 컨트롤러를 이용한 SDN 설계 및 네트워크 자원 관리

이는 효율적인 네트워크 자원의 관리를 위해 둘 이상의 컨트롤러를 이용하는 방법이다. 하나의 컨트롤러가 수용 가능한 범위 이상의 라우터(스위치)를 제어 할 때, 컨트롤러의 이상 등의 문제가 생길 수 있다. 이러한 문제점을 해결하기 위하여 고안된 방법으로, Load balancing 등을 통해 효율적인 네트워크 관리를 할 수 있는 장점이 있다. 또한 하나의 컨트롤러에 이상이 생겼을 경우, 다른 컨트롤러에 의해 패킷의 전송이 이루어지므로, network-down 시간을 0으로 유지 하는 것을 알 수 있다[3].

3. 제안 사항

3.1. 서비스별 패킷 전송 알고리즘

기존 SDN의 동작을 살펴보면, OVS 자체에 우선순위를 부여하여, 컨트롤러가 트래픽을 관리하고 분산하여 패킷

전송시 생기는 지연을 최소화 한다. 하지만 본 논문에서는 포트 번호 및 목적지의 IP주소를 기반으로 서비스 별로 우선순위를 부여하며, 패킷을 분류하고 각 서비스에 적합한 자원을 지니고 있는 링크를 탐색하여 해당 링크로 패킷을 전송하는 메커니즘을 제안한다. 이는 네트워크 자원을 효율적으로 사용할 수 있는 메커니즘으로써 각 서비스에 필요한 자원을 할당 하여 더 높은 QoS를 제공할 수 있는 장점을 지닌다. 그림 2는 본 연구에서 사용한 OpenDaylight 컨트롤러와 Data Plane의 구조도를 나타낸다. 아래 그림에서 TD(Traffic Definition) 패키지는 들어오는 패킷(Incoming Packet)을 분석하여 Flow Manager에게 전달하고 이는 Network Service Function에 의해 패킷에 대한 룰을 Data Plane에 전달하는 역할을 하는 모듈이다. 마지막으로 컨트롤러는 Data Plane인 OVS와 OpenFlow 프로토콜을 통해 통신한다.

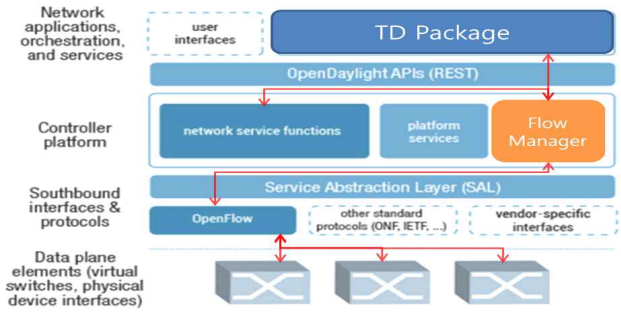

그림 2. 제안한 OpenDaylight의 구조도

아래 그림 3은 본 논문에서 제안한 알고리즘을 테스트해보기 위한 테스트베드의 토폴로지가 어떻게 구성 되었는지를 보여준다.

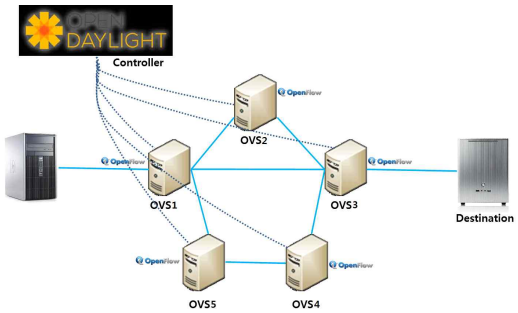

그림 3. 토폴로지 구성도

그림 3에서 OVS1, OVS2, OVS3을 통해 목적지로의 경로1에는 다른 링크들에 비해 더 높은 대역폭을 할당하여, 고 대역폭을 요하는 서비스, 즉 스트리밍 서비스와 같은 경우에는 컨트롤러에서 해당 패킷을 경로 1을 거쳐 목적지에 도착할 수 있도록 설정하였다. 또한 OVS1과 OVS3을 지나 목적지로 향하는 경로2에는 링크의 속도를 다른 링크에 비해 빠르게 설정하여 링크의 속도가 다른 요인들에 비해 우선시 되는 서비스의 경우 해당 경로로 패킷을 전송 할 수 있도록 하여 서비스에 따른 패킷 전송을 보장하는 알고리즘을 제안 하였다.

3.2. 서비스에 따른 패킷 전송시 지연 최소화 알고리즘

본 절에서는 서비스 별로 패킷을 분류하여 전송할 때, 발생하는 지연을 최소화하기 위한 알고리즘을 소개한다. 먼저, 3.1절에서 제안한 것처럼 각 서비스 패킷은 Flow Manager에 의해 특정 경로로 전송 되는 메커니즘을 갖는다. 이 때, 전송대기 시간을 최소화하기 위한 알고리즘이 그림 4에 의사코드로 정리되어 있다. 기본적으로 알고리즘의 과정을 살펴보면, 패킷이 전송대기 중일 경우 M/M/1 Queue 모델을 통한 Queue의 평균 대기시간($\overline{W}$)을 측정하여 $\overline{W}$값의 합이 가장 낮은 경로를 통해 패킷이 전송되는 것을 알 수 있다. 이는 서비스에 따라 패킷을 전송할 때, 각 패킷의 전송 대기시간의 감소와 더불어 Load Balancing의 효과를 얻을 수 있다는 장점을 갖는다.

```
Input:  <Packets with specific port number and destination IP address>
1:  Sending Packets
2:  if packets are not yet defined by the controller then
3:    TD Package ← incoming packet
4:  else Search for optimal path according to the service
5:    if the corresponding path is overloaded then
6:      for(number of the paths -1)
7:        P=CompWaitTime // Return Path which has the shortest queue waiting time
8:      endfor
9:      send p through Path L
10:   else
11:   send p through the current link
12:   endif
13: endif
Input:  λ, μ
1:  Comparing Queue Waiting Time in M/M/1 Queue Model
2:  for(number of switches in the path_i)
3:      sum_i = sum of W̄ on each switch in i path
4:  endfor
5:
6:  * λ indicates the packet arrival rate
7:  * The service times are assumed to be independent and exponentially
8:    distributed with parameter μ
9:  * utilization can be represented and calculated as ρ = λ/μ
10: * mean number of packets in the queue is N̄ = ρ/(1−ρ)
11: * finally W̄ can be calculated as W̄ = N̄ 1/μ
```

그림 4. 알고리즘 의사 코드

4. 시스템 구축 환경 및 서비스별 패킷 전송 테스트

4.1. 시스템 구축 환경

표 1은 본 연구에서 제안한 알고리즘의 구현을 위하여 테스트베드 구축 시 시스템 구축 환경을 보여준다. 컨트롤러 1대, OVSwitch 5대와 Source 및 Destination을 포함하여 8대의 장비를 이용하여 시스템을 구축하여 테스트를 진행 하였다.

표 1. 시스템 구축 환경

	운영체제	비고
컨트롤러	Ubuntu 12.04	OpenDaylight (OpenFlow v 1.3)
OVS1	Ubuntu 14.04	OVS v2.0.2
OVS2	Ubuntu 14.04	OVS v2.0.2
OVS3	Ubuntu 14.04	OVS v2.0.2
OVS4	Ubuntu 14.04	OVS v2.0.2
OVS5	Ubuntu 14.04	OVS v2.0.2
Source/ Destination	Ubuntu 12.04	Source : 패킷 생성 Destination : 모니터링

4.2. 서비스별 패킷 전송 테스트

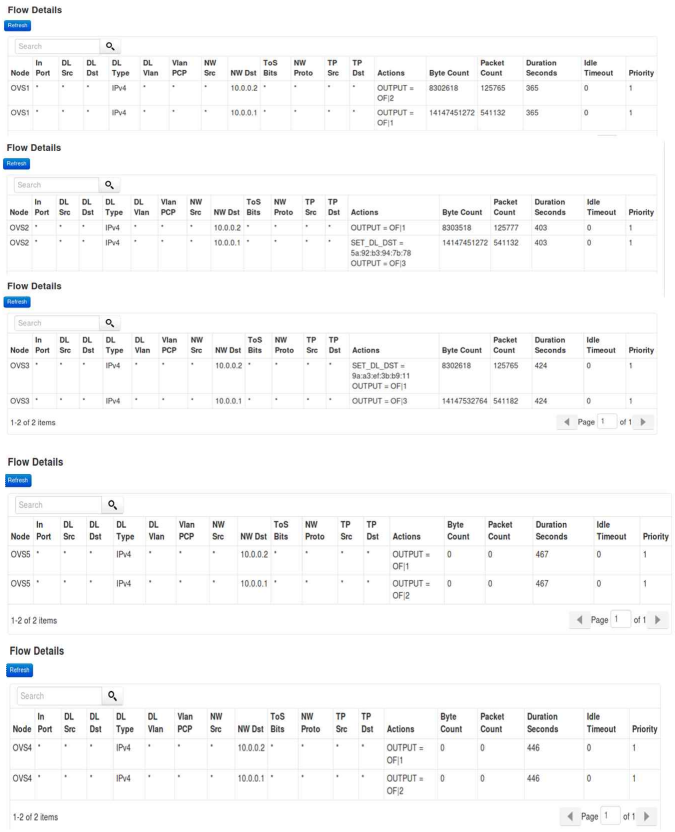

그림 5. 스트리밍 서비스 패킷 전송 결과

그림 5는 스트리밍 서비스를 일례로 하여, 패킷을 전송하고 각 OVS에 패킷과 바이트를 카운트하여 나타내주는 OpenDaylight 컨트롤러의 화면이다[5]. 위 그림에서 보이는 것처럼 스트리밍 서비스는 경로 1을 통해 목적지에 전달되므로, OVS1, OVS2, OVS3에서만 패킷이 카운트 되는 것을 알 수 있다. 이를 통해 서비스 별로 패킷이 전송 되는 것을 확인 할 수 있다.

5. 결론 및 향후 연구

본 연구에서는 SDN에 기반한 효율적인 네트워크 자원의 활용과 더불어 지연을 최소화하는 알고리즘을 제안하였다. 이는 서비스별로 패킷을 분류하고 모니터링하여 네트워크 및 서비스 관리 차원에서 장점을 지닌다고 할 수 있다. 또한 특정 서비스에 우선순위를 주어 패킷을 전송 및 처리하는데 있어 지연을 최소화 할 수 있다. 하지만 IPv4를 이용한 서비스의 분류에는 한계가 있어 차후, IPv6를 이용한 연구가 진행 되어야 할 것이다. 또한 향후 연구에서는 네트워크 자원의 활용 비율, 지연 등과 같은 성능을 평가하고 기존 Legacy 장비들과 비교를 통해 더 나은 서비스 제공 및 효율적인 네트워크 관리가 가능하도록 할 것이다.

6. 참 고 문 헌

[1] Bruno Astuto A. Nunes, Marc Mendonca, Xuan-Nam Nguyen, Katia Obraczka, and Thierry Turletti, “A survey of software-defined networking: Past, present, and future of programmable networks”, IEEE COMMUNICATIONS SURVEYS & TUTORIALS, VOL. 16, NO. 3, pp. 1617-1634 THIRD QUARTER 2014

[2] Hyojoon Kim and Nick Feamster, “Improving Network Management with Software Defined Networking“, IEEE Communications Magazine February 2013, pp. 114-119, February 2013

[3] Volkan Yazc, M. Oğuz Sunay, Ali O Ercan, “Controlling a Software-Defined Network via Distributed Controllers“, Proceedings of the 2012 NEM Summit, pp. 16-20, Istanbul, Turkey, October 16-18, 2012

[4] Sezer, S, Scott-Hayward, S., Chouhan, P.K., Fraser B., Lake, D., Finnegan, J., Viljoen, N., Miller, M. Rao, N., “Are We Ready for SDN? Implementation Challenges for Software-Defined Networks, IEEE Communications Magazine, pp. 36-43 July 2013

[5] https://git.opendaylight.org/gerrit, http://www.opendaylight.org/software/getting-started