논문 15-40-06-16 The Journal of Korean Institute of Communications and Information Sciences '15-06 Vol.40 No.06
http://dx.doi.org/10.7840/kics.2015.40.6.1114

SDN에서 컨트롤러 간의 부하 분배 방법

경 연 웅•, 홍 기 원*, 박 성 호**, 박 진 우°

Load Distribution Method over Multiple Controllers in SDN

Yeunwoong Kyung•, Kiwon Hong*,
Sungho Park**, Jinwoo Park°

요 약

SDN (Software-defined networking)에서는 중앙 집중적 구조로 인해 플로우 단위의 세밀한 트래픽 관리과정에서 컨트롤러의 병목 현상이 발생하여 입력되는 메시지들의 차단 및 처리 지연이 증가할 수 있다. 본 논문에서는 이 문제를 해결하기 위해 다수의 컨트롤러들 간의 플로우 단위의 부하 분배 방법을 제시하고 그 효과를 분석하였다. 제안된 방법은 특정 컨트롤러에 부하가 집중되었을 때, 새로 발생하는 플로우의 처리 및 관리를 다른 컨트롤러로 위임함으로써 컨트롤러들 간의 부하를 분배시킨다. 성능분석을 통하여 입력되는 메시지들에 대한 차단 확률이 낮아짐을 입증함으로써 제안된 방법의 효과를 증명하였다.

Key Words : SDN, Load Distribution, Scalability

ABSTRACT

In this paper, we propose a load distribution scheme in SDN utilizing load redirection, enabling incoming messages to be migrated to another controller. Specifically, when the capacity of a controller reaches a threshold, the controller makes incoming packets be migrated to a less-loaded controller to prevent them from being blocked. Analytical result shows that our scheme has lower blocking probability than the conventional scheme.

Ⅰ. 서 론

SDN (Software-Defined Networking)은 네트워크 장비들의 제어 평면 (control plane)을 전송 평면 (data plane)으로부터 분리시켜서[1], 개방형 인터페이스를 이용한 중앙 집중적인 관리를 통해 새로운 서비스의 신속한 구현 및 플로우 단위의 유연한 네트워크 관리를 가능하도록 하였다[2].

하지만 플로우 단위의 중앙 집중적인 특성은 확장성 문제를 야기할 수 있다. 즉, 플로우 단위의 엔트리 생성 및 관리로 인해 컨트롤러에 병목현상이 발생하여 입력되는 메시지들의 차단 및 처리 지연이 발생할 수 있다. 다수의 컨트롤러를 이용하여 이러한 문제를 해결하고자 하는 연구들이 있었지만, 특정 플로우들로 인해 부하가 집중되어 컨트롤러 간 부하 불균형 문제를 고려하지 못하였다.

부하가 큰 컨트롤러의 스위치를 다른 컨트롤러로 이전시켜 부하 균형을 추구한 기존 연구도 있었지만[3] 스위치 단위로 이전시키기 때문에 세밀한 부하 분배가 불가능하고 스위치의 담당 컨트롤러 변경을 위한 추가적인 configuration 과정이 요구되며 가변적인 부하로 인한 핑퐁현상이 발생할 수 있다.

그러므로 본 논문에서는 다수의 컨트롤러가 존재할 때, 컨트롤러의 부하가 정해진 한계치에 도달하면 이후에 입력되는 플로우들의 처리 및 관리는 다른 컨트롤러에 위임함으로써 컨트롤러들 간의 부하를 분산시킬 수 있는 방법을 제안하고자 한다.

Ⅱ. 컨트롤러 간의 부하 분배 방법

제안하는 방법을 설명하기 위해, 다음과 같은 시나리오를 가정한다. 컨트롤러는 다수의 액세스 스위치를 관리하고, 스위치들마다 자신을 관리하는 디폴트 컨트롤러가 설정되어 있으며, 컨트롤러들끼리는 부하 정보를 공유한다.

제안된 방법의 수행 절차는 그림 1과 같다. 모든 절차는 OpenFlow 표준[4]에 근거하여 동작한다. 액세스

※ 본 연구는 미래창조과학부 및 정보통신기술연구진흥센터의 정보통신·방송 연구개발사업의 일환으로 수행하였음. [B0101-15-1272, 단말 협업형 Giga급 스마트 클라우드릿 핵심기술 개발]
• First Author : Korea University School of Electrical Engineering, ywkyung@korea.ac.kr, 학생회원
° Corresponding Author : Korea University School of Electrical Engineering, jwpark@korea.ac.kr, 종신회원
* Korea University School of Electrical Engineering, shalaman@korea.ac.kr, 학생회원
** Korea University School of Electrical Engineering, kuyt@korea.ac.kr
논문번호 : KICS2015-04-137, Received April 24, 2015; Revised May 20, 2015; Accepted June 9, 2015

스위치는 단말의 패킷이 입력되면, 플로우 엔트리들과 해당 패킷과의 매칭을 수행한다. 매칭되는 엔트리가 없다면, 스위치는 그 패킷을 Packet-In 메시지로써 디폴트 컨트롤러인 컨트롤러 1에게 전송한다. 컨트롤러 1은 자신의 부하 상태가 한계치에 도달했기 때문에, 공유된 부하정보를 기반으로 해당 패킷을 전달할 컨트롤러들을 설정한다. 공유된 정보가 가변적인 부하상황 및 오류 등으로 인해 오차가 발생할 수 있기 때문에 해당 패킷의 처리 확률을 높이기 위해 부하가 낮은 순서대로 선택된 N개의 컨트롤러에게 전달하도록 한다. 해당 패킷을 전달 받은 컨트롤러들은 자신의 부하 상태가 해당 패킷을 수용할 수 있다면 패킷을 처리하여 플로우 규칙을 결정한다. 그리고 이 규칙을 전송 평면에 반영하기 위해 스위치에 Packet-Out 또는 Flow-Mod 메시지를 전송하여, 플로우 엔트리를 생성한다. 이 때, 다수의 컨트롤러로부터 메시지가 도착한다면 가장 먼저 도착한 메시지의 컨트롤러 (컨트롤러 2)가 담당 컨트롤러가 된다. 이후, 해당 플로우에 속한 패킷들은 생성된 엔트리와의 매칭에 따라 전달되고, 컨트롤러 2는 해당 플로우의 관리를 담당하게 된다. 이렇게 플로우 별로 다른 컨트롤러가 할당될 수 있지만, 분산된 데이터 저장 방법을 이용하여 글로벌한 네트워크 뷰를 공유한다고 가정한다[3]. 또한 중복 처리가 없도록 각 엔트리 설정에 대한 처리는 반드시 하나의 컨트롤러가 담당하는 것을 가정한다.

Ⅲ. 시스템 모델링

제안하는 기법의 컨트롤러는 M/M/m/m 큐잉 모델을 가정하며, 현재 컨트롤러 영역에서 발생한 새로운 플로우의 Packet-In 메시지와 다른 컨트롤러에서 전달된 Packet-In 메시지, 이렇게 두 종류의 메시지가 존재한다. 기존 방법[4]은 다수의 컨트롤러를 고려하였지만 컨트롤러 간 분배를 고려하지 않았기 때문에, 다른 컨트롤러에서 전달된 메시지는 존재하지 않는다. 새로운 플로우로 인한 Packet-In 메시지 및 다른 컨트롤러부터 전달된 Packet-In 메시지의 발생 프로세스가 각각 λ_I, λ_O을 평균으로 갖는 포아송 분포 (Poisson Distribution)를 따른다고 가정하고, 컨트롤러에서의 서비스 시간 프로세스는 μ를 평균으로 갖는 익스포넨셜 분포 (Exponential Distribution)를 따른다고 가정한다. 이를 통해, 총 부하는 ρ_I (=λ_I/μ)과 ρ_O (=λ_O/μ)의 합으로 나타낼 수 있다. 그림 2(a)와 2(b)는 각각 기존 방법과 제안된 방법의 마르코브 체인 (Markov chain) 모델이다.

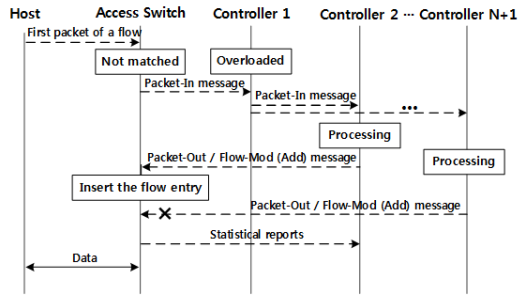

그림 1. 제안된 방법의 수행 절차
Fig. 1. Message flow in the proposed scheme

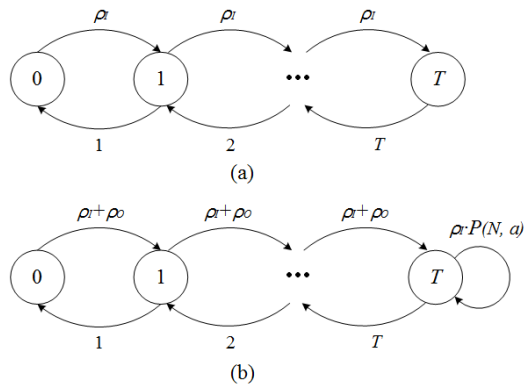

그림 2. 기존 방법[4]과 제안된 방법의 마르코브 체인 모델
Fig. 2. Markov chain model of the conventional and proposed schemes

T는 컨트롤러의 최대 수용량을 의미한다. 두 방법의 정상 상태 (steady state) 확률 분포를 구하기 위해서는 2차원 마르코브 체인 모델이 필요하지만, 분석의 간편함을 위해 본 논문에서는 근사적인 방법[5]을 이용한다. 이 방법에서는 그림 2에서와 같이 서비스율을 1로 만든 뒤에 분석이 진행된다. 제안된 방법의 i상태에서 i+1상태로 천이될 확률 (transition rate)은 (ρ_I + ρ_O)이며, 기존 방법에서는 (ρ_I)이다. 두 방법 모두, i+1 상태에서 i상태로의 천이 확률은 (i+1)이다. 천이 확률을 통하여, 제안된 방법의 정상 상태 확률 (p_k)을 구하면 식 (1)과 같이 나타난다.

$$p_k = \frac{(\rho_I+\rho_O)^k/k!}{\sum_{m=0}^{T}(\rho_I+\rho_O)^m/m!} \quad (1)$$

식 (1)을 통해 새로운 메시지의 차단 확률 (P_B^I)을 구하면 식 (2)와 같다. 차단 확률은 메시지가 컨트롤러에 입력될 때, 컨트롤러 부하가 최대치에 이르러서

(식 (1)에서 k=T: p_T), 해당 메시지가 N개의 다른 컨트롤러들로 전달됨에도 불구하고 처리되지 못하고 차단될 확률을 의미한다.

$$P_B^I = p_T \cdot P(N,a) \quad (2)$$

식 (2)에서 $P(N,a)$는 각 컨트롤러가 p_a의 확률로 새로운 메시지를 수용할 수 있을 때, N개의 컨트롤러들이 모두 새로운 메시지를 수용할 수 없는 확률을 의미하며 $(1-p_a)^N$ 로 표현된다. 기존 방법[4]의 경우, 다른 컨트롤러로부터 입력되는 메시지가 없기 때문에 정상상태 확률은 식 (1)에서 입력되는 부하를 (ρ_I)로 변경해주면 얻을 수 있으며, 이를 통해 차단 확률은 p_T로 표현된다.

Ⅳ. 성능 평가

이 장에서는 위에서 도출한 모델을 기반으로 제안하는 방법과 기존 방법[4]의 메시지 차단 확률을 비교하였다. 성능 평가를 위해 T, ρ_D은 각각 50, 10으로 가정하였다. 그림 3은 컨트롤러에 입력되는 메시지의 양이 증가할 때, 제안된 방법과 기존 방법의 차단 확률을 나타낸다. 제안 방법 (a, b)에서 a와 b는 각각 N과 p_a를 의미한다. 그림 3을 통하여 제안된 방법은 부하가 집중될 때, 다른 컨트롤러로 부하를 전이시킴으로써 기존 방법보다 낮은 차단 확률을 보임을 알 수 있다. 또한 N과 p_a가 증가할수록 전이되는 메시지가 처리될 확률이 커지기 때문에 차단 확률이 낮아지는 결과를 보인다. 하지만 N이 커질수록 오버헤드가 증가할 수 있기 때문에 네트워크 운영자 입장에서 네트워크 상황에 따라 적절하게 설정해야 한다.

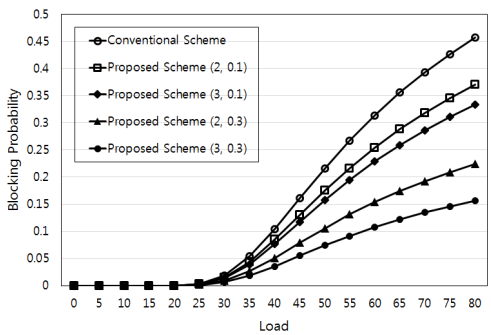

그림 3. 제안된 방법과 기존 방법의 차단 확률
Fig. 3. Blocking probability of the proposed scheme and conventional scheme

Ⅴ. 결 론

본 논문은 특정 컨트롤러에 부하가 집중될 때, 다른 컨트롤러로 부하를 전이시켜 처리함으로써 컨트롤러들 간의 부하 분배를 수행할 수 있는 방법을 제안하였다. 성능 평가를 통해 제안된 방법이 기존 방법보다 낮은 차단 확률을 보임을 확인하였다. 향후 컨트롤러 간 부하분배를 위한 오버헤드 분석 및 가상환경에서의 실험을 진행할 예정이다.

References

[1] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: Enabling innovation in campus networks," in *Proc. ACM SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 2, pp. 69-74, 2008.

[2] T. Yim, Y. Kyung, T. M. Nguyen, K. Hong, and J. Park, "A fast and scalable mobile flow management method for IP-based mobile networks," *J. KICS*, vol. 39B, no. 1, pp. 8-16, Jan. 2014.

[3] A. Dixit, F. Hao, S. Mukherjee, T. V. Lakshman, and R. Kompella, "Towards an elastic distributed SDN controller," in *Proc. ACM HotSDN*, pp. 7-12, Hong Kong, China, Aug. 2013.

[4] OpenFlow switch specification 1.5.1 (2015), April 2015, from https://www.opennetworking.org/sdn-resources/openflow

[5] S. Jeon, R. L. Aguiar, and N. Kang, "Load-balancing proxy mobile IPv6 networks with mobility session redirection," *IEEE Commun. Lett.*, vol. 17, no. 4, pp. 808-811, Apr. 2013.