SDN 컨트롤러 클러스터 환경에서 ODL 컨트롤러의 데이터 처리 성능 평가

구영훈, 심규석, 정우석, 김명섭

고려대학교

{gyh0808, kusuk007, hary5832, tmskim}@korea.ac.kr

Data Processing Performance Evaluation of ODL Controller in a SDN Controller Cluster Environment

Young-Hoon Goo, Kyu-Seok Shim, Woo-Suk Jung, Myung-Sup Kim

Korea Univ.

요 약

최근 새로운 기술 요구 사항에 빠르게 대응하기 위한 기술로 SDN이 시장의 광범위한 관심을 받고 있으며 캐리어 네트워크로 그 적용 범위를 확대시켜 나가고 있다. 클라우드 네트워크를 넘어서서 엔터프라이즈 및 유무선사업자망으로 SDN이 확산 되기 위해서는 확장성과 고가용성 지원을 위한 기술이 필수적이다. 본 논문에서는 고가용성 지원을 위하여 클러스터링 기능을 구현한 ODL의 클러스터 구조 및 규모에 따른 데이터 처리 명령 수행의 시간을 측정하여 성능 변화 양상을 파악하고 원인을 분석한다.

Ⅰ. 서 론

최근 네트워크 가상화와 중앙 집중형 제어를 통해 네트워크의 신속한 재구성과 트래픽 제어를 가능하게 하는 SDN(소프트웨어 정의 네트워크)는 새로운 기술 요구 사항에 빠르게 대응하기 위한 기술로 시장의 광범위한 관심을 받고 있다. 또한 최근 캐리어 네트워크로 그 적용 범위로 확대시켜 나가고 있는 실정이다.[1]

클라우드 네트워크를 넘어서서 엔터프라이즈 및 유무선 사업자망으로 SDN이 확산되기 위해서는 확장성과 고가용성 및 고성능을 지원할 수 있는 상용수준의 컨트롤러 플랫폼이 필요하다. 최근 각광받고 있는 SDN 컨트롤러 중 고가용성을 지원하는 SDN 컨트롤러에는 ODL(OpenDayLight)과 ONOS(OpenNetworkOperatingSystem)가 대두되고 있다. 고가용성 / 고확장성을 제공하기 위해 ODL에서는 Akka 프레임워크를 기반으로 클러스터링 기능을 구현하며 데이터 동기화를 위한 Raft 알고리즘을 이용하여 고가용성을 지원한다. ONOS에서는 분산처리 기능을 제공하는 Zookeeper를 활용하며 클러스터 내의 데이터 동기화 기능을 제공하는 Hazelcast를 활용한다.

한편, 상용화망에서의 안정된 서비스 지원을 위해 컨트롤러를 클러스터링으로 구성함으로써 가용성은 올라갈지라도 이에 따른 부작용으로 서비스 시간이 급격하게 느려진다면 큰 문제가 아닐 수 없다. 이에 본 논문에서는 단일 컨트롤러 환경과 클러스터링을 적용한 컨트롤러 환경에서 클러스터의 규모를 달리하며 데이터 read, write 수행 시간을 비교하고 수행시간의 변화 양상의 원인을 분석한다.

ODL 클러스터 구조 및 규모에 따른 데이터 처리 수행 양상에 대하여 section 2에서 설명하고 실험 결과를 section 3에서 설명한다. 마지막으로 section 4에서는 결론을 기술한다.

Ⅱ. ODL 클러스터 구조 및 규모에 따른 데이터 처리 수행 양상

ODL에서는 클러스터의 분산처리 작업에서의 신뢰성을 제공하기 위하여 Raft 알고리즘을 사용한다. Raft 알고리즘은 데이터를 복제하여 시스템의 일부에 이상이 생겨도 완전한 기능을 유지할 수 있게 해주는 알고리즘으로 데이터 복제를 위해 클러스터의 구성 컨트롤러들을 Leader, Follower, Candidate의 세 가지 상태로 정의한다. Leader는 모든 사우스바운드와 상호작용하며 데이터의 피복제노드가 된다. Follower는 Leader를 제외한 모든 노드로 Leader로부터 데이터를 복제 받는다. Candidate는 새로운 Leader 선출을 위한 상태로 시스템이 정상적인 경우 모든 노드는 Leader와 Follower의 상태를 유지한다.[2]

본 논문에서는 크게 세 가지 관점을 가지고 이를 복합적으로 종합하여 데이터 처리 수행 양상을 분석한다. 첫째는 클러스터 규모로 단일 컨트롤러와 3-Node 클러스터와 5-Node 클러스터, 둘째는 클러스터에서의 노드 상태인 Leader와 Follower, 셋째로 데이터 수행 명령인 read와 write이다.

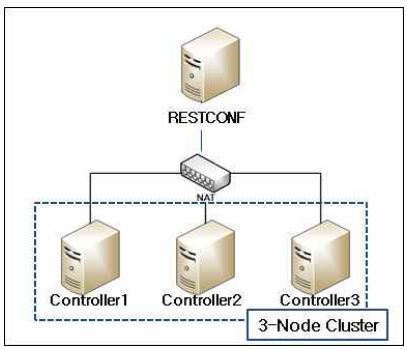

그림 1. 실험 환경(3-Node 클러스터)

실험 환경은 하나의 PC에서 RESTCONF를 이용하여 데이터 처리 명령을 전달하는 VM과 컨트롤러를 구성하는 VM들을 구축하였다. 그림 1은 실험 환경을 도식화한 것으로 대표적으로 3-Node 클러스터 환경을 나타낸 것이다. 이를 5-Node 클러스터 환경, 단일 컨트롤러 환경으로 바꾸어 가며 실험을 진행한다.

표 1은 실험을 위하여 앞서 언급한 세 가지의 관점을 복합적으로 종합하여 도출한 총 10가지 Case로 각각의 Case에 대하여 데이터 처리 수행 시간 및 데이터 동기화 시간을 측정하였다.

이 논문은 BK21 플러스 사업(No. T1300573) 및 2015년도 정부(교육부)의 재원으로 한국연구재단 기초연구사업의 지원(No.2015R1D1A3A01018057) 및 2015년도 정부(미래창조과학부)의 재원으로 정보통신기술진흥센터의 지원(B0101-15-233, 스마트 네트워킹핵심기술개발)을 받아 수행된 연구임.

Case	클러스터 규모	노드 상태	요청 명령
1	1(단일)	-	Read
2		-	Write
3	3-Node	Leader	Read
4			Write
5		Follower	Read
6			Write
7	5-Node	Leader	Read
8			Write
9		Follower	Read
10			Write

표 1. 데이터 처리 수행 및 동기화 시간 측정 케이스

Case1과 2는 단일 컨트롤러 환경에서의 read와 write 명령 요청, Case3, 4, 5, 6은 3-Node 클러스터 환경으로 Case3과 4는 Leader에게 read와 write 명령 요청, Case5와 6은 Follower에게 read와 write 명령 요청을 의미하며 Case7, 8, 9, 10은 5-Node 클러스터 환경으로 Case7과 8은 Leader에게 read와 write 명령 요청, Case9와 10은 Follower에게 read와 write 명령 요청을 의미한다.

그림 2는 클러스터 환경에서의 Leader가 write 요청을 받았을 경우 트래픽 추이를 나타내며 그림 3은 Follower가 write 요청을 받았을 경우 트래픽 추이를 나타낸다.

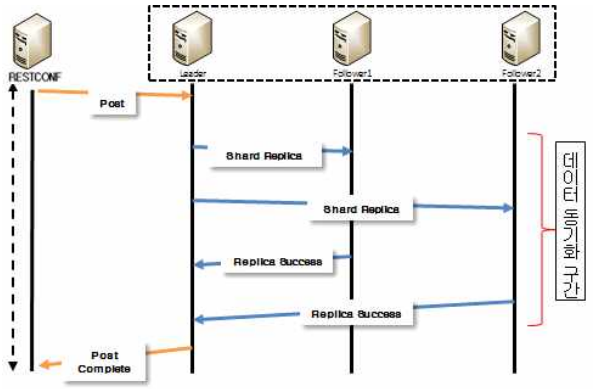

그림 2. Leader가 write 요청 받았을 경우 트래픽 추이

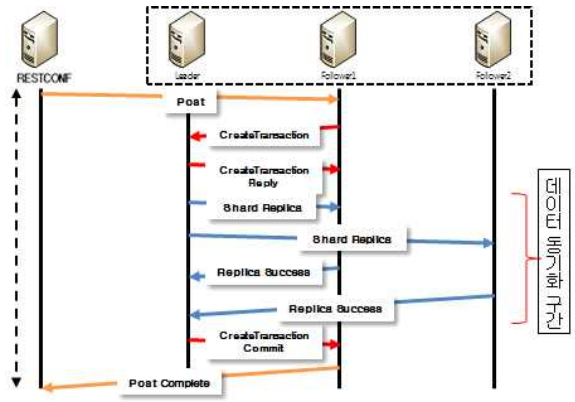

그림 3. Follower가 write 요청 받았을 경우 트래픽 추이

Ⅲ. 실험 결과

컨트롤러 버전은 ODL Lithuim SR1이며 데이터 처리 수행의 대상이 되는 shard는 ODL Integration Group 프로젝트에서 제공하는 car shard이다. 이 shard의 500개의 레코드를 클러스터 규모를 변경시키며 Leader와 Follower에게 read, write 요청을 표 1의 각 case별로 총 5회씩 수행하여 데이터 처리 수행 시간과 데이터 동기화 시간의 평균을 구하였다.

표 2는 실험 결과를 나타낸 것이다. 실험 결과를 보면 클러스터의 규모가 커질수록 write 요청의 총 수행시간이 증가하는 것을 확인할 수 있다. 한편, read 요청의 경우 단일 컨트롤러 환경, 3-Node 클러스터 환경에서 Leader에게 요청하였을 때와 Follower에게 요청하였을 때, 5-Node 클러스터 환경에서 Leader에게 요청하였을 때와 Follower에게 요청하였을 때의 총 수행시간의 변화가 크게 없음을 확인할 수 있다.

이는 Raft 알고리즘으로 인한 것으로 shard의 변화를 초래하는 write 요청의 경우 Leader는 요청을 수행하기 전에 Follower들에게 데이터를 복제해야 하므로 데이터 동기화 시간이 필요하며, 클러스터의 규모가 커질수록 데이터 동기화를 수행할 Follower들이 많아지게 되고, 이에 따라 데이터 동기화 시간이 증가하므로 총 수행시간도 증가하게 되는 것이다. 또한, Leader에게 write 명령을 요청하였을 때보다 Follower에게 write 명령을 요청하였을 때 데이터 처리 수행 시간이 증가하는 이유는 그림 2와 같이 Leader에게 요청을 할 경우 바로 전체 Follower들에게 데이터 동기화를 한 후 명령을 수행하지만 Follower에게 요청을 할 경우 그림 3과 같이 요청을 받은 Follower는 요청을 받은 내용을 Leader에게 먼저 전달을 하게 되고 Leader는 전달받은 내용을 Follower에게 잘 받았다고 응답한 후 전체 Follower들에게 데이터 동기화를 한 후 명령을 수행하기 때문이다. 반면에 shard의 변화가 없는 read 요청의 경우 데이터 동기화를 할 필요가 없으며 Leader든 Follower든 read 요청을 받은 노드의 데이터를 그대로 읽어오면 되기 때문에 read는 총 수행시간의 변화가 없게 된다.

클러스터 규모	1	3-Node		5-Node	
노드 상태 / 요청 명령	-	Leader	Follower	Leader	Follower
Read	1.15 (-)	1.18 (-)	1.16 (-)	1.16 (-)	1.18 (-)
Write	4.89 (-)	7.88 (2.66)	11.64 (5.18)	9.34 (4.08)	12.50 (6.00)

표 2. 데이터 처리 수행 시간(데이터 동기화 시간) 단위:ms

Ⅳ. 결론

본 논문에서는 ODL 클러스터 구조, 즉 Raft 알고리즘의 대상인 Leader와 Follower, 클러스터 규모의 변화에 따른 read와 write의 데이터 처리 수행에 관한 성능을 평가하고 그 원인을 분석하였다. 실험을 통하여 read 명령의 경우 클러스터 구조 및 규모에 관련없이 수행시간이 비슷하다는 것을 확인하였고 단지 데이터의 개수에만 영향을 받을 것으로 판단된다. write의 경우 Leader에게 요청을 하였을 경우 보다 Follower에게 요청을 하였을 경우가 그리고 클러스터 규모가 커질수록 수행시간이 증가하며 위의 결과는 ODL의 Raft 알고리즘에 의해서임을 분석하였다.

효율적인 SDN 운용을 위해서는 shard의 변화를 초래하는 데이터 write 요청의 빈도가 높은 데이터의 경우, 클러스터 크기에 따른 오버헤드를 고려하여 적당한 수의 클러스터를 구성하여 운용하는 것이 합리적이라고 사료된다.

참 고 문 헌

[1] 유재형, 김우성, 윤찬현, 'SDN/OpenFlow 기술 동향 및 전망', KNOM Review, 15.2 (2012)

[2] 정우석, 구영훈, 심규석, 김명섭. "분산 컨트롤러 클러스터 환경에서의 ODL 클러스터 고가용성 성능평가", 2015년도 한국통신학회 추계종합학술발표회 (2015)