TCP splicing 기반의 2단계 L4/L7 부하분산방법을 이용한 확장성 있는 클러스터형 웹서버의 설계

이 진, 권희웅, 정 규 식, *김동승
숭실대학교 정보통신전자공학부, *고려대학교 전기공학과

A Design of Scalable Clustering Web Server Using Two-level L4/L7 Load Balancing Scheme based on TCP splicing

Jin Lee, Huiung Kwon, Kyusik Chung, Dongseung Kim
School of Electronic Engineering, SoongSil University
E-mail : innurman@intizen.com

Abstract

중앙집중식 구조의 기존 내용기반 요구분산의 문제점인 확장성 및 성능을 향상시키기 위해, 본 논문에서는 2단계로 구성된 분산구조로 된 L4/L7 방법을 사용한다. L4 스위치를 이용하여 1차적으로 부하를 분산시키고, 이들에 대해 proxy의 성능상의 단점을 보완한 TCP splicing을 적용하여 2차로 L7 스위치 기능을 수행하도록 한다. 리눅스 환경에서 제안한 시스템을 구현하고 클러스터형 웹서버 시스템을 구성하여 실험하였다. 제안한 분산구조 방법이 중앙집중 방식에 비해 확장성, 성능면에서 우수함을 확인하였다.

I. 서론

클러스터형 웹서버는 클라이언트의 요구를 여러대의 서버에서 나누어 처리하여 늘어나는 트래픽에 대해 유연성을 제공한다. 확장성은 노드수에 따른 성능의 증가를 말하는 것으로 많은 벤치마크 결과에서 제품들 간의 비교우위 항목으로 존재한다.

L4 스위치는 TCP/IP 패킷레벨을 기반으로 부하를 분산시키고 BE로 포워딩 방식에 따라 direct routing과 NAT 방식으로 나뉘며, 확장성이 뛰어나다. 반면에 L7 스위치는 내용기반의 분하분산(content aware request distribution)을 통해 많은 이점을 가지고 있지만 성능상이나 확장성 측면에서는 뛰어나지 않다.

따라서 본 논문에서는 L7스위치를 구조적인 측면에서 개선하고 P-HTTP(HTTP/1.1 Persistent connection)의 효율적인 지원을 통하여 확장성 있는 클러스터형 웹서버 설계방법을 제안한다.

II. 연구배경

2.1 내용기반의 요구분산

L7 스위치의 이점은 1) 오브젝트에 따라 파일 적재를 분산화 하여 자원을 효율적으로 이용할 수 있는 파티션(partitioning)이 가능한 것과 2) 클래스의 우선순위에 따른 요구의 차등 처리를 하는 Web- QoS가 가능하다는 점이다. [그림 1]은 L7 스위치의 일반적인 예를 보여주고 있다.

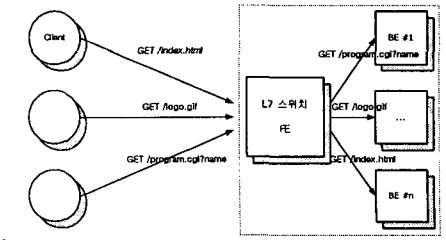

그림 3 L7 스위치의 동작

그러나 내용기반의 처리를 하기 위해서는 front-end(FE)에서 클라이언트의 요구에 대한 HTTP 헤더정보를 분석하여 back-end(BE)로 전달하고, BE의 응답을 FE를 통해서 클라이언트에게 전달한다. 기본적으로 클라이언트의 요구를 분석하기 위해서는 TCP/IP 스택을 거쳐야 하기 때문에 L4 스위치보다 많은 부하가 걸린다. 또한 FE를 통해서 모든 클라이언트의 요구가 분석되기 때문에 부하가 집중되어 성능이나 확장성이 좋지 않다.

2.2 기존 내용기반의 요구분산 구현 방법

HTTP/1.1의 Persistent Connection은 하나의 커넥션을 통하여 다중 요구/응답이 가능하도록 하여 커넥션 생성에 드는 비용을 줄여 성능을 높일 수 있다. 따라서 이에 대한 지원은 클러스터형 웹서버 전체의 부하를 줄일 수 있기 때문에 성능이 높아진다. [1]과 [2]에서는 P-HTTP의 지원 여부와 구현 방법에 대한 장단점을 제시하고 있다.

(1) Proxy

proxy 방식은 용이한 구현이라는 이점을 가지고 있다. FE는 클라이언트와 BE 간의 커넥션을 각각 맺고 있으며, 클라이언트의 요구를 읽어서 BE로 전해주거나 반대로 BE의 응답을 클라이언트로 전해준다.

그러나 일반적으로 유저레벨에서 데이터를 읽고, 쓰기 때문에 상당한 오버헤드가 발생하며 이는 HTTP 요구에 대한 처리량 감소와 CPU 이용도 증가, 그리고 응답지연시간을 늘이는 결과를 초래한다.

(2) TCP hand-off

TCP hand-off를 구현하여 클러스터형 웹서버에 적용한 연구는 [3]에 나타난 RICE 대학의 Locality-Aware Request Distribution(LARD)이 대표적이다. hand-off 메커니즘을 통해 P-HTTP을 지원하는 연구는 multiple hand-off와 back-end forwarding이 소개되고 있지만, 무엇보다 구현의 복잡성이 따르고 FE와 BE 양쪽에서 이를 지원하는 별도의 프로토콜을 가지고 있어야 하기 때문에 일반적인 상황에 적용하기에는 어려움이 따른다.

(3) TCP splicing

TCP splicing은 클라이언트와 서버를 각각의 커넥션으로 연결한다는 점에서는 proxy와 차이가 없지만, 데이터 패킷이 매번 TCP/IP 스택을 거치지 않고 클라이언트의 요구 분석 시에만 제한적으로 사용하도록 하고 대부분의 처리를 커널레벨에서 패킷의 일부 정보의 수정을 통하여 전달하는 방식을 통하여 성능을 높이도록 하였다.

III. 확장성 있는 L7 기반 클러스터링 시스템 설계

3.1 구조 설계

본 논문에서는 L7 스위치보다 성능이 뛰어난 L4 스위치를 이용하여 1차적으로 부하를 분산하고, 다수의 서버에 L7 스위치를 배치하여 2차로 클라이언트의 요구를 분산하도록 하는 두 단계의 접근 방법을 통하여 기존 중앙 집중식 L7 스위치의 단점인 확장성과 성능 문제를 해결할 수 있도록 하였다.

기본적인 구조 및 동작과정은 [그림 2]에 나타나 있다. 그림상의 FE는 L4 스위치 역할을 하여 1차로 부하를 분산시킨다. BE는 L7스위치와 웹서버가 함께 위치하고 1차로 분산된 부하를 BE상의 지정된 룰에 의한 L7 스위칭을 통해서 클라이언트의 요구가 적절하게 BE의 웹서버에 도달하도록 한다.

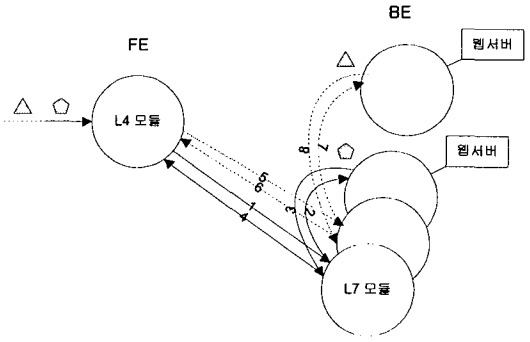

그림 4 L4/L7 모듈의 구성

3.2 L7 스위치의 구조

BE에서 L7의 기능을 적절히 수행할 수 있도록 설계하였다. [그림 3]과 다음은 두 모듈의 위치와 각각의 역할에 대한 설명이다.

(1) 커널레벨 L7 모듈

지정된 룰에 의해 클라이언트의 요구에 대한 해석을 수행하며, 해석된 결과에 따라 적절한 노드를 선택하는 역할을 수행한다. 클라이언트의 요구의 해석 및 전달은 proxy 방식으로 작동을 하며, 그 이후의 지정된

노드로부터 응답을 클라이언트로 전달할 수 있도록 TCP splicing 모듈을 제어한다. P-HTTP를 지원할 수 있도록 하기 위해 다중 요구에 대한 해석 및 전달이 반복적으로 수행하도록 구현되어야 하며, P-HTTP의 효율적인 지원 및 성능향상을 위한 커넥션 풀(connection pool)을 사용한다.

(2) TCP Splicing 모듈

TCP splicing을 담당하는 모듈로 NIC 드라이버와 IP 패킷 핸들러 사이에 위치하여 이를 통해 나가고 들어오는 모든 패킷을 모니터링 한다. TCP splicing이 지정된 커넥션은 투명성(transparency)을 보장하기 위해서 클라이언트와 노드 사이의 패킷 수정이나 생성이 적용된다. 그리고 P-HTTP를 위한 메커니즘을 지원하는데, 이는 다음절에서 자세히 설명한다.

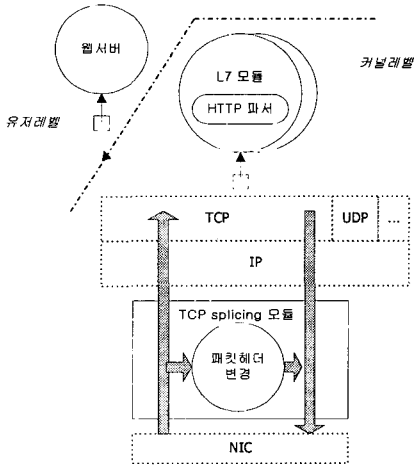

그림 5 TCP splicing 모듈

3.3 TCP splicing 구현

TCP splicing을 위해서는 클라이언트나 서버에서 들어오는 패킷의 TCP/IP 헤더변경이 필요하다. 이에 대한 구현은 [4]와 상당수 중복되므로 생략하도록 한다. [그림 4]는 BE를 구성하고 있는 L7 모듈과 TCP splicing 모듈, 웹서버 간의 커넥션 splicing이 일어나는 과정을 보여주고 있다. 먼저 클라이언트로부터 들어오는 요구는 FE의 L4 모듈의 스케줄링에 따라 BE로 전달된다. 이때 1) BE의 L7 모듈은 내부의 HTTP 파서에 의해서 요구정보를 읽고, 2) BE를 결정한다. 3) 선택된 BE로 커넥션이 연결된 뒤에는 패킷의 대부분을 차지하는 데이터 패킷이 4) 커널레벨에서 포워딩되는 과정을 볼 수 있다.

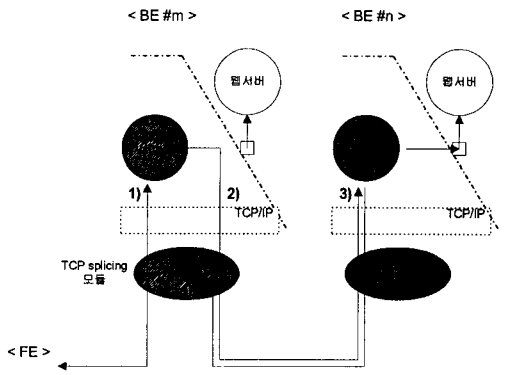

그림 6 요구/응답에 대한 커넥션 흐름

여기에서 소켓레벨과 커널레벨 사이의 위치차이로 인해서 하나의 요구(request)와 다음 요구 사이의 커널레벨 포워딩 과정이 소켓버퍼에는 반영이 안되기 때문에 문제가 발생한다. 같은 커넥션을 통해 들어오는 다음의 요구는 소켓으로 올라가기 때문에, 소켓버퍼의 관련 파라미터 상에서 이전 요구와 차이가 나는 것이다.

따라서 각각의 요구 사이에서 커널레벨 포워딩되는 데이터는 그에 해당하는 양만큼을 소켓레벨의 파라미터에 반영하는 것이 필요하다. 특히, 서버에서 오는 ACK의 경우 이를 단순히 클라이언트로 포워딩하는 것뿐만 아니라 새로 생성하여 TCP/IP stack 위의 소켓으로 올려주는 작업이 필요하다. [그림 5]는 소켓버퍼와 관련한 파라미터 {snd_buf: snd_una, snd_nxt, write_seq}와 {rcv_buf: rcv_nxt, copied_seq}를 나타내고 있다. 그리고 [그림 6]은 커널레벨에서 포워딩되는 ACK를 반영하기 위해서 이것이 도착하였을 때 새로 만들어서 소켓버퍼에 반영하는 것을 보여준다.

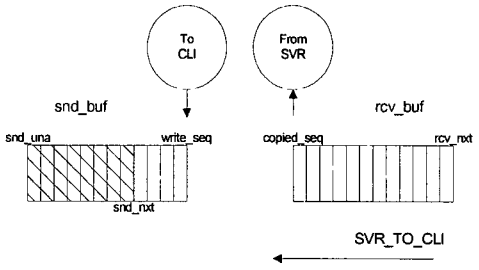

그림 7 소켓버퍼 관련 파라미터 수정

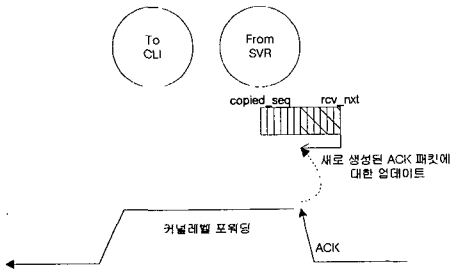

그림 8 ACK 패킷을 고려한 처리

성능을 위해서 TCP/IP options 필드를 통한 최적화가 필요하지만 구현의 편의를 위해서 최소한 부분만을 수정한다. TCP options의 필드에서는 Time-stamp 맵핑(mapping)을 통해 패킷 드랍(drop)을 막고, IP 옵션에서는 source route 등으로 야기될 커넥션의 뒤엉킴을 사전에 막도록 한다.

Ⅳ. 구현 및 실험

본 논문을 위한 모듈구현은 리눅스 커널 2.4.2-2에서 이루어졌으며, 실험은 FE 1대, 구현된 L7 스위치와 apache-1.9.19 웹서버가 설치된 9대의 BE를 100Mbps 이더넷 스위치를 통하여 클러스터로 구성하고, 2대의 Apache-Bench가 설치된 클라이언트를 통해서 부하를 생성하여 실험을 진행하였다.

제안된 방법에 대한 성능 및 확장성의 검증은 제안된 부하분산형 방법(a)과 현재 L7스위치에 가장 많이 사용되고 있는 TCP splicing을 통한 중앙집중형 방법(b)에 대해서 노드수에 따른 초당 처리능력(req/sec)을 기준으로 성능을 평가하였다.

실험결과 상에서 노드개수 3까지는 제안된 방법인 (a)가 저조한 결과를 보이고 있는데, 이는 제안된 방법에서 웹서버와 L7스위치가 같은 서버에서 스케쥴링 되기 때문이다. 하지만 노드개수가 4개 이상인 경우에 대해서는 제안한 방법의 성능이 일정하게 증가하는 경향을 보였다. 반면 (b)의 경우는 더이상 성능의 증가가 보이지 않았다.

결과적으로 제안된 방법이 각 서버에서 수행되는 L7 스위칭에 의해서 성능을 약간 감소시키는 경향은 있었지만 전체적인 성능이나 확장성면에서는 훨씬 우월하다는 것을 보여주었다.

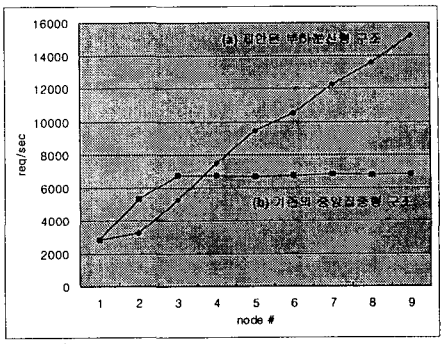

그림 9 제안방법에 대한 처리량 비교실험

Ⅴ. 결론

본 논문에서는 확장성 있는 클러스터형 웹서버를 설계하기 위해서 2단계의 부하분산형 구조를 채택하고 TCP splicing을 이용하여 P-HTTP를 지원하도록 하였다. 또, 제안한 방법을 구현과 실험을 통해서 BE에서 제안한 방법이 개별서버의 성능에 약간의 저하를 가지고 오지만 전체적인 확장성이나 성능 면에서 우수하다는 것을 입증하였다.

참고문헌(또는 Reference)

[1] Mohit Aron, Darren Sanders, Peter Druschel and Willy Zwaenepoel, "Efficient Support for P-HTTP in Cluster-based Web Servers", 2000Annual USENIX Technical Conference.

[2] Mohit Aron, Darren Sanders, Peter Druschel, Willy Zwaenepoel, "Scalable Content-aware Request Distribution in Cluster-based Network Servers", In Proceedings of the USENIX 2000.

[3] Vivek Pai, Mohit Aron, Gaurav Banga, Michael Svendsen, Peter Druschel, Willy Zwaenepoel, and Erich Nahum. "Locality-aware Request Distribution in Cluster-based Network Servers", In Proceedings of the Eighth International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS-VIII), 1998

[4] David A. Maltz, Pravin Bhagwat, "TCP Splicing for Application Layer Proxy Performance", IBM Technical Report RC 21139