

아파치 웹 서버의 성능 비교 분석

정기훈⁰ 염미령 노삼혁

홍익대학교 컴퓨터공학과, 전자계산학과, 정보컴퓨터공학부

{khchong, miryeom}@cs.hongik.ac.kr, samhnoh@hongik.ac.kr

Performance Measurement and Analysis of the Apache Web Server

Ki-hun Chong⁰ Mi-ryeong Yeom Sam H. Noh

Dept. of Computer Engineering, Computer Science, Information and Computer Engineering, Hongik Univ.

요 약

현재 웹서버 중 가장 많이 이용하는 아파치 웹서버는 안정적이며 다양한 기능을 제공한다. 특히, 차기 버전인 아파치 2.0은 다중 스레드 및 파일 캐시를 기본적으로 지원한다. 본 논문에서는 아직 까지 연구가 활발하지 않은 아파치 2.0에 대한 성능 분석을 하였다. 아파치 2.0과 1.3의 성능 비교에 대한 실험 결과 아파치 2.0은 1.3보다 처리율이 최고 25% 향상된 모습을 나타내었고, 사용자 응답시간은 평균 23% 향상되었다.

1. 서론

개인용 컴퓨터와 통신 환경이 발달함에 따라 인터넷도 눈부시게 발전하였다. 그에 따라 WWW역시 발전하였으며, 많은 사용자가 웹을 이용하고 있다. 이에 따라 웹서버의 중요성 또한 커지게 되었는데, 끊임없이 증가하는 사용자의 요구를 만족시키기 위하여 웹서버 역시 나날이 발전하고 있다. 알려진 수천개의 웹서버 중에서 현재 가장 많이 사용되고 있는 웹서버는 아파치 웹서버이다. 아파치 웹서버는 원래 NCSA httpd 1.3을 모체로 발전하였으며 전세계 웹서버의 60%를 차지하는 매우 인기 있는 웹서버이다 [5]. 다양하고, 강력한 기능을 갖추었으며 매우 안정적인 데다 결정적으로 무료 소프트웨어라는 점에서 점점 입지를 넓혀가고 있는데, 아파치는 소스코드가 공개된 오픈 프로젝트(Open project)방식으로 진행 중이며, 사용자가 성능을 더 좋게 하기 위하여 자유로이 코드를 수정할 수 있다. 현재까지는 안정 버전 1.3.20까지 개발되었다. 동시에 이와는 별도로 새로운 기능과 더 보강된 기능을 갖춘 2.0버전도 개발하고 있다. 버전 2.0은 beta 버전으로 2.0.16까지 나왔으며, 2001년 8월 말 현재 2.0.18 alpha버전이 나왔고 2.0.24 alpha 버전을 test 중에 있다. 아파치 2.0은 아파치 1.3에 비해 많은 점이 변했는데 가장 큰 변화는 다중 스레드(multi-thread)를 지원하는 것이다. 본 논문에서는 이러한 변화가 전체 아파치 웹서버의 성능에 얼마나 많은 영향을 미쳤는지 실험을 통해서 아파치 1.3과 아파치 2.0을 비교 분석하였다.

본 논문의 구성은 다음과 같다. 제 2절에서는 아파치 웹서버에 대한 관련 연구를 논한다. 제 3절에서는 아파치 웹서버의 요청 사이클을 알아보고 아파치 1.3과 2.0에 대한 간단한

비교를 한다. 제 4절에서는 아파치 웹서버의 성능 평가를 위한 실험 환경 및 실험 방법을 소개하고, 제 5절에서는 실험에 대한 결과를 논한 후, 제 6절에서 결론을 맺는다.

2. 관련 연구

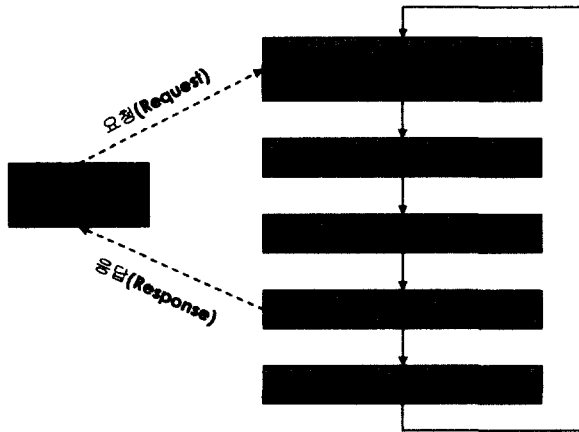
아파치 웹서버는 웹서버 자체에 대한 연구보다 아파치 웹서버를 이용한 연구에 많이 사용되었으며, 보통 실험 환경에서 사용하는 대부분의 웹서버가 아파치 웹서버이다. 아파치 웹서버에서 제공하는 가상 주소대신 단일 서버로 서비스하려는 연구 [2], 어플리케이션 단계에서의 분산 서버 [4], 파일 시스템 연구 [3] 등 웹 환경을 이용하는 많은 연구가 이러한 예가 된다. 아파치 웹서버에 대한 직접적인 연구는 아파치 웹서버의 성능을 측정하고 분석하여 성능을 개선하는 연구가 대부분이다 [1]. 아파치 1.3과 2.0에는 이러한 연구의 결과가 반영되었으나 아직 아파치 2.0에 대한 성능 분석 연구는 없었다. 따라서 본 논문에서의 아파치 2.0에 대한 성능 분석과 아파치 1.3과의 성능 비교는 매우 의미 있는 연구라고 할 수 있겠다.

3. 아파치 웹서버 버전 2.0

3.1 요청 수행 사이클(Request Processing Cycle)

최초로 아파치가 실행하면 main 프로세스는 설정 파일(configuration file)을 읽어서 각 설정 모듈을 초기화 한 후, 자식 스레드를 생성한다. 각각의 자식 스레드는 요청 수행 사이클에 들어가게 된다. 아파치 2.0 버전의 요청 수행은 크게 다섯 단계의 과정으로 나타낼 수 있다. 그 과정을 그림 1에 나타내었다.

첫 단계, 클라이언트와 컨넥션이 일어날 때까지 기다리다가



<그림 1. 아파치 2.0 요청 수행 사이클>

요청이 들어오면 읽는다(Wait and Read Request). 두 번째 단계, 요청 메시지로부터 헤더를 파싱해서(Header Parsing) 어떤 문서를 서비스 해 주어야 하는지 알아낸다. 세 번째 단계, 서비스할 문서를 메모리로 가져온다(Data Read). 네 번째 단계, 네트워크로 보낸다(Data Write). 즉, 클라이언트에게 응답 메시지를 보내는 것이다(Response). 마지막으로 해당 사이트에 대한 정보를 기록한다(Logging).

3.2 아파치 1.3과 아파치 2.0

아파치 1.3은 현재 사용중인 아파치 웹서버의 대부분을 차지하고 있다. 아파치 1.3은 다중 프로세스 방식으로 작동하며, 아파치 1.3.15 버전부터는 MS Windows 환경을 지원하기 시작하면서 다중 플랫폼(multi-platform) 지향적으로 발전하였다. 따라서 다양한 플랫폼에 맞도록 소스 코드를 작성했기 때문에 소스 코드의 관독성이 많이 떨어지게 되었다. 아파치 1.3과 아파치 2.0의 가장 큰 차이점을 표 1에 정리하였다.

아파치 1.3	아파치 2.0
다중 프로세스 방식	다중 프로세스/스레드 혼합 방식
캐쉬 지원 안함	파일 캐쉬
필터링 지원 안함	I/O 필터링
IPv6 지원 안함	IPv6 지원

<표 1. Apache 1.3과 Apache 2.0의 차이점>

아파치 2.0에서는 다중 프로세스 및 다중 스레드를 지원하며, 특히 Linux를 포함한 유닉스 시스템에서는 POSIX 스레드를 지원한다. 따라서 아파치 1.3보다 시스템의 부담을 상대적으로 덜 수 있으며, 더 좋은 확장성을 제공한다. 아파치 2.0에서는 모듈이 필요에 따라 입출력의 형태를 바꿀 수 있는 I/O 필터링을 제공한다. 이러한 기능은 특히 CGI나 Server Side Include를 사용할 때 유용하게 활용할 수 있다. 또, IPv6를 기본적으로 제공하여 IPv4와 IPv6를 바로 처리할 수 있다.

4. 실험 환경

4.1 SURGE(Scalable URL Reference Generator)

SURGE[6]는 웹서버의 성능 분석을 위한 툴이며, 요청하는 문서의 빈도, 크기, 시간 지역성(temporal locality), 공간 지역성(spatial locality) 등을 이용하여 웹서버측의 정적 문서들을 생성하며, 이러한 문서들을 GET 메소드로 요청하는 클라이언트를 자동 생성한다.

4.2 실험 장비

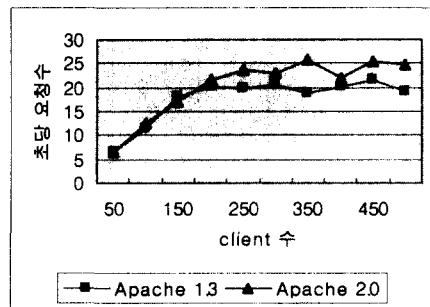
Apache web server의 성능을 측정하기 위하여 표 2와 같은 장비를 사용하였다.

	Server	Client
CPU	550MHz Pentium III	1GHz Pentium IV
RAM	512MB	256MB
OS	Linux 2.4.2	Linux 2.4.2
Network	2 × 100Mbps Ethernet	

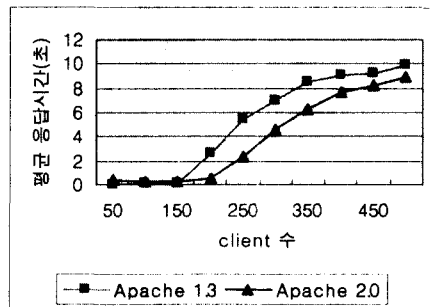
<표 2. 실험 장비>

4.3 실험 방법

실험은 두 가지 방법으로 하였는데, 첫 번째로 SURGE를 이용하여 클라이언트 수를 50부터 시작하여 50단위로 500까지 증가 시키며, 아파치 1.3과 2.0의 성능을 측정하였다. 두



<그림 2. 클라이언트 수에 따른 초당 요청 처리 개수>



<그림 3. 클라이언트 수에 따른 평균 응답시간>

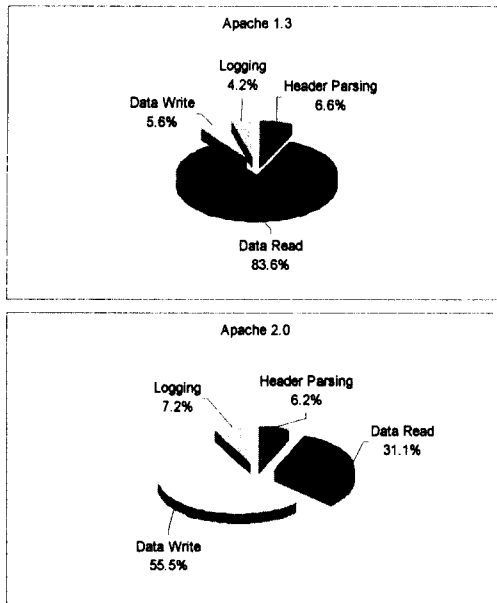
번째로는 일정 시간 동안 아파치 1.3과 2.0의 요청 처리 시간을 측정하여 요청 처리 비율을 구하였다.

5. 실험 결과

5.1 클라이언트 증가에 따른 아파치의 성능

매 실험마다 주어진 클라이언트는 5분 동안 문서를 요청 하였다. 그림 2에서 보는 것과 같이 서버의 처리율은 아파치 2.0이 1.3보다 더 좋은 성능을 보여준다. 특히 클라이언트 수가 200이상의 고부하일 때도 아파치 1.3보다 좋은 성능을 보여준다. 이것은 아파치 2.0이 다중 스레드를 지원하고, 파일 캐쉬를 지원하기 때문에 가능하다.

그림 3은 사용자의 평균 응답 시간이다. 클라이언트 수가 150일 때 까지는 아파치 2.0과 1.3이 큰 성능의 차이를 보이지 않지만, 클라이언트 수가 200을 넘기면서부터는 아파치 2.0이 더 좋은 성능을 보이고 있다. 그 이유는 사용자의 수에 비례해서 스레드와 프로세스의 수가 증가하는데 다중 프로세스를 사용하는 아파치 1.3과는 달리 다중 스레드를 사용하는 아파치 2.0의 스레드간의 문맥 전환 오버헤드가 프로세스간의 문맥 전환 오버헤드에 비해 대단히 작기 때문이다. 특히, 클라이언트 수가 200에서 350일 때 까지는 파일 캐쉬에 대한 영향도 작음하였다.



<그림 4. 아파치 1.3과 2.0의 요청 처리 비율>

5.2 일정 시간 동안의 요청 처리 비율

앞의 실험을 하면서 동시에 요청에 대한 처리 비율을 측정하여 평균을 구하였다. 그 결과는 그림 4에 있다. 아파치 1.3에서는 요청을 처리하는 데 있어서 데이터를 읽는 시간이 가

장 길었다. 아파치 1.3에서는 캐쉬를 지원하지 않기 때문에 위와 같은 결과가 나타났으며, 특이할 점은 로깅이 4.2%로 어느 정도의 비중을 차지한다는 것이다. 아파치 2.0의 경우는 1.3에 비해 데이터 읽기 시간이 상당히 줄었는데 기본적으로 파일 캐쉬를 하기 때문이며 따라서 상대적으로 데이터 쓰기 시간이 증가하였다. 아파치 1.3에 비해 로깅시간이 증가한 이유는 같은 시간 동안 아파치 2.0이 더 많은 요청을 처리 하였으며, 매 요청마다 로깅을 하기 때문에 상대적으로 아파치 1.3보다 로깅을 많이 하게 되었기 때문이다.

6. 결론

지금까지 아파치 1.3과 아파치 2.0에 대한 성능을 비교하였는데, 대체로 다중 스레드와 파일 캐쉬를 지원하는 아파치 2.0의 성능이 더 좋게 나타났다. 특히, 고부하 일수록, 요청이 많을수록 성능의 차이가 뚜렷하게 나타났다. 아파치 2.0은 1.3과 달리 기본적으로 파일 캐쉬를 하기 때문에 요청 처리 사이클에 있어서도 데이터 읽기 시간이 아파치 1.3에 비해 훨씬 적은 비중을 차지 하였다. 또한 다중 스레드를 지원하기 때문에 웹서버의 처리율이 최고 25% 향상되었으며 사용자 응답 시간도 평균 23% 향상되었다.

참고 문헌

- [1] Yiming Hu, Ashwini Nanda, and Qing Yang, "Measurement, Analysis and Performance Improvement of the Apache Web Server", Proceedings of 18th IEEE International Performance, Computing and Communications Conference, 1999
- [2] John Reumann, Ashish Mehra, Kang G. Shin, and Dilip Kandlur, "Virtual Services: A New Abstraction For Server Consolidation", Proceedings of USENIX Annual Technical Conference, 2000
- [3] Aaron Brown and David A. Patterson, "Towards Availability Benchmarks: A Case Study Of Software RAID Systems", Proceedings of USENIX Annual Technical Conference, 2000
- [4] Quanzhong Li, Bongki Moon, "Distributed Cooperative Apache Web Server", Proceedings of the 10th International World Wide Web Conference, 2001
- [5] The Apache Software Foundation, "The Apache HTTP server", <http://www.apache.org/>, 1999
- [6] Paul Barford and Mark Crovella, "Generating Representative Web Workloads for Network and Server Performance Evaluation", In Proceedings of SIGMETRICS '98, pages 151-160, 1998