2001년도 대한전기학회 하계학술대회 논문집 2001.7.18-20

가상 호스팅 서버에서 QoS 보장을 위한 기법에 관한 연구

류상우*, 이상문, 김학배
연세대학교 전기전자공학과

A Scheme Guaranteeing QoS in a Virtual Hosting Server

Sangwoo Ryou*, Sangmoon Lee and Hagbae Kim.
Dept. of Electrical and Electronics Engineering, Yonsei University

Abstract - 가상 멀티 호스팅 서버란 한 서버 안에 여러 도메인을 관리하면서 각각의 도메인에 해당되는 사이트를 운영할 수 있도록 하는 서버를 말한다. 보통 한 단일 서버 내에 200~300개의 사이트를 운영하게 된다. 하지만 외부로부터의 트래픽이 많아지면 각 사이트가 제대로 동작하지 않을 수 있다. 본 논문에서는 각 사이트들이 독립적으로 운영되면서 컨텐츠 전달의 본 목적을 충분히 발휘하면서 최적의 성능을 유지할 수 있도록 하는 QoS를 보장할 수 있는 구조를 연구한다. 또한 부하의 상태에 따라 피드백을 통한 컨텐츠의 질을 차등 적용함으로서 QoS를 보장하고자 한다.

1. 서 론

인터넷이 일반화되면서 홈페이지와 이와 관련된 웹서버가 급속히 보급되었다. 각 수요 예측 기관이 예측한 것보다 훨씬 빠른 속도로 발전하고 있는 실정이며 이제 사업을 하려면 최소한 그 회사 홈페이지 정도는 마련해야하는 실정이다. 회사의 홈페이지를 인터넷에 올리기 위해서는 서버가 필요하지만 용량이 얼마 되지 않는 홈페이지를 운영하기 위해 고액의 서버를 구입하기에는 부담이 될 수 밖에 없다. 이 문제를 해결하기 위한 솔루션으로 등장한 것이 멀티호스팅 서비스이다.[1] 멀티 호스팅이란 서버에 각 공간을 할당하고 URL 요청이 들어왔을 때 웹서버에서 각각의 공간으로 연결시켜주는 방식을 말한다. 클라이언트 입장에서는 마치 그 홈페이지가 독립된 서버에서 공급이 되고 있는 듯하게 보이게 될 것이다. 이 방식은 한 서버에 하드 용량에 따라 다르지만 200~300개의 사이트를 수용할 수 있게 되어 자원을 효율적으로 쓸 수 있는 장점이 있으면 방식에 따라 차이가 있지만 IP를 하나만 사용하여도 많은 수의 사이트를 운영할 수 있어서 IP를 절약할 수 있는 장점도 있다.[2]

이런 장점이 있는 반면 서버에 상주하는 호스트중에 한 사이트에라도 접속량이 많아지면 즉 트래픽이 급격히 늘어나게 되면 그 사이트와 상관없는 호스팅 서버의 다른 모든 사이트가 한꺼번에 속도가 느려질 수 있는 약점이 있다. 자원을 공유하기 때문에 자원 배분이 공평하지 못하면 다 같은 문제를 공유할 수 밖에 없는 것이다. 본 논문에서는 호스팅의 기법과 이런 멀티 호스팅 서비스에서 발생할 수 있는 문제에 대해서 알아보고 고급의 서비스 제공할 수 있는 방법에 대해 논의하고자 한다.

2. 본 론

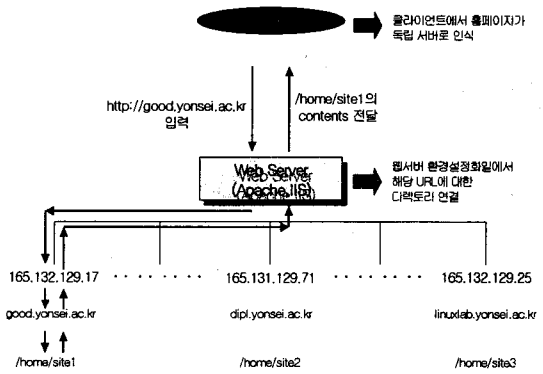
2.1. 가상 호스팅 서버

가상 호스팅 서버라 함은 하나의 컴퓨터에 하나의 도메인 이름을 포함하고 있는 웹 서비스를 여러 개 운영하는 방법이라 할 수 있다. 현재 가상호스트로 이용되어지고 있는 방법은 두 가지가 있는데 IP 주소를 기반으로 한 "IP address Based Virtual Hosting" 기법과 호스트 이름을 기반으로 한 "Name Based Virtual Hosting" 기법이 있다.[3] 두 기법의 차이는 전자는 말 그대로 IP 주소를 기반으로 한 서비스이다. 이 방식은 서버에 상주하는 모든 사이트가 다 각자의 고유의 IP를 소유하고 웹서는 그 IP를 보내주는 방식이다. 즉 서버에 n개의 호스트가 상주한다면 그 서버에는 n개의 IP를 가지고 있어야 서비스가 가능하다.

규모가 작은 기업이나 개인에게 할당되는 IP 는 제한적이며 몇 백 개의 웹 호스팅을 운영하려면 실제로 아주 많은 IP 주소들이 필요하게 된다. 또한 IP 기반의 가상호스트 방법은 동일 머신 안의 모든 가상호스트에 고정된 IP 를 하나씩 할당하여야 하기 때문에 관리상의 문제와 동시에 여러 제약이 따른다. 바로 이럴 때 사용할 수 있는 방법이 후자의 Name Based Virtual Hosting의 가상호스트 기법이다.

2.1.1 IP Address Based Virtual Hosting

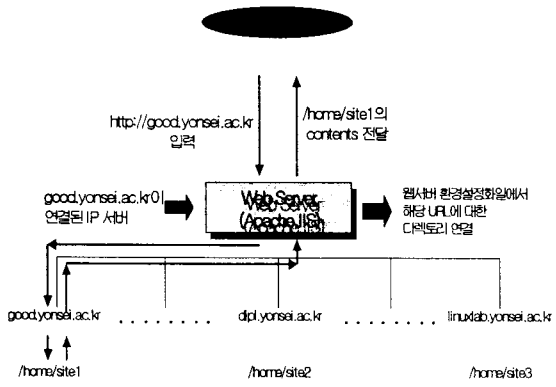
이것은 서버에 있는 하나 또는 그 이상의 LAN 카드에 여러 IP를 등록하고 웹서버에서 각 IP에 따른 디렉토리를 지정하여 사용하는 방법이다. 즉 클라이언트가 어느 사이트의 주소 aaa.bbb.ccc.ddd를 입력하여 가상 서버에 요청이 들어오면 그 서버의 웹서버는 그 IP에 해당하는 디렉토리를 클라이언트 브라우저로 보내주는 것이다. 각 사이트가 각각의 IP를 갖게 되므로 보다 안정적인 서비스를 할 수 있고 별도의 DNS 서버 세팅이 필요 없어 간단하게 서버를 구현할 수 있다. IP를 독립적으로 사용함으로 성능측면에서 효율적인 서비스 방식이다.



2.1.2 Name Based Virtual Hosting

이 방식은 여러 도메인을 하나의 IP로 할당시켜 놓고 외부에서 요청이 들어왔을 때 웹서버에서 해석하여 각 사이트를 연결시키는 방식이다. 예에서 볼 수 있듯이 서버의 각 호스팅 서버의 도메인들은 모두 한 IP로 등록이 되어있어야 한다. 그러면 일단 어떤 사이트를 요청하면 그 호스트가 있는 서버에 리퀘스트가 도착하고 서버에서 연결시키는 것이다. IP를 절약하면서 서버를 운영할 수 있으나 각각의 도메인이 대표 IP를 가르키도록 DNS 서

버에서 세팅을 별도로 해주어야 하는 제약이 따르게 된다. 비용 측면에서 효율적인 서비스 방식이다.



2.1.3. 호스팅 서버와 트래픽

하지만 어느 방식이나 가진 공통적인 문제가 있는데 외부의 급격한 리퀘스트의 증가이다. 200~300개의 사이트중 어느 사이트에 대한 홈페이지 요청이 많아지면 서버의 부하가 증가하고 요청 빈도가 없거나 낮은 사이트까지 서비스 속도가 떨어지게 되는 문제점이 있다. 1개의 사이트 때문에 나머지 많은 수의 사이트의 속도가 지연이 발생하게 된다.

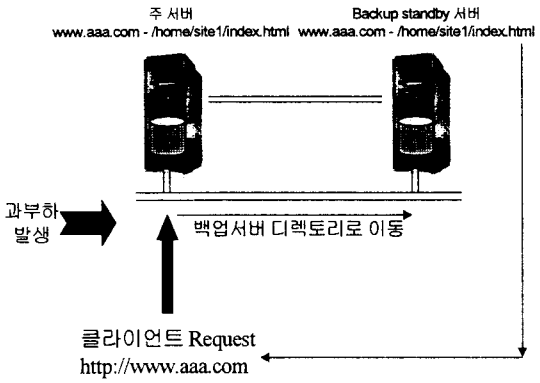
이런 부정적 현상을 막기 위해서 호스팅 업체들은 요청되는 데이터를 일별 또는 월별로 총량으로 제한하여 서버의 상태를 유지하는 다소 수동적인 방법을 쓴다. 호스팅 계약시 정한 리퀘스트와 부하량를 초과하면 접속을 차단하고 더 이상 서비스를 제공하지 않는 방법으로 대처하고 있다.

서비스를 유지하면서 서버의 상태를 최적으로 유지할 수 있는 방법을 생각해보자. 이런 문제점이 발생하는 것을 막기 위해서 생각할 수 있는 것이 리퀘스트가 높은 사이트의 부하를 줄여주는 방법이다. 부하를 줄일 수 있는 방법으로 여러 가지를 생각할 수 있으나 이 논문에서는 다음 2가지로 고려해본다.

2.2. 사이트 분산

리퀘스트가 높은 사이트의 여벌(replica)을 미리 만들어 놓는 방법이다. 즉 stand-by 디렉토리를 만들어 놓고 웹서버에서 사이트에 들어오는 트래픽을 측정하고 있다가 어느 정도 이상의 요청이 들어오면 그 다음부터는 복제된 사이트로 요청을 분산을 시키는 방법이다. 보통 호스팅 서비는 만약을 대비해서 언제나 본 서버 대신에 백업용 서버를 운영하는 것이 일반적이다. 하지만 이 서버는 주기적인 데이터를 백업하기 위해서 있을 뿐 그 외에서는 stand-by 상태를 유지한다. 주 서버의 특정 사이트에 부하가 많이 걸렸을 때 그 사이트의 IP를 바꾸어 백업 서버로 부하를 분산시킨다. 백업 서버는 CPU 리소스를 다른 사이트와 공유하지 않고 부하가 걸린 사이트만을 위해 돌아가기 때문에 부하가 많이 줄어들 수 있게 된다.

부하가 걸렸을 때도 부하를 모니터링 하고 일정 리퀘스트 아래로 떨어지면 본래대로 주 서버가 전담하게 되며 백업 서버는 다시 stnad-by 상태로 돌아가게 된다



이 방법은 비교적 부하를 빨리 분산시킬 수 있으며 컨텐츠의 질(quality)도 보장할 수 있으나 또 다른 시스템을 써야하므로 비용이 발생한다. 이 기법은 기존의 리눅스 가상 서버에서 사용하는 가상 IP와는 다른 방법이다[4]. 가상 IP는 부하가 걸린 서버를 완전히 이전하고 주 서버를 사용하지 않는데 반해 제안된 방법은 주 서버에서 부하가 걸린 디렉토리만 백업서버에서 기능을 담당하는 것으로 더 효율적이라 할 수 있다. 이 기법을 적용하여 이용하는 사이트는 부가적인 이용료를 받아야 할 것이다.

이 방식을 리미터(limiter)라 정의한다. 리미터에는 방식에 따라 경성리미터와 연성 리미터로 나눈다

2.2.1 연성 리미터

연성 리미터는 부하량을 측정하여 트래픽이 발생했을 때 부하량이 많이 걸리는 사이트의 컨텐츠중 용량이 크고 CPU 점유율이 높은 파일을 stand-by 서버에서 클라이언트에 제공하는 방식이다.

```
<html>
. . .
. . . . .
<a> <img src="/picture/mypic.jpg" </a>
. . . . .
. . . . .
```

```
<html>
. . .
. . . . .
<a> <img src="http://백업서버URL/picture/mypic.jpg"
</a>
. . . . .
. . . . .
```

표 1 소스 html 과 부하 발생시 제공되는 html

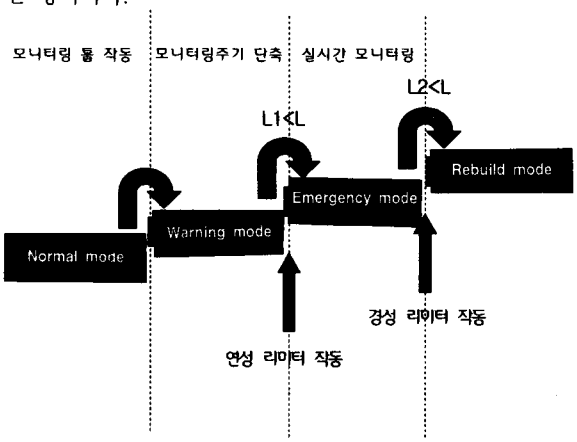
2.2.2 경성 리미터

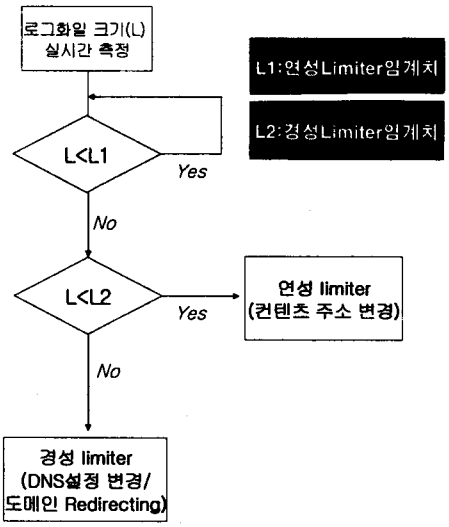
제한량이 훨씬 초과했을 때 DNS 서버에서 리다이렉팅을 하는 방법이다. 트랙픽이 많이 걸려서 서비스가 어렵다고 판단이 되면 DNS 서버에서 그 사이트에 요청이 들어올 때 예비 서버의 도메인으로 연결시켜 부하를 분산한다. 경성 리미터는 연성 리미터의 제한 정도를 넘어 트래픽이 폭주할 때 쓴다

2.2.3 모니터링 및 모드 변환

모니터링 방법으로는 서버의 로그 파일을 분석하는 방법을 사용한다. 로그 파일의 특정 문자열 즉 각 사이트에 해당하는 IP 문자열을 주기적으로 체크한다. 시간에 대해 미분을 하여 주기 사이에 값이 급격히 증가하면 일

단 그 사이트를 "주의 모드(warning mode)"로 세팅한다. 주의 모드로 세팅된 사이트는 체크 주기를 줄이며 로그 파일을 모니터링한다. 로그 파일 크기가 소프트 리미터 임계치인 L1를 넘는지를 체크하다가 만약 L1을 넘으면 소프트 리미터 스크립트를 실행시킨다. 이 스크립트는 준비된 다른 html 문서를 호출하는 것이다. 새로 호출된 html 문서는 텍스트 내용은 같지만 제공하는 그림 파일등 점유율이 높은 리소스의 주소가 같은 서버가 아닌 백업 Stand-by 서버 주소로 세팅이 된 html 문서이다. 즉 텍스트는 주 서버에서 제공하고 그림 파일은 백업 서버에서 제공하는 방법이다. 소프트 리미터가 작동하게 되면 비상모드(emergency mode)에 들어가게 된다. 이 모드는 로그화일을 가장 짧은 주기로 체크하면서 모니터링한다. 만약 로그 파일의 크기가 하드 리미터 임계치인 L2에 도달하면 스크립트는 DNS 서버의 설정을 미리 예비해 둔 설정 파일로 바꾸는 스크립트를 호출한다. 새로운 DNS 서버 설정 파일은 그 IP에 대한 도메인 리다이렉팅을 수행하게 된다. 그 사이트 전체가 백업 서버에서 서비스하게 되는 것이다. 역시 가상 서버와 달리 그 사이트가 상주하는 디렉토리만 백업 서버에서 작동하는 방식이다.





2.2.4 해킹 방지를 통한 QoS 보장

해킹 방법중 일반화된 해킹이라면 DOS(Denial Of Service)로 무수한 http 요청을 보내어 그 서버가 다운되도록 하는 방법이다. 이 방법은 서버의 루트 권한을 뺏는 등 고급의 기술 필요 없이 간단하게 서비스를 중지시키는 기법이다. 위에서 제안한 리미터를 이용하면 이런 해킹 기법은 초기에 막을 수 있을 것이다. 즉 로그 파일을 분석하여 들어오는 IP를 분석, 만약 특정 IP에서 시간당 집중적으로 들어오는 패턴이 발생하거나 비슷한 IP에서 과도한 접속이 이루어진다면 그 IP를 차단하여 DOS 공격을 미연에 방지할 수 있을 것이다. 해킹에 의한 서비스 중단을 막고 정상적인 서비스를 제공하는 방법이며 위의 리미터 기능을 하면서 부수적으로 같이 제공할 수 있기 때문에 QoS 측면에서 효율적이라고 할 수 있다.

2.3 컨텐츠 변환

만약 백업의 서버를 둘 수 없고 한 대의 서버로 운영을 할 때를 가정해보자. 일단 들어오는 리퀘스트를 수용하면서 사이트를 운영해야 한다. 여기서 제안하는 것은 컨텐츠의 등급화하는 것이다. 즉 같은 내용의 사이트를 운영하면서 각 사이트에 들어가는 컨텐츠의 등급을 달리해서 제공하는 것이다. 홈페이지를 구성은 크게 보면 글이 들어가는 텍스트와 그림 파일로 이루어 진다. 보통 사이트라면 텍스트는 아무리 들어가도 3KByte를 넘지 않지만 그림 파일은 크기가 작은 아이콘이라도 1KByte에서 큰 것은 수백 KByte의 가지게 된다. 이런 파일들은 클라이언트의 요청이 들어오면 서버의 CPU에서 읽기를 명령하고 이 파일들은 일단 서버의 메모리에 읽혀지게 된다. 파일의 크기가 크면 CPU의 부하량과 메모리 점유율이 올라가게 되는 것이다. 그렇다면 이런 크기가 큰 파일들을 줄일 수 있다면 CPU나 메모리의 점유 시간을 줄일 수 있고 시간이 준다는 것은 줄어드는 타임만큼 리퀘스트를 받을 수 있게 된다.[5]

이런 컨텐츠를 여러 크기로 만들고 정상적인 상태일 때는 가장 높은 등급 즉 원래 파일을 클라이언트에 제공하고 들어오는 부하의 양에 따라 상대적으로 크기가 작은 파일을 제공함으로서 서비스 상태를 유지하는 것이다. 이런 방법은 서버를 하나만 두고 서비스가 가능하므로 저렴하게 시스템을 설계할 수 있다.

3. 결 론

호스팅 서버에서 걸리는 부하를 분산하고 컨텐츠를 변환시키는 방법에 대해 알아보았다. 모니터링 방법으로는 로그 파일의 크기를 모니터링 하는 방법을 이용하였지만 정확한 모니터링을 하려면 Traffic Analyzer 같은 전문 장비를 이용해야 할 것이다. 하지만 제안된 방법은 비교적 간단히 꾸밀 수 있으며 플랫폼과 독립적으로 용이하게 구현할 수 있다는 장점이 있다.

[참 고 문 헌]

[1] Mohammed Kabir(홍승필 역), "레드햇 리눅스 6 서버", 파워북 pp385-386, 2000

[2] 박성수, "아파치 웹서버를 100% 활용한 호스팅사업 실무지침서" 한림미디어 pp 10-11, 2000

[3] http://www.apache.kr.net/documents/name-virtual.html

[4] 박지현, "고가용성 리눅스 클러스터링 가상서버와 CODA 고장포용네트워크 분산 파일 시스템 활용에 관한 연구"", 연세대학교 석사학위논문, pp5-6, 2001

[5] Tarek F. abdelzher, "Web server QoS management by adaptive content delivery ",Quality of Service, 1999. IWQoS '99. 1999 Seventh International Workshop on , 1999 pp 216 -225