웹 캐시 통합 관리 시스템 설계 및 구현

[1]박진원[0] [1]권대현 [1]김명균 [2]홍윤환
[1]울산대학교 컴퓨터.정보통신공학과
[2]닥터 소프트
{[1]zwsonic@cnlab.ulsan.ac.kr[0], [1]dhkwan@mail.ulsan.ac.kr, [1]mkkim@mail.ulsan.ac.kr [2]hong@drsoft.co.kr}

Design and Implementation of Web-Caches Management System

[1]Zin-Won Park[0] [1]Dae-Hyun Kwan [1]Myung-Kyun Kim [2]Yoon-Hwan Hong Youn-Hwan
[1]Computer Information Communication Dept. University of Ulsan
[2]Dr. Soft

요 약

인터넷 사용자의 수가 폭발적으로 늘어나고 웹 컨텐츠의 양이 많아지면서 웹캐시의 수요는 날로 높아져 가고 있다. 동시 접속자의 수가 수천에서 수만에 이르는 대형 웹사이트의 경우, 다수의 웹캐시를 두어 웹 서버의 부하를 줄이기도 한다. 이러한 웹서비스 환경에서 여러 대의 웹캐시를 효율적으로 관리하고 모니터링 할 수 있는 관리 시스템이 요구되고 있다. 본 논문에서는 다수의 웹캐시를 통합 관리하고 모니터링 할 수 있는 웹 기반 통합 관리 시스템을 설계하고 구현해 보았다.

1. 서 론

인터넷 사용자의 수가 많아지면서 대부분의 정보들이 인터넷을 통하여 검색되어지고 있다. 비단 정보뿐만 아니라 각종 서비스 역시 인터넷을 통한 웹 기반으로 이루어지고 있어 대부분의 인터넷 사용자들은 웹을 통하여 정보를 주고받고 있는 실정이다. 이렇게 웹 기반으로 이동하고 있는 정보의 양이 많아짐으로 해서 웹을 사용하는 사용자의 입장에서나 웹을 통한 정보 제공자의 입장에서 빠른 웹 서비스 이용과 안정적인 서비스 제공을 위해 웹캐시 시스템을 도입하고 있다[1].

많은 이용자를 가진 대학이나 기업, 관공서뿐만 아니라 동시 접속자의 수가 수만에 이르는 대형 웹사이트에서는 여러 대의 웹캐시가 필요하게 된다. 그리고 복수의 웹캐시 시스템은 관리와 모니터링을 하기 위해 각 시스템의 관리 장비를 따로 두어야 했다.

본 논문에서는 이런 복수의 웹캐시 환경에서 하나의 관리 툴을 이용하여 여러 대의 웹캐시 시스템을 통합적으로 관리하고 모니터링 할 수 있는 웹 기반의 통합 관리 시스템을 설계하고 구현해 보았다.

본 논문의 순서로는 2장에서 웹캐시 시스템의 구성과 복수의 웹캐시 시스템 구성 환경을 살펴 본 다음, 3장에서는 본 논문의 구현에 사용된 캐시 시스템인 Squid 캐시에 대해서 알아보고, 4장에서 본 논문의 주제인 웹 기반 통합 관리 시스템의 구성 및 구현 방법을 설명한 후 5장에서는 결론을 맺고 및 향후 과제에 대해서 알아본다.

2. 웹 캐시 시스템

캐시의 역할은 정보를 일정한 시간 동안 저장소에 저장해 두었다가 사용자의 요청이 있을 경우 서비스 제공자 대신에 정보를 서비스 해 주는 것이다. 현재 캐시 시스템은 종류에 따라 HTTP 뿐만 아니라 FTP, Gopher 등 다양한 정보를 캐싱 해 주고 있다. 그러나 대부분의 캐시들이 HTTP 캐싱을 위해 사용되고 있는 점을 고려하여 본 고에서는 웹캐시 시스템을 중점으로 다룬다. 하지만 HTTP 캐시뿐만 아니라 다른 캐시 서버 역시 구성이나 작동 방식은 웹캐시와 유사하다.

웹캐시 시스템은 설치 방식과 목적에 따라 포워드 캐시(forward cache)와 리버스 캐시(reverse cache)로 나눌 수 있다.

2.1 포워드 캐시(Forward Cache) 구성

포워드 캐시는 <그림 1>(a)와 같이 클라이언트 측에 위치한다. 이 캐시의 역할은 클라이언트들의 요청을 받으면 자신의 저장 장치를 검색하여 해당 정보가 없을 경우 웹서버에게 요청을 보내고 웹 서버가 보내온 정보를 저장하고 클라이언트에게 정보를 제공해 준다. 여기서 한번 저장된 정보는 일정 시간동안 계속 캐시 서버에 남아 있게 되는데 정보가 저장되어 있는 시간은 캐시 서버의 정책으로 정하거나 웹 페이지 작성자가 명시적으로 지정할 수 있다[2].

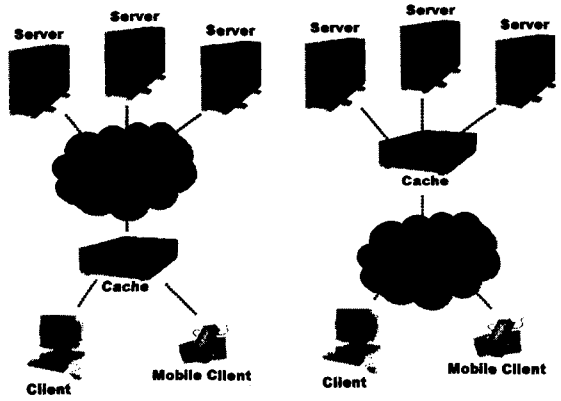
포워드 캐시의 역할은 클라이언트에게 제공했던 정보들을 캐시 서버가 저장해 두었다가 다음 번 클라이언트가 요청을 해 오면 캐시된 정보를 서비스 해 주는 것이다. 이 방식의 캐시 서버는 불특정 다수의 웹서버 정보를 저장하고 있게 되며 특정 클라이언트의 요청만을 받는다는 특징이 있다. 또한 정보가 인터넷 구간을 거쳐오는 동안의 지연시간을 없애줌으로써 웹 페이지의 로딩(loading) 속도가 빨라지고 클라이언트 측 백본 네트웍의 트래픽(backbone traffic)을 줄일 수 있다는 장점이 있다.

2.2 리버스 캐시(Reverse Cache) 구성

리버스 캐시는 <그림 1>(b)와 같이 웹서버 측에 위치한다. 이 방식은 포워드 캐시와 같이 클라이언트의 요청을 캐시가 받아서 웹 서버를 대신하여 클라이언트에게 서비

스한다. 저장시간은 포워드 캐시와 같은 방법으로 결정된다.

리버스 캐시는 불특정 다수의 클라이언트의 요청을 받게 되며 특정 서버의 정보만을 캐싱하는 특징이 있다. 이 방식은 수많은 클라이언트의 요청에 웹 서버가 다운되거나 서비스 속도가 느려지는 것을 막아 주고 서버의 부하를 줄이는데 목적이 있다[3]. 뿐만 아니라 클라이언트가 서버에 직접적으로 접속하지 않기 때문에 방화벽의 역할도 한다.



(a)포워드 캐시 (b)리버스 캐시

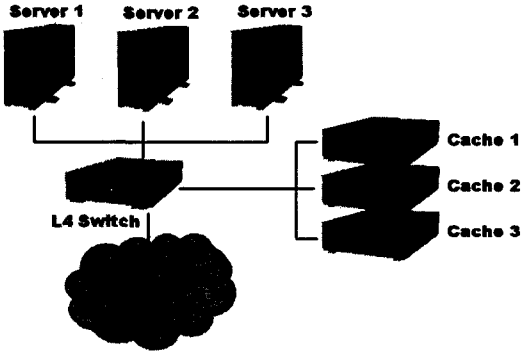
<그림 1> 캐시 서버 구성

2.3 복수의 캐시 서버 구성

동시 접속자 수가 많은 웹사이트의 경우 웹서버를 여러 대 두기도 한다. 이렇게 하는 이유는 한 서버에 많은 접속자가 동시에 접속했을 경우 웹 서버가 다운되거나 혹은 서비스를 하는 속도가 매우 느려지기 때문이다. 하지만 여러 대의 웹 서버를 두어도 클라이언트의 수가 매우 많을 경우 몇 대의 웹서버로는 부족할 수 있다. 하지만 데이터 베이스 서버와의 연계 및 컨텐츠 관리에 어려움이 따르기 때문에 웹서버의 수를 계속 늘일 수는 없다. 따라서 이런 경우 여러 대의 캐시 서버를 두어 부하는 분산시킨다. 캐시 서버 역시 한 대의 캐시 서버에 부하가 집중되는 것을 막기 위해 여러 대의 캐시 서버를 두어야 한다.

웹서버와 캐시 서버 사이에는 스위치를 두어 로드 밸런싱(load balancing)을 한다. 뿐만 아니라 Layer-4 스위치를 사용할 경우 트랜스페어런트 캐시(transparent cache)로 동작하여 클라이언트에게는 캐시 서버가 존재하지 않는 것처럼 구성 할 수도 있다.

각 캐시 서버는 ICP(Internet Cache Protocol)를 이용하여 서로간의 정보를 교환한다[4]. 예를 들어, 클라이언트의 요청을 cache1이 받았을 때, cache1에 클라이언트가 요청하는 정보가 없다면 cache1은 다른 캐시 서버들에게 ICP request 메시지를 보내게 되고 해당 정보를 가진 캐시 서버가 있을 경우 그 캐시 서버로부터 정보를 얻어와 자신의 저장소에 저장하고 클라이언트에게 정보를 제공한다. 만약 정보를 가진 캐시 서버가 없을 경우 캐시 서버는 웹 서버에게 요청을 보내고 정보를 가져와야 한다.



<그림 2> 복수의 캐시 서버 구성

3. Squid 캐시

Squid 캐시는 인터넷에 공개된 캐시 프로그램으로 유닉스 기반의 시스템에서 동작한다. 또한 오픈 소스 프로젝트로 개발되어 소스 코드가 공개되어 있어 많은 캐시 시스템이 Squid 캐시를 기반으로 개발되어 상품화되어 있다[5].

Squid 캐시는 트랜스페어런트 캐싱, SSL 캐싱, DNS 캐싱, HTTP, FTP, WCCP, ICP, SNMP 등을 지원한다. 이렇게 다양한 프로토콜과 관리 기능을 제공함으로 해서 캐시 시스템을 개발하는 많은 개발자가 Squid 캐시를 참고하여 캐시 시스템을 개발하였다.

Squid 캐시는 설정 파일인 squid.conf 파일을 이용하여 acl(access control list)기능을 제공하는데 이 acl을 이용하여 캐시 시스템으로의 접근 제어 및 캐시 시스템의 사용자를 제한을 할 수 있다.

Squid 캐시는 웹 기반의 모니터링 툴(cachemgr.cgi)을 제공한다. 이 툴은 CGI기반으로 동작하며 Squid 캐시와 소켓 통신을 이용하여 메시지를 주고받는다. cachemgr.cgi는 cache_object라는 요청 메시지를 전송하는데 cache_object 뒤에 따르는 요청 항목에 따라 Squid 캐시는 자신이 가지고 있는 정보를 cachemgr.cgi로 전송한다.

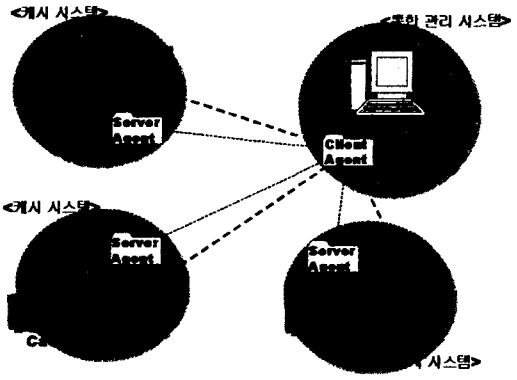
Squid 캐시가 제공하는 cachemgr.cgi는 텍스트로 된 정보만 제공하고 있으며 각 항목에 캐시 프로토콜 전문 용어를 별도의 설명 없이 그대로 사용하여 이해하는데 있어서 많은 어려움이 있다. 또한, 표나 그림이 제공되지 않아 독립적인 관리 툴로 사용하기에는 부적합하다는 한 계점을 가지고 있다.

4. Squid 기반 웹 캐시 통합 관리 시스템

본 논문에서는 Squid 캐시를 기반으로 복수의 캐시 시스템을 사용하는 웹 서비스 환경에서 여러 대의 캐시 시스템을 통합된 환경에서 관리하고 모니터링 할 수 있는 관리 시스템을 개발하였다. 또한 별도의 장비 없이 웹 브라우저만으로 관리가 가능하도록 웹 기반의 인터페이스를 제공한다.

<그림 3>에서는 통합 관리 시스템의 구성을 보여주고 있다.

관리자는 관리 시스템에 접속하여 캐시 서버의 상태를 모니터링 하거나 설정을 변경하기도 하고 캐시 서버를 재시작 시킬 수도 있다.



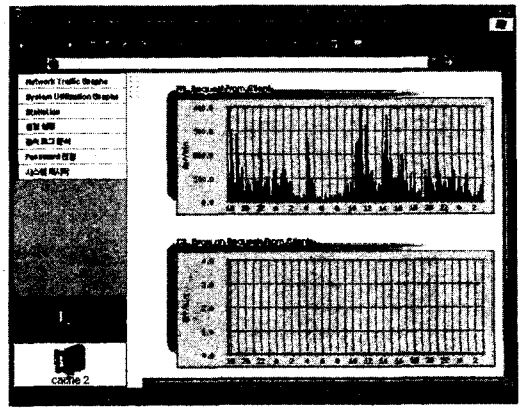
<그림 3> Squid 기반 웹 캐시 통합 관리 시스템 구조

관리 시스템에 존재하는 Client Agent는 캐시 서버와의 직접적인 통신을 담당한다(굵은 점선). 이 에이전트는 캐시 서버가 클라이언트로부터 요청 받은 회수, 요청에 대한 에러 수, 웹 서버로부터 가져온 데이터의 양(bytes), 웹 서버로 전송한 데이터의 양(bytes), CPU 사용률, 메모리 사용률, 시스템 페이지 폴트가 발생한 회수 정보를 캐시로부터 1분마다 수집하여 데이터 베이스에 저장한다. 이 정보는 관리 프로그램이 만들어내는 그래프의 자료가 된다<그림 4>. 또한 관리자가 원하는 실시간의 정보를 캐시에게 요청하고 캐시 서버가 보내오는 응답 메시지를 관리 프로그램으로 전달한다. 여기서 요청될 수 있는 실시간 정보로는 메모리 점유 및 이용 상태, 런 타임 정보, 파일 오픈 정보, DNS 상태, IP 캐시 상태, 현재 처리중인 요청 리스트, 클라이언트의 리스트 등이 있다. 관리 프로그램은 에이전트가 전달해 주는 메시지를 분석하여 표를 만들어 관리자에게 보여준다. Client Agent가 캐시 서버와 주고받는 메시지는 Squid 캐시가 제공하는 cachemgr.cgi이 사용하는 메시지 형식을 그대로 사용하도록 하였다. 또한 캐시 서버의 acl 설정에서 관리 시스템의 접근만을 허용하게 하여 보안 측면도 고려하였다.

각 캐시 시스템에는 Server Agent라는 에이전트를 두고 있다. 이 에이전트는 Client Agent와 통신을 하면서 (가는 점선) Squid 캐시의 설정 파일(squid.conf)을 읽거나 수정하고 로그 파일(access. log)을 읽어서 Client Agent로 전송하는 것이다. 이렇게 해서 관리 프로그램은 웹 기반으로 Squid의 설정 파일을 수정할 수 있고 로그 파일을 관리 시스템에서 분석할 수 있게 된다. 관리 프로그램을 캐시 시스템에 두지 않는 이유는 관리 기능으로 인한 캐시 시스템의 부하를 최소화하기 위한 것이다.

관리자는 <그림 4>의 좌측 하단에 있는 캐시 서버 리스트에서 설정을 변경하거나 모니터링 하고자 하는 캐시 서버를 선택할 수 있게 하였다.

Squid 기반 웹 캐시 통합 관리 시스템 구현에 사용된 운영체제는 리눅스 커널 2.4이고 관리 프로그램은 PHP4, 에이전트는 C언어를 이용하여 구현했으며 데이터 베이스는 MySQL을 이용하였다.



<그림 4> Squid 기반 웹 캐시 통합 관리 시스템 구현

5. 결론 및 향후 과제

본 논문에서 제시한 Squid 기반 웹 캐시 통합 관리 시스템을 이용하여 복수의 웹 캐시 시스템을 하나의 통합된 환경으로 관리하고 모니터링 할 수 있게 되었다. 또한 표 형식과 그래프화 시킨 이미지를 제공하여 캐시 시스템의 사용률과 동작 상태를 모니터링하고 웹 환경에서 설정파일을 수정하여 캐시 시스템에 적용할 수 있도록 하였다. 캐시 시스템의 관리 기능을 캐시 시스템과 독립적인 환경에 구축함으로 해서 캐시 시스템의 부하를 최소화로 줄여 성능 저하를 막고 이해하기 쉽고 관리하기 쉬운 환경을 제공할 수 있었다.

향후 계획으로는 그래프가 웹 브라우저에 캐싱되어 실시간으로 변하지 않는 문제를 해결하기 위해 자바 애플릿을 이용하여 그래프를 구현하고 웹 서버의 설정 및 로그 분석 기능을 추가하여 통합 웹사이트 관리 툴로 발전시켜 나갈 것이다.

6. 참고 문헌

[1] Caching Tutorial for Web Authors and Webmasters. online documents "http://www.mnot.net /cache_docs/"

[2] Ker Kob, Sam H.Nob, Sang Lyul Min. "Efficient Replacement of Nonuniform Of Objects in Web Cache" IEEE Computer, Volume: 35 Issue: 6, June. 2002

[3] Hyokyung Bahn, Kern Koh, Noh, S.H., Lyul, S.M. "Site-Based Approach in HTTP Proxy design" *Proc. Int'l Conf. Parallel Processing, Workshop on Internet*, IEEE Computer Soc. Press, Los Alamitos, Calif. 1999

[4] Duane Wessels and K. Claffy. "ICP and the Squid Web Cache" IEEE JOURNAL. 1998

[5] Squid-based Products "http://www.squid-cache .org/products.html"