
웹서버의 부하균형을 위한 트래픽상황분석 모니터링 시스템

Traffic Analysis Monitoring System for Web Server Load Balancing

김석수, 최이정, 이은석
한남대학교 멀티미디어학과

Seok-Soo Kim(sskim@hannam.ac.kr), E-Jung Chol(ejchoi@hannam.ac.kr)
Eun-Seok Lee(eslee@hannam.ac.kr)

요약

웹 서버 클러스터 환경에서 다중의 서버들이 원활하게 상호연동을 수행하여 클라이언트의 요청을 처리하기 위해서는 TCP 정보나 요청되는 대상의 콘텐츠(content)정보를 이용하여 라우팅을 수행하는 라우터의 구현이 필수적이다.

본 논문의 구현 패키지 S/W에 의해 데이터 생성기, 가상서버, 서버1,2,3 에서 발생하는 패킷의 량을 측정하는 것으로 서버1,2,3에게 트래픽을 분산시키는지 파악할 수 있었으며, 실험결과와 같이 들어오는 데이터의 크기가 많이 차이나지 않는 이상 라운드 로빈 알고리즘은 확실한 형태의 트래픽 분산을 가능하게 해주었다. 그리고 일부분에서 오차가 심한경우도 있지만 횡수를 거듭하고 테스트가 장기화 될수록 오차는 줄어들었다.

■ 중심어 : | 웹서버 | 부하균형 | 트래픽 |

Abstract

In order to handle client's requests while multiple servers seamlessly work in the web server cluster environment, it is vital to implement a router that execute a routing using TCP information and requested target content.

The implemented package software measured packet volume that was generated from data generator, virtual server, and server 1, 2, 3, and could find out traffic distribution toward Server 1, 2, 3. As the result of the study shows, Round Robin Algorithm ensured definite traffic distribution, unless incoming data loads differ much. Although error levels were high in some partial cases, they were eventually alleviated by repeated tests for a longer time.

■ Keyword : | Web server | Load balancing | Traffic |

1. 서론

그동안 네트워크에서 보안 기능은 많은 어려움이 있었

으며, 이는 암호화와 인증과정을 거치는 과정에서 전송 지연을 초래하였고, 현재 보안세션을 제공하는 서비스 표준인 TLS(Transport Layer Security, Netscape사

* 본 연구는 산업자원부 지역협력연구사업(R12-2003-004-03003-0)지원으로 수행되었음.

접수번호 : #040916-001

접수일자 : 2004년 09월 16일

심사완료일 : 2004년 11월 03일

교신저자 : 김석수, e-mail : sskim@hannam.ac.kr

의 SSL(Secure Socket Layer)의 개명)은 고속의 전송을 보장 못하는 문제점을 가지고 있다.

그래서 TLS 프로토콜을 사용할 때 레코드 프로토콜이 보안세션을 생성하기 위하여 암호화 및 인증작업을 위한 보안키가 쌍방향통신채간에 미리 설정되어 있어야 하는데, 이를 위하여 핸드셰이크 프로토콜이 별도로 존재하게 된다[1,2].

중소규모의 다중 서버들로 구성된 웹 서버 클러스터를 수용한 네트워크 구조를 [그림 1]에 나타내었다. 그림에서 보는 것처럼 일반적으로 클라이언트들로 구성된 네트워크 도메인(domain)은 프록시에 의하여 관리되고, 인터넷과 같은 공중망(public network)을 통하여 접속되는 서버들도 디스패처(dispatcher)라고 명명되는 프록시에 의하여 관리되는 구조를 갖게 된다.

웹 서버를 이용한 전자상거래에 대한 관심과 수요가 증가하면서 이러한 웹 서버 클러스터 구조를 수용한 네트워크의 수요가 증가하고 있다. 따라서, 확장성(scalability)을 보장하는 웹 서버 클러스터 네트워크 환경을 구축하는 것은 중요한 기술적 이슈가 되고 있다. 즉, 웹 서버 클러스터를 외부의 클라이언트들이 볼 때 마치 단일 서버로 보일 수 있도록 디스패처가 후위(back-end)의 서버들을 관리하는 기능을 구현하는 것이 중요하다는 것이다. 이때, 디스패처는 후위 서버들간에 부하(load)가 균형을 이룰 수 있도록 효과적인 세션 분배정책(session distribution policy)을 수행 할 수 있어야 한다.

이러한 네트워크 고속의 경향에 부합하도록 웹 서버 클러스터 환경에서도 기존의 후위 다중 서버들을 관리하는 기술을 바탕으로 하여 고속의 TCP 커넥션을 설정하고 관리하는 기술의 중요성이 부각되고 있다.

이를 위하여 TCP Splicing과 같은 TCP 커넥션 고속화 기법이 등장하게 되었다. 이를 활용하여 웹 서버 클러스터 환경에서 고속이며 특정목적에 갖는 세션들을 효율적으로 생성하고 관리하는 기술은 현재 전자상거래 서비스를 강화하는 핵심기술이 된다. 즉, TCP Splicing이란 클라이언트나 서버와 같은 독립 시스템의 TCP/IP 스택에 적용되는 기술이 아니라, 클라이언트와 서버를 연결하는 프록시 서버에서 (L4 switch에 해당)에서

TCP 커넥션을 구성하는 기술이다. 기존의 프록시 서버의 경우는 클라이언트와 서버간에 커넥션을 생성하기 위하여 클라이언트로부터 받은 패킷에 대한 헤더처리를 수행하고 패킷을 프록시의 주프로세서(main processor)에 상주하는 응용계층 프로세스 까지 올리는 작업을 수행하는 방식을 취하였다.

이러한 이유로, 본 논문에서는 전송성능 최적화를 고려할 대상으로 웹 서버 클러스터를 선택하게 되었으며 논의의 초점은 이러한 환경에서 클라이언트와 서버간의 데이터 전송 고속화를 구현하는데 맞추기로 한다. 즉, 가상으로 네트워크 공간을 마련하여 그 공간에서 서버 클라이언트 환경을 구축한 후 클러스터환경(저비용 고성능컴퓨팅환경: 쉽게 말하여 성능이 조금 못 미치는 서버 여러 대를 연결하여 고성능을 창출해 내는 환경)을 구축한다. 이를 위한 준비 과정으로 라운드 로빈 알고리즘에 대하여 고찰 해보고 이를 테스트 한다[3,4].

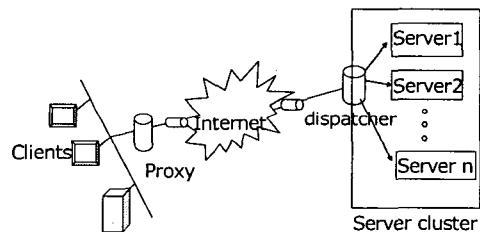


그림 1. 웹 서버 클러스터를 수용한 네트워크 구조

II. 서버의 트래픽 분산

웹서버에 많은 부하가 발생시 이를 여러 대의 컴퓨터에 분산하여 처리하는 스위치와 라우터 시스템들이 겪게 되는 부하는 점점 증가하고 있어 전송속도의 고려가 없이는 전체적인 네트워크의 전송성능은 자연히 저하되게 될 것이다.

이러한 문제점해결로 서버의 트래픽을 분산하여 좀더 안정적인 서비스를 제공하였고, 여러 컴퓨터에 작업을 균등하게 분산시킴으로써 한 대의 컴퓨터를 사용하는 것에 비해 짧은 시간에 사용자가 원하는 작업을 수행하고자 하는 것이다. 수십 대의 컴퓨터를 사용한다고 하더

라도 주로 한 대의 컴퓨터에서 작업이 이루어진다면 원하는 성능 향상을 얻을 수 없을 것이기 때문에, 작업을 “균등”하게 분산시키는 것이 중요하다[5,6].

- 1) 서버에 걸리는 트래픽을 분산 시켜야 한다. 라운드 로빈 알고리즘을 사용하여 서버에 걸리는 부하를 균등하게 분산 시켜야 한다.
- 2) 사용자가 알아보기 편하게 GUI 환경이어야 한다. 트래픽이 분산되어지는 결과를 알아보기 쉽게 그래픽으로 처리하여 보여준다.

1. 시스템 환경

총 5대의 컴퓨터를 필요로 하며 5대의 컴퓨터의 위치는 다음에 나오는 [그림 1] 가상서버 같은 구조를 취한다.

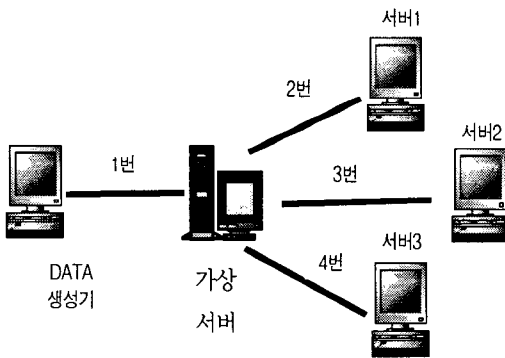


그림 2. 가상서버

1) DATA 생성기

네트워크 상에 클라이언트 역할을 하며 가상의 데이터를 서버에 요청을 하게 된다. 이 때 클라이언트는 가상서버의 IP만 확인이 가능하며 알고 있다.

2) 가상서버

클라이언트에게 보여주는 서버 역할을 하며 실제 서버에게 데이터를 분배해주는 역할을 한다.

3) 서버 3대

실제 일을 처리하는 서버로써 현 단계에서는 데이터를 받아들이는 역할만 수행한다.

III. 시스템 설계

위의 [그림 2]에서 보여지는 것과 같이 가상서버의 환경을 기준으로 네트워크를 설치한다. 다시 말해 5대의 컴퓨터가 네트워크에 연결이 되어 있어야 한다. 데이터 생성기(가상 클라이언트)에서 가상 서버에 접속을 하면 가상서버에서 실제 서버 1,2,3에게 라운드 로빈 알고리즘을 적용하여 데이터를 분배한다. 이 때 패킷의 유동을 파악하며 알아보기 쉽게 그래프로 그려서 표현한다. 먼저 패킷의 유동을 파악할 수 있는 방법을 알아보고 이를 그래프로 처리하는 방법을 익힌 후 라운드 로빈 알고리즘을 사용하여 가상 서버를 구현한다.

1. 패킷 캡처

시스템에 장착된 데이터의 송·수신을 담당하는 NIC (Network Interface Card)가 자기 자신의 시스템으로 오는 패킷만이 아닌 동일 네트워크 상에서 브로드캐스트 되는 모든 패킷을 받아들인다는 사실을 기초로 하여 패킷을 캡처하고 캡처된 패킷의 헤더 정보를 분석하여 패킷의 유량을 그래프로 출력하고 또한 보다 자세한 정보를 출력한다[7].

패킷을 수집하는 패킷 캡처 모듈과 분석모듈 및 출력모듈로 구성되는데 첫째, 패킷 캡처 모듈은 동일 네트워크 상에서 패킷들을 실시간으로 캡처하여 분석모듈로 전달되는데 Libpcap(Portable Packet Capturing Library) 라이브러리를 사용한다. 둘째, 출력모듈은 이렇게 분석된 패킷에 대한 정보에서 프로토콜에 따라 유량(크기)을 그래프로 출력하고 보다 자세한 정보를 보여준다.

2. 시스템 설치환경

Libpcap 프로그램은 패킷을 캡처하는 라이브러리를 제공하는 프로그램으로써 패킷 모니터링하는데 반드시 필요한 라이브러리이다. 이 라이브러리는 다음과 같이 구성 되어져 있다.

Lookup Device는 실제 패킷 캡처에 사용되는 네트워크 인터페이스 카드(NIC)의 디바이스 명을 검색하기 위한 작업이다. 함수로는 pcap_lookupdev() 함수를 사

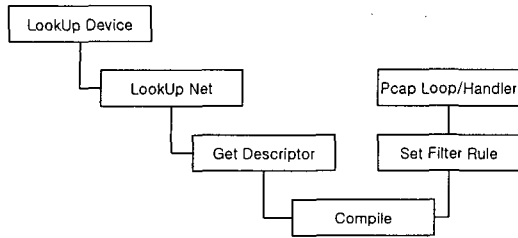


그림 3. Libpcap 흐름도

용한다. 이 함수는 현재 시스템에서 네트워크와 직접 연결되어 있는 NIC 디바이스명의 문자열을 반환하는데 pcap_openlive() 함수와 pcap_lookupnet() 함수의 전달 인자로 사용한다.

Lookup Net는 단계 ①에서 검색된 NIC의 디바이스명에 해당하는 네트워크 아이디(NetID)를 알아내기 위해 사용한다. 사용되는 함수는 pcap_lookup() 함수이며 전달인자로는 NIC의 디바이스 문자열이다.

Get Descriptor는 패킷 캡처를 위한 디스크립터(Discrptor)를 얻는 과정이다. 이 과정에서는 pcap_open_live() 함수가 사용되는데 5개의 전달 인자가 필요하다. 첫 번째 전달인자는 NIC의 디바이스 문자열이고, 두 번째는 캡처할 패킷의 크기를 지정하는데 SNAPLEN이라 한다. 이 크기를 지정하는 이유는 패킷의 최대 크기는 MTU를 고려해서 1514Bytes이다. 그러나 패킷의 헤더부분은 처음 68바이트까지이며 이후부터는 데이터 부분이 되는데 패킷의 정보를 알아내는 데는 헤더부분만 있으면 되기 때문에 일반적으로 이 크기는 "68"을 지정한다. 세 번째 인자는 PROMISCOUS 모드를 지정하는 부분인데 이 곳에는 "1"을 지정한다. 네 번째는 패킷 캡처의 타임아웃(Time Out)값을 지정하고, 마지막으로 에러가 발행하였을 경우 에러 내용을 저장할 버퍼를 지정한다.

Compile 및 Set Filter Rule 단계는 룰셋을 지정하는 부분인데 Libpcap에서는 이 단계에서 지정된 룰의 규칙을 따른다.

Libpcap의 마지막 단계는 캡처를 하는 단계와 캡처된 패킷을 처리하는 단계이다. 패킷을 캡처하기 위해서는 pcap_loop() 함수를 사용하는데 이 함수에도 역시 4

개의 전달 인자가 필요하다. 첫 번째 전달인자는 ③단계에서 얻어진 패킷 캡처 디스크립터를 적어주고, 두 번째는 캡처할 패킷의 수를 지정하는데 "-1"의 값을 적어주면 무한루프를 돌면서 패킷을 캡처한다. 세 번째는 핸들러(Handler) 함수의 포인터를 지정해 주는데 pcap_loop() 함수는 패킷을 캡처할 때마다 지정된 핸들러(Handler) 함수를 호출해 패킷을 처리한다. 마지막 인수는 사용자 정의 인수로서 일반적으로 "NULL"을 명시한다[1].

3. 패킷 캡처모듈

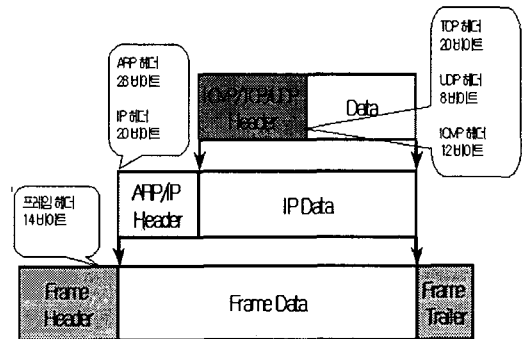


그림 4. 헤더 분석

Libpcap 라이브러리를 사용하여 캡처한 패킷을 헤더 부분을 분석하여 각각 프로토콜 별로 그 량을 파악하여 저장하는 모듈 여러 가지 프로토콜이 존재하나 로드밸런싱 시스템 분석에 필요한 TCP와 IP 그리고 전체 패킷량을 측정하여 분석하고 ARP 프로토콜도 IP에 관련한 프로토콜이므로 같이 측정 분석하여 파일로 저장하여 그래픽 모듈에 넘겨주는 역할을 한다.

4. 그래픽 모듈

파일에 저장되어진 값들을 실시간으로 읽어들이고 그 데이터 값을 토대로 그래프를 그린다. 리눅스 그래픽 프로그램인 GTK를 이용하여 윈도우를 생성하고 각각 프로토콜의 버튼을 생성하여 버튼이 눌러진 프로토콜의 값만 그래프로 변환시켜 보여주는 모듈로 다음과 같은 세 부분으로 나뉜다.

- ① 데이터 입력 저장 : 데이터를 입력 받아서 특정 배열에 저장하는 부분
- ② 윈도우 표시 : 화면에 윈도우 창을 표시하는 부분 (윈도우창, 메뉴, 버튼)
- ③ 데이터 윈도우 표시 : 배열에 저장되어 있는 값들을 차례대로 불러와 그래프를 그리는 부분.(배열 값이 증가해도 처음부터 끝까지 불러와 새로 그린다.)

5. 라운드 로빈 테스트[9]

- ① 데이터 생성기 : 일반 텍스트 값을 생성하는 것으로 유동의 길이의 데이터(문자열)를 서버에 넘겨주는 역할을 한다.
- ② 라운드 로빈 사용 가상 서버 : 서버 1,2,3 에게 차례대로 데이터를 분배하는 역할을 하는 서버를 만드는 것으로 데이터 생성기에서 받아들여지는 데이터(문자열)를 골고루 분배한다.
- ③ 서버1,2,3 : 가상 서버에게서 들어오는 데이터를 각각의 서버에서 텍스트 파일로 저장을 해서 들어온 값들의 양을 비교할 수 있도록 한다.

IV. 시스템 구현

1. 모듈 구성도

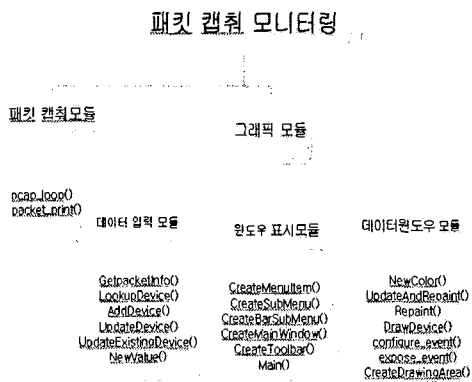


그림 5. 패킷 모니터링

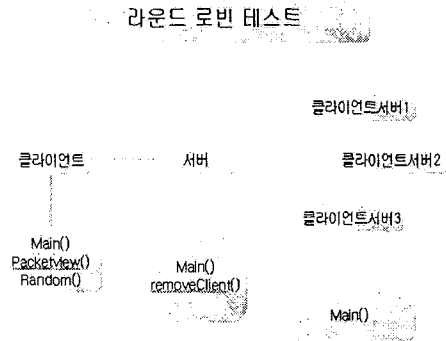


그림 6. 라운드로빈 테스트

2. 연구 결과 및 분석

라운드 로빈 알고리즘에 의해 네트워크의 트래픽이 분산되어지는 과정을 표현하고, 시스템 모니터링은 패킷 모니터링을 하는 것을 말하며 시스템에 Total, TCP, IP, ARP, Other 이 5가지 종류의 프로토콜에서 들어오는 패킷의 량을 측정하는 프로그램으로 다음과 같은 기능을 가진다.

- ① 윈도우 : 리눅스 환경에서 실행화면은 주로 텍스트이다. 그래프를 표현하기 위하여 GTK를 이용한 리눅스 윈도우를 사용하였으며 여기에는 1개의 메뉴와 5개의 버튼이 들어가고 100픽셀로 적용하여 가로세로 100단계로 표현이 가능하다 실제 패킷의 량을 그 최대 크기를 가늠할 수 없지만 적은 경우 10단위부터 많은 경우에는 100만이 넘을 수도 있다 실제 동시접속자 수가 100명 정도 되어지는 사이트에서 보통 30만 정도의 패킷량을 파악할 수 있었다. 이러한 기복 때문에 그래프 표현은 정확하게 표현할 수 없고 같은 상황에서의 비교 파악만이 가능하다. 본 S/W에서는 비교파악만 가능하면 되므로 이정도만 표현 하도록 했으며 추후 기능이 요구 되면 색으로 나누어 표현하는 방법을 사용하여 좀더 정확한 측정을 할 수 있도록 하겠다.
- ② 메뉴 : 종료 메뉴 윈도우 창을 종료시킴
- ③ 아이콘 : 5가지 버튼을 두고 색으로 구별하여 5종류의 프로토콜 량을 측정함

④ 고급 기능 : 5가지 종류의 프로토콜(Total, TCP, IP, ARP, Other)의 량을 표현 할 수 있도록 하여 어떠한 프로토콜들이 많이 발생을 하는지 파악을 할 수 있다[10,11,12].

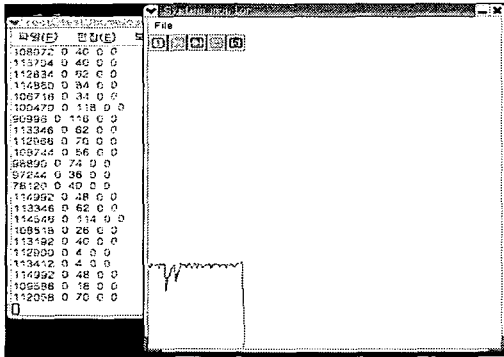


그림 7. 트래픽 분산처리 시스템의 실행화면

[표 1, 2]들은 각각 데이터 생성기, 가상서버, 서버 1,2,3에서 발생하는 패킷의 량을 측정 한 것으로 서버 1,2,3에게 트래픽을 분산시키는지 파악할 수 있다.

지금까지 추출한 데이터에서 보듯 들어오는 데이터의 크기가 많이 차이나지 않는 이상 라운드 로빈 알고리즘은 확실한 형태의 트래픽 분산을 가능하게 해준다. 일부 분에서 오차가 심한경우도 있지만 횡수를 거듭하고 테스트가 장기화 될수록 오차는 줄어든다.

표 1. 각 클라이언트에 10초 30초 동안 데이터전송

<단위 : Byte>

	10초 동안 데이터 전송			30초 동안 데이터 전송		
	1	2	3	1	2	3
1	146776	127916	340039	340039	243058	351996
2	146526	146157	141433	341379	240173	352137
3	144911	141311	147127	342610	359903	354562

표 2. 각 클라이언트에 1분 동안 데이터 전송

<단위 : Byte>

	1분 동안 데이터 전송						
	1	2	3	4	5	6	7
1	697353	702175	758584	693860	726315	696345	724228
2	688999	701005	747809	676327	708681	705556	697063
3	697549	703640	749016	673256	711579	716269	719180

V. 결론

웹서버에 많은 부하가 발생시 이를 여러 대의 컴퓨터에 분산하여 처리하는 스위치와 라우터 시스템들이 겪게 되는 부하는 점점 증가하고 있어 전송속도의 고려없이 전체적인 네트워크의 전송성능은 자연히 저하되게 될 것이다.

이러한 문제점해결로 서버의 트래픽을 분산하여 좀더 안정적인 서비스를 제공하였고, 여러 컴퓨터에 작업을 균등하게 분산시킴으로써 한 대의 컴퓨터를 사용하는 것에 비해 짧은 시간에 사용자가 원하는 작업을 수행하고자 하는 것이다. 수십 대의 컴퓨터를 사용한다고 하더라도 주로 한 대의 컴퓨터에서 작업이 이루어진다면 원하는 성능 향상을 얻을 수 없을 것이기 때문에, 작업을 “균등”하게 분산시키는 것이 중요하다.

다음에 풀어야할 연구 과제로는 캐쉬 알고리즘과 라운드 로빈 알고리즘의 연동과 이에 따르는 보안적인 문제의 해결 방안을 들 수 있겠다. 이유는 다음과 같다. 캐쉬 알고리즘과 라운드 로빈 알고리즘 현 연구과제에서는 서버에 데이터를 분배하는 부분만 작성이 되어져 있다. 하지만 실제 웹서버는 데이터의 요청이 들어오면 이를 클라이언트에 보내어주게 되어있다. 현재 이를 처리하는 부분에 있어서 웹브라우저는 최근에 불러왔던 요청 페이지를 캐쉬에 저장을 해두었다가 같은 명령이 내려진 경우 이를 사용하여 보다 빠른 결과물을 보여주는 역할을 하고 있다. 이에 따라 앞으로의 연구에서는 이 캐쉬의 역할을 파악하여 좀더 빠른 결과를 보여줄 수 있도록 하고 또 부분적인 콘텐츠 라우팅의 기능도 추가해야 할 것이다. 콘텐츠 라우팅이란 같은 응용서비스의 URL정보나 클라이언트 쿠키 정보를 참조하여 이에 맞게 라우팅을 하거나 특수세션(보안이나 QoS)의 재사용성을 높이기 위하여 세션 식별자에 의한 라우팅을 수행하는 것을 의미한다.

참고 문헌

- [1] Craig Hunt저 박창민 역, TCP/IP 네트워크 관리, 한빛미디어, 2000.
- [2] 한국 데비안 사용자모임 공저, 리눅스 서버 최강자 데비안 GNU/리눅스, 한빛미디어, 2002.
- [3] Pat Eyler저 윤성민·김형균 역, 네트워킹 리눅스 TCP/IP 실무 가이드, 인포북, 2001.
- [4] 조셉 슬론저 지순기 역, Network Troubleshooting Tools 한빛미디어 2001.
- [5] Eric Harlow저 노규형 역, GTK+ GDK를 이용한 리눅스 애플리케이션 개발 정보문화사 2000
- [6] 레비 밀호트라 저 김진규 역, IP라우팅, 한빛미디어, 2002.
- [7]http://doit.ajou.ac.kr/~kagi/pub/using_libpcap/libpcap설명
- [8] 진승의, 김태일, 이형호, “웹 서버 클러스터 환경에서의 보안세션 고속화 기술”, 정보처리 학회 논문지 제7권, 2001.
- [9] 김화중 저, 컴퓨터 네트워크 프로그래밍, 홍릉과학출판사, 1997.
- [10] T. Schroeder, S.Goddard, and B. Ramamurthy, "Scalable Web Server Clustering Technologies," IEEE Network, pp.38~45, 2000.
- [11] Web Server Farm Performance. White Paper, Arrowpoint Communications, 1998.
- [12] V. Pai, M.Aron, G.Banga, M.Svendsen, P.Druschel, W.Zwae nepoel, and E, Nahum "Locality Aware Request Distribution in Cluster-based Network Servers," Architectural Support for Programming Languages and Operating systems pp.1~12, 1998.

저자 소개

김 석 수(Seok-Soo Kim)

중신회원



- 1991년 2월 : 성균관대학교 대학원 정보공학과 공학석사
- 2002년 2월 : 성균관대학교 대학원 공학박사
- 1991년 2월~1996년 5월 : 정풍물산(주) 중앙연구소 주임연구원

- 1997년 4월~1998년 1월 : (주)한국 탐웨어 멀티미디어기술연구소 책임연구원
- 1998년 3월~2000년 2월 : 도립거창전문대학 교수
- 2000년 3월~2003년 2월 : 동양대학교 교수
- 2003년 3월~현재 : 한남대학교 멀티미디어학과 교수

최 이 정(E-Jung Choi)

정회원



- 1991년 4월~1998년 2월 : TV-PD(MBC 및 UBC)
- 1998년 3월~1999년 2월 : 혜천대학 영상디자인과 교수
- 1999년 3월~현재 : 한남대학교 멀티미디어학과 교수

이 은 석(Eun-Seok Lee)

정회원



- 1999년 2월 : 세종대학교 대학원 산업디자인학과 미술학석사(컴퓨터그래픽전공)
- 2005년~현재 : 세종대학교 대학원 디자인학과 박사과정중

- 1992년 12월~1998년 7월 : 한국디자인진흥원(KIDP) 디자인개발팀 주임연구원
- 1998년 8월~2000년 8월 : (주)니드 디자인 실장
- 2000년 3월~2000년 8월 : 안동정보대학 애니메이션학과 겸임교수
- 2000년 9월~현재 : 한남대학교 멀티미디어학과 교수