# 메일 서버의 성능과 확장성 향상을 위한 SIRMS 기법

송영호[0] 권택근
충남대학교 컴퓨터공학과
{yhsong, tgkwon}@ce.cnu.ac.kr

# SIRMS Techniques for Improving Performance and Scalability in a Mail Server

Young-Ho Song[0] Taeck-Geun Kwon
Dept. of Computer Engineering, Chungnam National Univ.

**요 약**

최근 전자 메일에서 여러 타입의 데이터를 지원함으로 인하여 메일의 용량이 커지고, 사용자는 다양한 통신 단말을 통하여 메일을 송수신하고 있어 메일 서버의 용량 및 처리 성능에 대한 고속화의 요구가 활발하다. 따라서 이러한 요구에 부흥하기 위한 새로운 기법으로 본 논문은 소스 IP 라우팅과 저렴한 PC의 클러스터를 기반으로 하는 메일 서버를 구현하여 메일 서버의 성능과 확장성을 실현할 수 있게되었다.

## 1. 서 론

전자메일은 인터넷을 통한 중요한 서비스중의 하나로 인터넷의 사용자 증가는 전자메일의 트래픽을 점점 더 증가시키게 될 것이며 각 사용자는 또한 좀더 많은 용량의 메일을 처리할 수 있는 대용량 메일 처리시스템을 찾게 될 것이다.

본 논문은 이러한 요구를 좀더 효율적으로 대응하기 위해 새로운 메일 서버를 설계하고 구현하여 고가의 메일 서버 구입 비용과 관리비용을 최소화할 수 있는 방법으로 SIRMS(Source IP Routing Message System)를 제안하게 되었다. SIRMS는 입력되는 메일의 발신지 주소를 이용해 클러스터(cluster)기반의 하위 노드로 분산하여 할당된 하위 노드가 프톨토콜 처리를 독립적으로 수행하는 기법을 적용하였다. 이러한 처리 기법은 불특정 사고에 의한 메일 서버의 가용성을 높일 수 있고 한 노드에 작업이 집중하지 않도록 하여 좀더 나은 부하분산을 이룰 수 있게 하였다.

본 논문의 2장에서는 SIRMS구현을 위한 관련연구로 메일 프로토콜과 NFS, Qmail 등을 소개하고 3장에서는 기존의 대용량 메일 서버의 구조와 이러한 구조의 문제점을 살펴보고, 4장에서는 SIRMS 즉, 본 논문에서 제안된 메일 서버의 설계와 구현에 대해 기술하였다. 5장에서는 구현된 SIRMS의 실험 및 성능평가를 통해 기존 메일 서버와 비교하여 어떠한 장점들이 있는지 살펴보고, 마지막으로 6장에서는 제안된 메일 서버 구현 기술을 여러 각도로 고찰하고 앞으로의 활용 방안을 기술하는 것으로 결론을 맺는다.

## 2. 관련연구

메일을 보내고 받는데 관련된 메일 프로토콜은 SMTP (Simple Message Transfer Protocol), POP3(Post Office Protocol Version 3), 그리고 IMAP(Internet Message Access Protocol)이 있다[1]. 메일 서버는 메일 서비스 기능을 갖춘 서버(혹은 호스트)를 말하며 단독적으로 하나의 메일서비스 기능을 할 수 있고, 다른 서비스 기능과 복합적으로 기능을 수행하는 서버를 말한다. 메일을 보내는 것과 관련된 메일 프로토콜이 바로 SMTP이며, 메일을 받는 입장에서의 서버 측에 설치되어야 하는 것이 바로 POP3와 IMAP프로토콜을 지원하는 사실이다. 이러한 프로토콜은 메일을 받는 측의 서버가 사용자의 인증단계를 거쳐 자신의 메일 박스에 도착한 메일을 읽어 들일 수 있게 하는 프로토콜이다. 그림 1은 망과 프로토콜의 관계를 보여주고 있다.

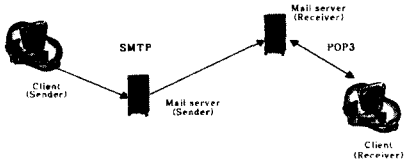


**그림 1 망과 프로토콜의 관계**

NFS(Network File System)는 리모트 파일 시스템의 특정 디렉토리를 마치 자신의 로컬 파일 시스템으로 인식하도록 하여 사용할 수 있는 기법으로, NFS를 위해 서버측 설정과 클라이언트측 설정을 달리하여 구현할 수 있다.

Qmail은 대부분의 UNIX와 UNIX류 시스템에 설치되고 운영되고있으며 현재까지는 가장 안전하고, 간단하며, 신뢰성 있는 MTA(Mail Transfer Agent)로 평가받고 있다[2]. 본 논문의 실험 환경에서 qmail을 사용하고 있는 가장 큰 이유중의 하나로 바로 이러한 안전성을 들 수 있다.

## 3. 기존의 메일 서버

전형적인 대용량 메일 서버의 구조는 프로토콜의 관리와 처리가 중앙에 집중되는 중앙 집중형 메일 서버 구조를 가

* 본 논문은 BK21 대전·충남 정보통신 인력양성 사업단의 RA연구비 지원에 의한 것임.

지고 있다. 그림 2는 이러한 전형적인 중앙 집중형 메일 서버 구조를 보여주고 있다.

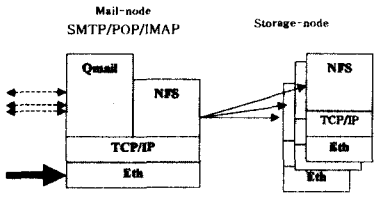


그림 2 기존방식의 메일-노드 및 저장장치-노드 구조

이러한 중앙 집중형 메일 서버 구조는 지금까지 논의된 SMTP/POP3/IMAP 등의 메일 프로토콜을 한 지점에서 처리하게 되고 각 메일에 대한 모든 책임을 분산하지 않으므로해서 점점 더 많아지는 사용자의 요구를 충족할 수 없게 되었다. 인터넷에서의 메일 서비스 요구에 부응하기 위해서 하나의 메일 서버는 하루에도 수천에서 수십만의 사용자에게 서비스할 수 있어야 하고 각 사용자별 수십 메가-바이트 이상의 저장공간을 할당할 수 있는 수십 테라-바이트 이상의 시스템으로 구축되어야 하는데 이러한 전형적인 중앙 집중형 구조는 모든 서비스에 대한 처리가 한 노드에 집중되는 병목현상을 보이게 되며, 더 이상의 확장성을 가지고 있지 않으므로 사용자의 증가에 능동적으로 대처하지 못하게 된다. 이러한 프로토콜 처리의 병목현상은 메일 서버의 전체 성능을 저하시키는 주원인이 된다[3].

## 4. 제안된 메일 서버

지금까지 논의된 전형적인 대용량 메일 서버들에 대한 기술된 문제점을 해결하기 위해, 본 논문은 상호 연결된 PC를 기반으로 하는 클러스터 구조의 새로운 모델을 설계하였다. 이러한 설계는 엄청난 투자비용을 줄이고 값싼 PC기반의 빠른 상호접속을 통해서 구현될 수 있다. 구현된 PC기반의 클러스터 서버는 사용자에게는 단지 전형적인 MTA로 인식되어질 것이다. 하지만 서버의 각 메일 프로토콜 처리가 하위 노드로 분산되는 구조를 갖게 된다. 그림 3은 제안된 새로운 메일 서버의 구조를 보이고 있다.

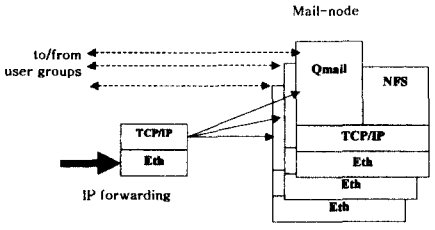


그림 3 제안된 메일 서버 구조

외부 사용자가 보낸 메시지는 "IP forwarding"으로 들어가고 "IP forwarding"은 SMTP메시지의 소스 IP주소를 분석한다[4,5]. 이때 "IP forwarding"은 외부에서 들어오는 패킷에 대한 정보를 미리 가지고 있어야 하며 이러한 정보는 관리자가 어떠한 소스 IP주소를 가진 메시지를 어느 노드에 할당할 것인지에 대한 정책을 설정함으로서 자신의 환경에 적합한 메일 서버를 구현할 수 있게 한다. 이러한 분산 처리방법은 L7(OSI 7 계층)의 사용자 데이터를 사용자별로 그룹화 하여 소스 IP주소를 이용해 L3(OSI 3 계층)로 대체 구현한 것과 같은 효율성을 얻게 되며, 필요시 하위 노드의 추가만으로 늘어나는 사용자의 수에 비례하여 확장이 가능한 구조로 설계되었다. 그림 4는 이와 같은 분산처리 방법하의 부하분산을 위한 패킷 변환 과정을 보이고있다.

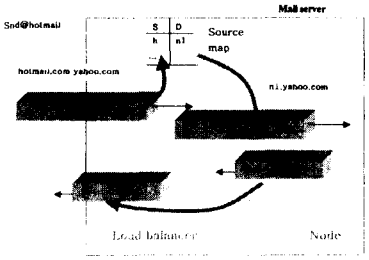


그림 4 부하분산을 위한 패킷 변환과정

Snd@hotmail이 Rcv@yahoo.com에게 메일을 보낼 경우, 보내는 이의 소스 IP "h"와 변환 테이블의 소스 IP를 비교하여 목적지 IP를 내부의 하위노드 "n1"으로 변환한다. 패킷은 n1으로 전송되어지고 n1은 hotmail과의 메일 세션을 유지하기 위해 응답 메시지를 보낸다. 이 패킷은 각각 S와 D에 "n1"과 "h"를 달고 부하분산기에 들어간다. 부하분산기는 다시 관리테이블을 조사하여 S의 "n1"이 내부 노드로부터 나가는 응답 패킷임을 확인하고 S의 "n1"을 "y"로 재 변환하여 전송한다. 이때 하위 각 노드는 엘리어스된 IP를 갖게되는데, 바로 "보내는 이와 받는 이의 불일치 문제"를 해결하기 위한 방법으로 "보내는 노드의 정적 할당 기법"을 적용했기 때문이다. 그림 5는 이러한 보내는 이와 받는 이의 불일치 문제"를 보이고 있다.

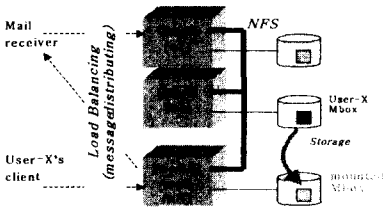


그림 5 메일을 보낼 때의 불일치 문제

외로부터 메일을 받을 때 각 메시지의 소스 IP를 이용해 하위 노드를 할당한 것과는 다르게, 구현된 메일 서버를 이용하는 클라이언트가 메일을 보내고자 할 경우 즉, User-X'의 클라이언트가 Qmail(nN)을 할당받아 메일을 보낼 경우, 메일을 받는 이는 메일 프로토콜 처리에 대한 응답을 보내는 측의 메일 서버에게 다시 보내게 되는데 이때 응답 메시지의 소스 IP를 확인하여 Qmail(n1)에 할당하게 되면 메일을 보내는 이와 받는 이의 불일치 문제가

발생하게 된다. 다시 말해, "IP forwarding"이 사용자 요구에 대해 분산된 두 개의 노드를 할당함으로써 문제가 발생하게 되는 것이다.

이러한 불일치 문제를 해결하기 위해서 본 논문의 메일 서버는 여러 개의 IP를 갖게 하고 클라이언트가 메일 서버를 이용해 메일을 보내고자 할 경우 메일을 받는 절차와 구분하여 처리하게 함으로서 정적인 부하분산(Load balancing)을 하도록 하였다.

## 5. 실험/성능평가

새롭게 설계된 메일 서버의 성능과 확장성을 실험하기 위해, IP forwarding을 위한 SIR(Source IP Router) 하나와 세 개의 하위 노드를 각각 Linux Kernel 2.2.11 source를 변경하여 구현하였고, 각 하위 노드가 메일 프로토콜을 처리하도록 MTA는 qmail과 imapd등을 이용해 구성했다.

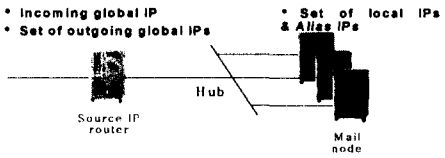


그림 6 실험 환경

그림 6은 성능평가를 위한 기본적인 실험환경을 보이고 있다. 소스 IP 라우터는 들어오는 메일을 받기 위한 글로벌 IP 하나와 메일을 외부로 보내기 위한 여러 개의 엘리어스된 글로벌 IP의 집합으로 이루어지고, 각 하위 메일-노드는 하나의 로컬 IP와 엘리어스된 또 하나의 로컬 IP를 갖게된다.

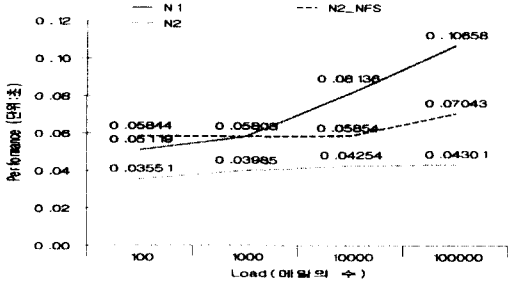


그림 7 메일 서버의 프로토콜 처리 시간 비교

제안된 메일 서버의 성능을 평가하기 위해서 먼저 여러 개의 메일을 단일 서버(N1)에서 보낼 경우 그 프로토콜 처리시간이 얼마나 되는지 알아보았고, 다시 제안된 메일 서버 상에서 하위 노드 2개로 분산하여 처리하는 경우와 비교하였다. 이 때 하위 노드는 각각 NFS를 이용하는 경우(N2_NFS)와 NFS를 이용하지 않은 경우(N2)로 나누어 실험하였다.

그림7의 각 값은 $10^2$개, $10^3$개, $10^4$개, $10^5$개의 메일을 10차례 반복하여 전송한 후, 하나의 메일을 보내는데 걸리는 시간의 평균값을 나타낸 것으로 N1은 메일수의 증가에 따라 하나의 메일을 보내는데 걸리는 평균 시간이 점점 커지고 있음을 확인할 수 있었고, N2는 메일의 양이 많아질수록 N1과 비교해 상당한 처리시간이 감소하는 것을 확인할 수 있었다. 마지막으로 N2_NFS는 하위 두 노드가 NFS를 이용하므로 N2과 비교하여 좀더 많은 처리시간이 걸리는 것을 확인할 수 있었으나 N1과 비교하여 두 개의 노드가 서로 독립적으로 메일 프로토콜을 처리하므로 $10^5$개의 메일을 발송할 때 이론적으로 각각 $5*10^4$개의 메일을 전송하게 되므로 전체 처리시간이 줄어드는 것을 확인할 수 있었다.

## 6. 결 론

본 논문에서 새로운 메일 서버의 설계는 크게 세 가지 관점에서 시도되었고 각 요구에 부응하기 위한 구조로 설계되었다.

첫째, 병렬처리 관점에서 메일 서버는 여러 PC를 클러스터 하여 구현되었고 각 PC가 독립된 프로토콜 처리와 세션관리를 함으로 메일에 대한 병렬 처리를 수행할 수 있게 하였다.

둘째, 확장성 측면에서 기존의 메일 서버가 관리자의 많은 노력을 필요로 하는 반면 제안된 메일 서버는 각 노드의 확장이 쉽고, 불필요한 관리자의 노력을 절감하게 하는 구조로 설계되었다.

셋째, 경제적 측면을 고려하여 값싼 PC를 기반으로 설계하였으며 PC는 고속의 상호 접속을 통하여 외부에서는 단지 하나의 메일 서버로 인식되어지므로 비용을 절감할 수 있었다.

이러한 제안된 구조의 메일 서버는 메일 서버의 중앙 집중화를 피하고 새로운 인터넷환경에 쉽게 적응 할 수 있는 구조로 좀더 많은 데이터를 처리하는 각종 서버의 구현기술에 적용될 수 있을 것이다. 또한 노드의 정적할당 기법을 보완하여 좀더 지능적인 서버구현을 위해 소스 관리테이블을 동적으로 할당하는 기법에 대한 논의가 진행된다면 좀더 나은 부하분산을 얻을 수 있게 될 것이다.

## 7. 참고문헌

[1] Network Working Group Request for comment (RFC) 2060/1939/821/http://www.faqs.org/rfcs.

[2] Dave Sill, "Life with qmail,"(lwq) 16 November 1999.

[3] Yasushi Saito, Brian Bershad, Henry Levy, and Eric Hoffman, "The Porcupine Scalable Mail server." SIGOPS European Workshop 1998.

[4] Nina Bhatti and Rich Friendrich "Web Server support for Tiered Services," Hewlett-Packard Research Labs <IEEE Network> September/October 1999.

[5] David Ranch, Ambrose Au, "Linux IP Masquerade HOWTO,"http://kldp.org/HOWTO/html/IP-Masquerade/IP-Mas querade-HOWTO.html, October 1997.