발신지 IP 주소 포워딩을 이용한 메일 서버의 설계 및 구현

송영호[†] · 권택근[††]

요 약

인터넷 사용자의 증가로 전자 메일은 여러 형태의 미디어를 지원하게 되었고, 멀티미디어 데이터를 포함한 전자 메일의 크기는 점점 더 커지고 있다. 더욱이, 이러한 멀티미디어 전자 메일은 모바일 장치들을 포함한 여러 통신 단말을 통해서 이용되어지고 있다. 본 논문의 주된 목적은 메일 서버의 저장 용량과 사용자 처리 측면에서 고성능과 확장성을 제공하는 효과적인 메일 서버를 설계하고 구현하는데 있다. 따라서 본 논문은 외부에서 들어오는 메일 메시지를 IP 주소 포워딩을 이용해 여러 메일 서버 노드로 분산하는 부하분산-노드를 가진 클러스터 기반의 메일 서버 구조를 제안한다.

Design and Implementation of a Large-Scale Mail Server using Source IP Address Forwarding

Youngho Song[†] and Taeck-Geun Kwon[††]

ABSTRACT

As growing the population of the Internet, e-mail should provide various types of media, and the size of e-mail including multimedia data becomes larger and larger. Furthermore, the multimedia e-mail has been used through various communication terminals including mobile devices. The main objectives of this thesis is designing and Implementing efficient mail server which should provide high performance and scalability in terms of storage capacity and the number of mail users. This thesis proposes a cluster-based mail server architecture which has a load-balance node using IP address forwarding for distributing incoming mail messages into distributed mail server nodes.

Key words: 저장장치-노드, 메일-노드, 부하분산-노드, IP 포워딩, 중앙 집중형 메일 서버, 발신지 주소 포워딩 메시지 시스템(SIFMS)

1. 서 론

전자메일은 인터넷을 통한 중요한 서비스 중의 하나로 인터넷의 사용자 증가는 전자메일의 트래픽을 점점 더 증가시키게 될 것이며 각 사용자는 또한 좀 더 많은 용량의 메일을 처리할 수 있는 대용량 메일 처리 시스템을 찾게 될 것이다[1,2].

본 논문은 이러한 요구를 좀더 효율적으로 대응하기 위해 새로운 메일 서버를 설계하고 구현하여 고가의 메일 서버 구입 비용과 관리 비용을 최소화할 수 있는 방법으로 발신지 IP 주소 포워딩을 이용한 메일 서버를 제안한다.

발신지 IP 주소 포워딩을 위한 부하분산-노드는 기존 대용량 메일 서버의 성능을 향상시키는 방안으로, 입력되는 메일의 발신지 IP 주소를 이용해 클러스터(cluster)기반의 하위 노드로 분산하였고 프로토콜의 처리를 할당된 하위 노드가 독립적으로 수행하는 기법을 적용하였다. 이러한 처리 기법은 불특정 사고에 의한 메일 서버의 가용성을 높일 수 있고 한 노드에 작업이 집중하지 않도록 하여 좀더 나은 부하

[†] 해동정보통신(주) 기술연구소 연구원
[††] 충남대학교 정보통신공학부 조교수

분산(load balancing)을 이룰 수 있게 하였다. 본 논문의 2장에서는 제안된 구조의 메일 서버 구현을 위한 관련연구로 SMTP(Simple Mail Transfer Protocol), POP3(Post Office Protocol V3), IMAP (Internet Message Access Protocol)을 소개하고 3장에서는 기존의 대용량 메일 서버의 구조와 이러한 구조의 문제점을 살펴본다. 4장에서는 본 논문에서 제안된 메일 서버의 설계와 구현에 대해 기술하였고, 5장에서는 이렇게 구현된 발신지 IP 주소 포워딩을 이용한 메일 서버의 실험과 성능평가를 통해서 기존 대용량 메일 서버와 비교하여 어떠한 장점들이 있는지 살펴보고, 6장에서는 제안된 메일 서버의 관리와 활용 면을 다루었다, 마지막으로 7장에서는 본 논문에서 제안된 메일 서버 구현 기술을 여러 각도로 고찰하고 앞으로의 활용 방안을 기술하는 것으로 결론을 맺는다.

2. 메일 프로토콜

메일의 송수신에 관련된 메일 프로토콜은 SMTP, POP3 그리고 IMAP이 있다[4-6]. 메일 서버는 메일 서비스 기능을 갖춘 서버(혹은 호스트)를 말하며 단독적으로 하나의 메일 서비스 기능을 할 수 있고, 다른 서비스 기능과 복합적으로 기능을 수행하는 서버를 말한다[7]. 메일을 보내는 것과 관련된 메일 프로토콜이 바로 SMTP이며, 메일을 받는 입장에서의 서버측에 설치되어야 하는 것이 바로 POP3와 IMAP을 지원하는 데몬으로, 이러한 프로토콜은 메일을 받은 측의 서버를 사용하는 사용자가 인증단계를 거쳐 자신의 메일 박스에 도착한 메일을 읽어 들일 수 있게 하는 프로토콜이다. 이처럼 메일을 처리하는 프로토콜은 상당히 복잡하여 메일을 송수신하기 위한 프로토콜 처리의 병목현상으로 메일 서버의 고성능화에 어려움이 있으며, 메일 사용자의 특성상 여러 발신지로부터 메일이 전송되기 때문에 단순히 저장장치-노드의 증가와 NFS(Network File System)를 통한 성능 향상은 그 한계가 있다.

2.1 SMTP

SMTP는 보내는 SMTP와 받는 SMTP로 구성된다. 보내는 SMTP는 밖으로 나가는 메일 메시지와 관련되어 있고, 받는 SMTP는 인터넷으로부터 들어오는 메시지와 관련이 있다. 이처럼 SMTP는 단지 한 기계에서 다른 기계들에게 메일을 전달하는 것으로, 그림 1은 이러한 SMTP사이의 교체되는 양식을 간단히 정의하고 있다[3].사용자가 작성한 메일은 큐에 들어가고 보내는 SMTP는 큐를 주기적으로 검사하여 메일이 있는지 확인한 후 TCP를 열어서 메일 전송을 위해 원격지의 목적지와 연결을 설정한다. 받는 SMTP는 로컬 TCP 25번 포트로 유입되는 메일을 받아들여 메일 박스에 메일을 복사한다.

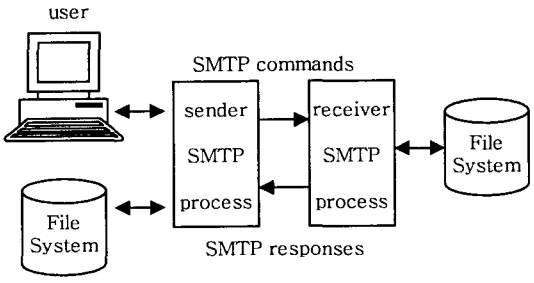


그림 1. 보내는 SMTP와 받는 SMTP사이의 관계

2.2 POP3

POP3는 인터넷을 통해 자신에게 온 메일을 읽기 위해 직접 메일 서버에 접속하지 않아도 도착한 메일을 자신의 컴퓨터로 가져올 수 있도록 해준다. 즉 MUA(Mail User Agent)가 MTA(Mail Transfer Agent)의 메일 박스에서 메일을 가져오는 프로토콜이다. 그림 2는 이러한 POP3와 SMTP가 인터넷 망에서 클라이언트와 어떻게 연결되는지에 대한 관계를 보여주고 있다.

메일을 보내는 클라이언트는 SMTP를 이용하여 메일서버에 TCP접속을 요구하고 요구를 받은 메일 서버는 리모트의 받는 서버와 다시 TCP연결을 설정

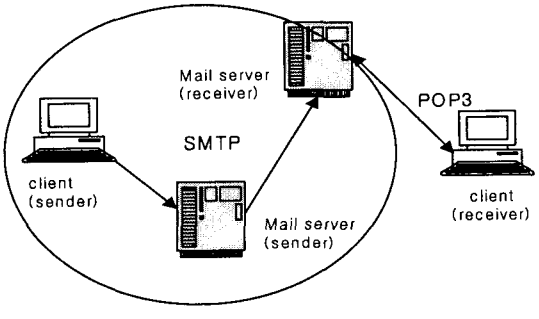


그림 2. 망과 POP3의 관계

하여 메일을 전송한다. 리모트의 클라이언트는 자신의 메일 서버에 POP3를 이용해 자신의 메일 박스에 담겨진 메일을 가져간다[4,7].

2.3 IMAP

IMAP의 기본적인 역할은 POP3와 동일하다. 메일 클라이언트는 리모트의 서버에 있는 메시지를 마치 자신의 로컬 컴퓨터에 있는 것처럼 메일을 다룰 수 있다. POP3의 "저장 후 전달" 비교하여 "리모트 파일 서버"에 비유하는 이유가 바로 여기에 있다. IMAP 서버는 TCP포트 143번을 링크 레벨로 이용하며, POP3의 단순 저장방식에 비해 사용자의 메일을 관리하는 방법들을 제시한다. 그림 3은 이러한 IMAP이 메시지를 가져오기 위한 각 메시지의 흐름과 상태를 보여주고 있다. IMAP서버는 클라이언트로부터 TCP 143번 접속이 요구되면 OK greeting이라는 메시지와 함께 접속이 성공되었음을 알리고 곧바로 사용자 인증 단계로 들어간다, 만약 접속과 동시에 접속이 종료되거나 혹은 인증에 실패 시 BYE greeting이란 메시지를 보내고 연결을 종료한다. 사용자가 자신의 아이디와 패스워드를 입력하면 인증을 거쳐 successful LOGIN이라는 메시지를 보낸다. 이후 메일을 읽어오기 위한 메일 박스를 선택하고 선택한 메일 박스가 유효하면 이때부터 메일 박스로부터 메일을 읽어 들일 수 있게 된다. 모든 종료는 Logout으로 접속 종료된다[5].

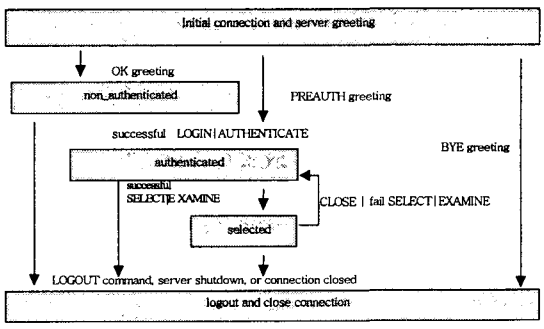


그림 3. IMAP의 각 상태와 메시지 흐름도

3. 기존의 메일 서버

전형적인 대용량 메일 서버의 구조는 프로토콜 관리와 사용자 메일 박스가 중앙에 모여서 모든 처리를 전담하는 중앙 집중형 메일 서버 구조를 가지고 있다. 그림 4는 이러한 전형적인 중앙 집중형 메일 서버 구조를 보여주고 있다[1,2].

중앙 집중형 메일 서버 구조는 지금까지 논의된 SMTP/POP3/IMAP 등의 메일 프로토콜을 한 지점에서 처리하게 되고 각 메일에 대한 모든 책임을 분산하지 않으며 각 저장장치의 파일시스템을 독립적으로 구성된 하나의 시스템으로 사용하고 있으므로 해서 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. 또한 이러한 시스템을 구축하는데 적지 않은 투자비용이 든다는 사실이다[2,3].

그림 5의 기존방식의 메일-노드 및 저장장치-노드 구조를 살펴보면서 전형적인 대용량 메일 서버의 구조적인 문제점을 좀더 자세히 살펴보자. 먼저 외부로부터 메일이 들어오는 과정을 보면, 메일-노드로 들어오는 모든 메시지는 Qmail의 받는 SMTP서버에 의해서 각각 응답되어지고 처리되어진다. 이러한 Qmail의 세션관리가 끝나고 나면 큐에 각각의 사용자 메일이 들어가게 되는데 이때 NFS를 통해서 사용자 메일 박스가 저장장치-노드로 분산되어 관리된다[7]. 결국 저장장치-노드의 분산을 통해서 저장장치

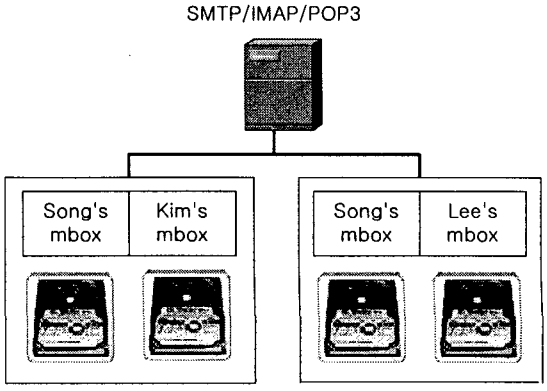


그림 4. 전형적인 중앙 집중형 메일 서버

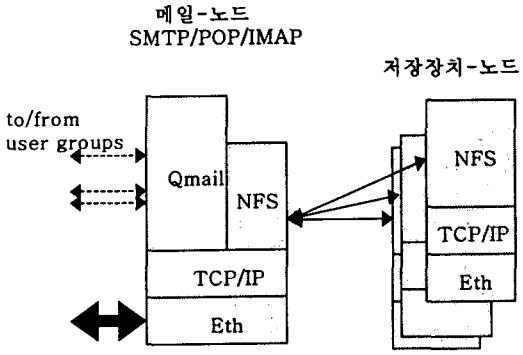


그림 5. 기존방식의 메일-노드 및 저장장치-노드 구조

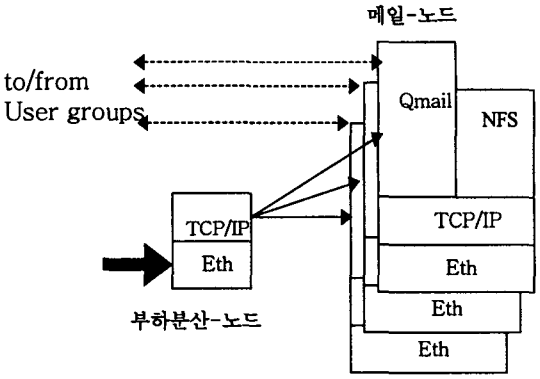


그림 6. 제안된 메일 서버 구조

의 한계를 극복한 반면 SMTP프로토콜에 대한 처리를 위해 프로세서는 모든 메일에 대한 세션을 관리하게 된다. 이러한 작업의 집중은 프로토콜 처리의 병목현상을 발생시킨다. 그러므로 중앙 집중형 프로토콜 처리의 병목현상은 메일 서버의 전체 성능을 저하시키는 주원인이 된다. 마찬가지로 메일을 외부로 발송할 경우에도 이와 같은 프로토콜 처리의 병목현상을 피할 수 없게 된다.

4. 제안된 메일 서버(SIFMS: Source IP Forwarding Message System)

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[6]. 구현된 PC기반의 클러스터 서버는 사용자에게는 단지 전형 적인 MTA로 인식되어질 것이다. 하지만 서버의 각 메일 프로토콜 처리가 하위 노드로 분산되는 구조를 갖게 된다. 그림 6은 제안된 새로운 메일 서버의 구조를 보이고 있다. 이러한 분산 프로토콜 처리 구조에서의 메일을 받는 과정을 기존 방식과 비교하여 보면, 먼저 리모트의 사용자는 구현된 메일 서버의 하위 노드에 무엇이 있는지 그리고 어떠한 노드가 자신의 메일을 관리하는 서버인지 알 수 없으며 보통의 메일 서버에 메일을 보내는 것과 같은 방법으로 SMTP를 이용해 메일을 보내게 된다. 이렇게 보낸 메시지는 부하분산-노드로 들어가고 부하분산-노드는 SMTP메시지의 발신지 IP 주소를 분석한다. 이때 부하분산-노드는 외부에서 들어오는 패킷에 대한 정보를 미리 가지고 있어야 하며 이러한 정보는 관리자가 어떠한 발신지 IP 주소를 가진 메시지를 어느 노드에 할당할 것인지에 대한 정책을 설정함으로서 자신의 환경에 적합한 메일 서버를 구현할 수 있게 한다. 이렇게 가지고 있는 정보를 이용해 부하분산-노드는 메시지를 내부의 한 메일-노드에 할당한다. 내부의 각 메일-노드는 각자의 메일 프로토콜 서버를 가지고 있고 이렇게 할당받은 메시지에 대해 Qmail은 SMTP에 대한 연결 설정에서 세션의 관리까지 모든 책임을 단독으로 지게 된다. 즉, 할당받은 메일-노드는 각 메시지에 대한 처리를 독립적으로 수행할 수 있게 되고 프로토콜 처리의 병목현상은 일어나지 않게 된다. 이러한 분산 프로토콜 처리는 메일 서버의 성능을 향상시키는 중요한 요소가 된다.

이러한 분산 처리방법은 L7(OSI 7 계층)의 사용자 데이터를 사용자별로 그룹화하여 발신지 IP주소를 이용해 L3(OSI 3 계층)로 대체 구현한 것과 같은 효율성을 얻게 되며, 필요 시 하위 노드의 추가만으로 늘어나는 사용자의 수에 비례하여 확장이 가능한 구조로 설계되었다. 그림 7은 이와 같은 분산처리 방법하의 부하분산을 위한 패킷 변환 과정을 보이고 있다. Snd@hotmail이 Rcv@yahoo.com에게 메일을 보낼 경우, 보내는 이의 발신지 IP h와 변환 테이블의 발신지 IP를 비교하여 목적지 IP를 내부의 하위노드 n1으로 변환한다. 패킷은 n1으로 전송되어지고 n1은 hotmail과의 메일 세션을 유지하기 위해 응답 메시지를 보낸다. 이 패킷은 각각 S와 D에 n1과 h를 달고 부하분산-노드에 들어간다. 부하분산-노드는 다시

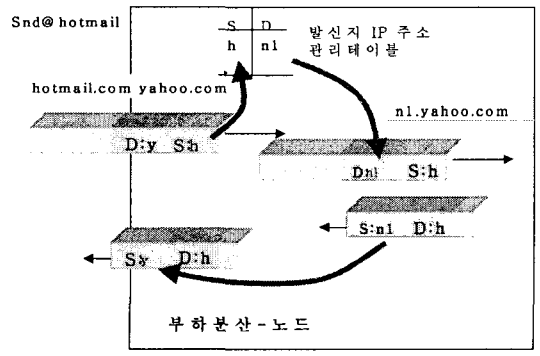


그림 7. 부하분산을 위한 패킷 변환과정

발신지 IP 주소 관리테이블을 조사하여 S의 n1이 내부 노드로부터 나가는 응답 패킷임을 확인하고 S의 n1을 y로 재 변환하여 전송한다.

이러한 방식으로 hotmail로부터 유입되는 메일의 모든 세션 관리는 n1이 처리하게 되는 것이다[8-13]. 하위 각 노드는 엘리어스된 IP를 갖게 되는데, 바로 "보내는 이와 받는 이의 불일치 문제"를 해결하기 위한 방법으로 "보내는 노드의 정적 할당 기법"을 적용했기 때문이다.

그림 8은 이러한 "보내는 이와 받는 이의 불일치 문제" 를 보이고 있다. 외부로부터 메일을 받을 때 각 메시지의 발신지 IP를 이용해 하위 노드를 할당한 것과는 다르게, 구현된 메일 서버를 이용하는 클라이언트가 메일을 보내고자 할 경우 즉, User-X'의 클라이언트가 Pop/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 발신지 IP를 확인하여 Qmail (n1)에 할당하게 되면 메일을 보내는 이와 받는 이의 불일치 문제가 발생하게 된다. 다시 말해, 부하분산-노드가 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 부하분산을 하도록 하였다.

적용된 "보내는 노드의 정적할당 기법"은 그림 9에 잘 나타나있는 것처럼, 하위 시스템이 보내는 메일을 위해서 엘리어스된 로컬 IP를 하나 더 갖게 하여 응답 시 엘리어스된 IP를 사용하여 부하분산-노드에게 메일을 보내는 것임을 알리고 외부로 나가는 발신지 IP를 엘리어스된 IP로 변경하여 보낼 수 있도록 하였다.

이상과 같이 구현된 메일 서버가 엘리어스된 IP를 갖는 이유를 설명하였다. 이젠 실제 메시지가 어떻게 변환되고 처리되는지 살펴보도록 하자. 살펴볼 메시지의 흐름은 크게 두 가지로 나누어 볼 수 있는데 첫째, 메일이 외부로부터 들어올 때 즉, 다른 메일 서버를 사용하는 클라이언트가 본 논문의 메일 서버의 클라이언트에게 메일을 보낼 때의 메시지 흐름이고 둘째, 로컬 클라이언트가 메일 서버를 이용해 리모트의 사용자에게 메일을 보내고자 할 경우의 메시지 흐름이다. 첫 번째 경우의 메시지 흐름은 부하분산을 위한 패킷 변환 과정에서 이미 살펴보았고, 두 번째 경우의 메시지 흐름을 살펴보도록 하자. 그림 10은 외부로 나가는 메일에 대한 메시지 흐름을 보이

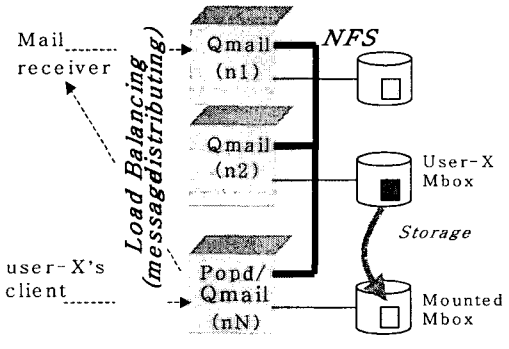


그림 8. 메일을 보낼 때의 불일치 문제

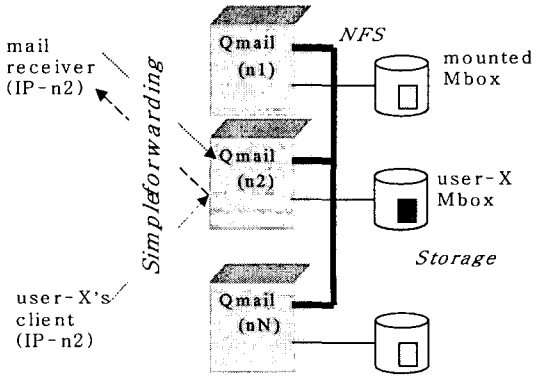


그림 9. 보내는 노드의 정적할당 기법

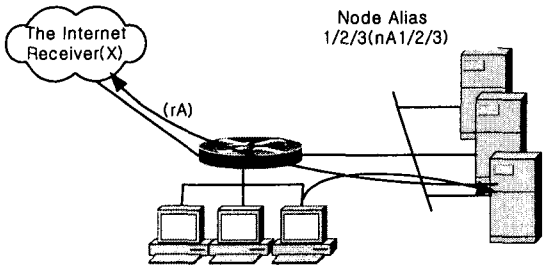


그림 10. 외부로 나가는 메일에 대한 메시지 흐름

고 있다. 클라이언트3(c3)에서 들어온 패킷은 그 목적지 IP에 엘리어스된 글로벌 IP rA를 달고 부하분산-노드에 들어오게 된다. 이때 부하분산-노드는 발신지 IP를 확인하여 사용자가 그룹 3에 속한다는 것을 알고 목적지 IP를 엘리어스된 로컬노드 nA3로 할당한다. 이때부터 nA3는 모든 프로토콜과 세션의 관리를 책임지게 되고 nA3로부터 나오는 패킷은 부하분산-노드에서 그 발신지 IP를 다시 rA로 변환하여 c3와의 세션을 계속 유지하도록 설계하였다.

5. 실험 및 성능평가

새롭게 설계된 메일 서버의 성능과 확장성을 실험하기 위해, 본 논문은 발신지 IP 주소 포워딩을 위한 하나의 부하분산-노드와 3개의 하위 노드를 각각 Pentium Ⅲ 450 MHz, RAM 128 MB 환경 하에서 Linux 2.2.11 Kernel 소스를 변경하여 구현하였고, 각 하위 노드가 메일 프로토콜을 처리하도록 qmail V1.0.3, imap V4.5, NFS V2, imap-4.5-3mdir4.i386 등을 이용해 구성했다. 그림 11은 성능평가를 위한 기본적인 실험환경을 보이고 있다. 부하분산-노드는 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 갖게 된다.

메일 서버의 성능은 보내는 측, 혹은 받는 측의 메일 서버의 성능에 무척 의존적이며, 하드웨어의 성능에 매우 민감하므로 제안된 메일 서버의 성능을 상대적으로 평가하기 위하여 같은 성능의 머신에 리눅스를 설치하여 단일 메일 서버를 구축하였다.

먼저 부하분산-노드의 오버헤드를 측정하기 위해 제안된 구조의 특정 하위 한 노드의 사용자에게 한 개의 메일을 10차례 반복 전송하여 TCP접속을 위한 부하와 SMTP 세션에 대한 부하를 측정하고, 다시 부하분산-노드를 제거한 후, 하위 한 노드의 사용자에게 같은 실험을 반복하여 평균 처리 시간을 측정하였다. 이렇게 측정된 값을 비교하여 한 개의 메일을 처리하기 위한 부하분산-노드의 오버헤드를 측정하였다. 측정결과 TCP 연결설정과 SMTP 처리를 위해 총 10.86ms의 추가 부하가 있었다. 한 개의 메일을 처리하는 경우의 오버헤드는 전체 메일(한 개)의 처리 시간에 대한 오버헤드가 적지 않았다. 그러나 메일을 두개 발송하는 경우의 오버헤드는 7.2%, 다섯 개인 경우 6.2%, 열 개인 경우 약 4% 정도로 낮아졌다. 이러한 부하분산-노드의 오버헤드가 전체 메일 서버의 성능에 어떠한 영향을 미치는지 실험하기 위해 메일의 수를 증가시키면서 메일 서버의 총 메일 처리 시간에 대한 발신지 IP 라우터의 오버헤드를 같은 방법으로 측정하였다. 그림 12는 메일 10, 10^2, 10^3, 10^4, 10^5개를 전송하여 이때의 부하분산-노드의 오버헤드를 측정한 그래프이다. 메일 수의 증가에 따

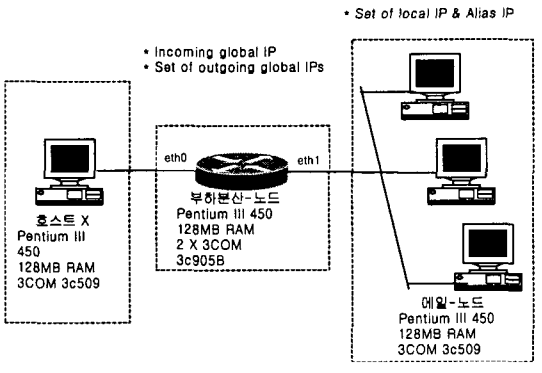


그림 11. 실험환경

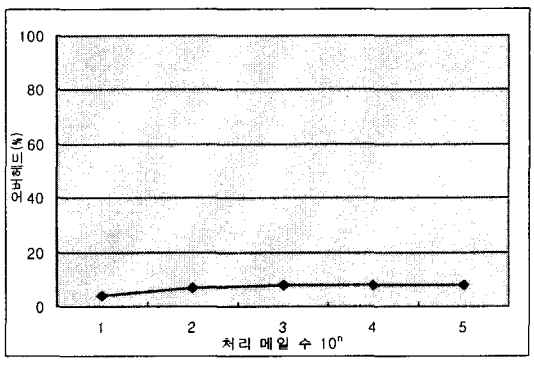


그림 12. 부하분산-노드의 오버헤드

라 즉, 메일을 처리하기 위한 작업량(부하)의 증가에 따라 메일을 처리하기 위한 부하분산-노드의 오버헤드는 비교적 일정한 것을 알 수 있다.

제안된 구조의 메일 서버는 Qmail의 기본 설정을 변경하지 않고 20개의 프로세스, 다시 말해 20개의 쓰레드를 이용해 메일을 전송하고 받게 되며, 각 메일을 위한 TCP 연결 설정이 필요하게 된다.

다음은 구현된 메일 서버의 메일 수와 노드 수를 증가시켜 성능을 평가하였다. 그림 13의 최적 실행가능 곡선(optimally feasible)은 부하분산-노드를 빠른 하드웨어로 구현하여 오버헤드를 줄인 것으로 현재의 실험 환경에서 구현된 서버의 성능을 평가하는 최적의 기준으로 삼았다. 그림에서 보이는 것처럼 하위 노드의 증가로 단위 시간당 처리될 수 있는 메일의 양을 증가시킬 수 있게 되므로 제안된 메일 서버(SIFMS)가 확장성을 갖게 된다는 것을 알 수 있다.

그림 14는 제안된 메일 서버의 성능을 상대적으로 평가하기 위하여 초당 처리할 수 있는 메일의 수(처리율)를 측정하여 그래프로 나타내었다. 그래프의 각 값은 중앙 집중형(CENT)과 하위노드를 각각 2개(SIFMS_N2), 3개(SIFMS_N3) 갖는 제안된 메일 서버에 메일의 수를 증가시키면서 10회씩 반복 실험하여 평균값을 나타낸 것이다. 메일이 103개인 경우 초당 처리할 수 있는 메일의 평균수는 CENT의 경우 약 8.07개임을 알 수 있으며, SIFMS_N2의 경우 약 14.84, SIFMS_N3의 경우 약 22.26개로 증가하는 것을 알 수 있다

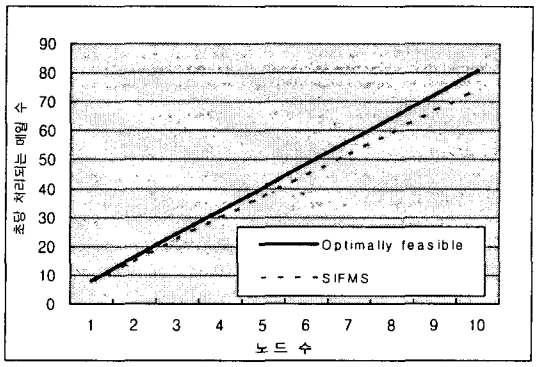


그림 13. 노드 수에 따른 단위 시간당 처리되는 메일의 수

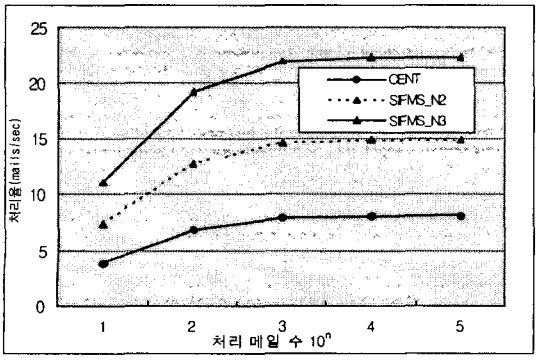


그림 14. 처리율 비교 그래프

이상과 같이 본 논문에서 제안된 메일 서버는 하위 노드의 추가를 통해서 메일 서버의 확장성을 갖게 되며, 확장된 메일 서버는 단위 시간당 메일의 처리량을 증가시킬 수 있게 된다.

6. SIFMS의 관리와 활용

메일 서비스 제공자는 수백만의 사용자를 관리해야 하므로 관리의 효율성은 메일 서비스에서 중요한 요인이 된다. SIFMS는 독립적인 소 용량 메일 서버가 상호 연결된 방식으로 장애 발생 시 그 범위가 최소화되고, 사용자 당 할당된 메일 박스는 단순한 파일 형태로 관리되므로 그 관리의 효율성이 뛰어나다.

현재 인터넷 서비스를 유료화 하려는 시도가 활발한데, 이는 서비스 구축비용의 증가에 그 요인이 있다. 따라서 서비스 구축비용을 최소화하는 경우에 현재의 인터넷 서비스를 지속적으로 유지할 수 있고, 새로운 서비스의 도입이 용이해진다. 따라서 새로 확장되는 메일 서버를 본 논문을 바탕으로 시스템을 구축하는 경우에 비용절감을 꾀할 수 있다. 하지만 본 논문을 활용한 상용화에는 실제 메일 트래픽에 대한 성능 분석과 시스템의 안정화 그리고 사용자 메일 박스가 일반 파일로 관리되어 운용상 편리하지만 수십만 사용자의 메일 박스를 관리하기 위해 필요한 관리 시스템의 개발이 필요하다. 이러한 관리 시스템에서는 장애를 찾고, 사용자에 대한 정보를 관리하거나 사용 현황 등 통계 정보를 수집하여 전체적인 시스템 관리에 활용할 필요가 있다.

7. 결론 및 향후 연구 과제

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 하였으며, 각 노드의 저장 장치를 분산 공유하여 작업량을 분산할 수 있었다.

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.

기존의 메일 서버는 대용량 서버 제품이나 대용량 저장 장치의 수입을 통해 구축되는 실정이다. 그러나 이러한 제안된 구조의 메일 서버는 전형적인 메일 서버의 중앙 집중화를 피하고 새로운 인터넷환경에 쉽게 적응 할 수 있는 구조이므로, 메일 서버를 새로 확장하는 경우와 새로운 메일 서비스를 구축하는 경우에 본 논문의 결과를 활용할 수 있으며 좀더 많은 데이터를 처리하는 각종 서버의 구현기술에 적용될 수 있을 것이다. 또한 실제 인터넷 상황에서의 세밀한 성능평가와 더불어, 본 논문에서의 노드 정적할당 기법을 보완하여 좀더 지능적인 서버구현을 위해 발신지 관리테이블을 동적으로 할당하는 기법에 대한 논의가 진행된다면 좀더 나은 부하분산을 얻을 수 있게 될 것이다.

참 고 문 헌

[1] Y. Saito, B. Bershad, H. Levy, and E. Hoffman, "The Porcupine Scalable Mail Server," *SIGOPS European Workshop*, 1998.

[2] A. Westerlund, L. H. Astrand, J. Danielsson, "Meta-a Freely Available Scalable MTA," USENIX Technical Conference. 1999.

[3] J. B. Postel "Simple Mail Transfer Protocol," *IETF RFC821,* August 1982

[4] J. Myers, C. Mellon, M. Rose, "Post Office Protocol-Version 3," *IETF RFC1939,* May 1996.

[5] M. Crispin "Internet Message Access Protocol," *IETF RFC2060,* December 1996.

[6] A. Fox, S. D. Gribble, Y. Chawathe, E. A. Brewer, P. Gauthier, "Cluster-Based Scalable Network Services," *SOSP(Symposium on Operating Systems Principles)* pp.38-91, 1997.

[7] D. Sill, "Life With Qmail," *http://lifewithqmail.org,* September 2001.

[8] M. Hasenstein, "IP Address Translation," *http://www.csn.tu-chemnitz.de /HyperNews/get/linux-ip-nat.html,* 1997.

[9] D. Ranch, A. Au, "Linux IP Masqurade HOWTO," *http://kldp.org/HOWTO/html/IP-Masqurerade/IP-asqurerade-HOWTO.html,* October 1997.

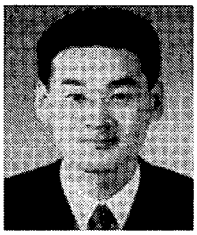
[10] P. Srisuresh, M. Holdrege, "IP Network Address Translator (NAT) Terminology and Considerations," *IETF RFC2663,* August 1999.

[11] M. Borella, J. Lo, "Realm Specific IP:Framework," *IETF Internet Draft <draft-ieft-nat-rsip-framework-05.txt>*, July 2000.

[12] M. Holdrege, P. Srisuresh, "Protocol Complications with the IP Network Address Translator," *Internet Draft <draft-ietf-nat-protocol-complications-06.txt>*,October 2000.

[13] D. Senie, "NAT Friendly Application Design Guidelines," *Internet Draft <draft-ietf-nat-app-guide-03.txt>*, July 2000.

[14] M. Andree, "MTA Benchmark," *CEST (The Center for Science & Technology Studies)*, October 2001. also in *http://www-dt.e-technik.uni-dortmund.de/~ma/postfix/bench2.html.*



송 영 호

1996년 2월 한밭대학교 컴퓨터공학과(공학사)
2002년 2월 충남대학교 대학원 컴퓨터공학과(공학석사)
2002년 1월~현재 해동정보통신㈜ 기술연구소 연구원
관심분야 : 멀티미디어 통신, 멀티미디어 서버, 임베디드 시스템, 정보보호, VoIP(Voice over Internet Protocol)
E-mail: yhsong@haedong.re.kr

권 택 근

1988년 2월 서울대학교 컴퓨터공학과(공학사)
1990년 2월 서울대학교 대학원 컴퓨터공학과(공학석사)
1996년 2월 서울대학교 대학원 컴퓨터공학과(공학박사)
1992년 1월~1998년 8월 LG 전자 R&D 센터 연구원
1993년 1월~1994년 1월 미국 Washington University, St. Louis 방문 연구원
1998년 9월~현재 충남대학교 정보통신공학부 조교수
관심분야 : 멀티미디어 통신, 멀티미디어 서버, 초고속통신망, 시스템 소프트웨어
E-mail: tgkwon@ce.cnu.ac.kr