

# 기가비트 라우터 시스템에서의 내부 데이터 처리를 위한 소프트웨어 구조

이 왕 봉<sup>†</sup> · 정 영 식<sup>††</sup> · 김 태 일<sup>††</sup> · 방 영 철<sup>†††</sup>

## 요 약

인터넷 사용자의 증가와 인터넷을 이용한 전자상거래(E-commerce)의 확산 그리고 네트워크 게임 등으로 인해 인터넷상의 사용자 데이터는 끊임없이 증가하고 있는 상태이다. 이러한 인터넷의 확산을 지원하기 위해 고속 통신을 가능하게 할 초고속 라우터가 상용화되는 추세이다. 고속의 패킷 라우팅 처리를 위해 고안된 라우터 구조를 살펴보면, 라인 인터페이스와 호스트 프로세서는 각각 제어용 프로세서를 가지고 있어 독립된 디바이스로 동작하며 패킷 스위칭과 고속의 패킷 포워딩, 신속한 FIB(Forwarding Information Base) 처리 등을 구현하고 있다. 본 논문에서는 라우팅 정보를 관리하는 유니캐스트 및 멀티캐스트 라우팅 프로토콜과 OAM(Operation And Maintenance) 관련 패킷을 비포워딩(non-forwarding) 패킷으로 정의하고, 이를 처리하는 라인 인터페이스와 호스트 프로세서에서의 소프트웨어 구조를 제시하였다. 또한 분산 시스템에 요구되는 프로세서 간의 통신 메커니즘으로 프로세서간 통신 처리용 프로토콜(Inter-Processor Communication Message Protocol)을 설계 및 적용하여 기존의 UDP/IP를 이용하는 통신 메커니즘에 비해 성능이 향상됨을 확인하였다.

## The software architecture for the internal data processing in Gigabit IP Router

Wang-Bong Lee<sup>†</sup> · Young-Sik Chung<sup>††</sup> · Tae-Il Kim<sup>††</sup> · Young-Cheol Bang<sup>†††</sup>

## ABSTRACT

Internet traffic is getting tremendously heavier due to the exponential growth of the Internet users, the spread of the E-commerce and the network games. High-speed routers for fast packet forwarding are commercially available to satisfy the growing bandwidth. A high-speed router, which has the decentralized multiprocessing architecture for IP and routing functions, consists of host processors, line interfaces and switch fabrics. In this paper, we propose a software architecture tuned for high-speed non-forwarding packet manipulation. IPCMP (Inter-Processor Communication Message Protocol), which is a mechanism for IPC (Inter-Processor Communication), is also proposed and implemented as well. Proposed IPC mechanism results in faster packet-processing rate by 10% as compared to the conventional IPC mechanism using UDP/IP.

키워드 : 고속 라우터(IP Router), 라우팅 프로토콜(Routing Protocol), 분산처리구조(Distributed Processing), 프로세서간 통신(IPC)

## 1. 서 론

인터넷 사용자의 증가와 인터넷을 이용한 전자 상거래(E-commerce)의 확산 그리고 네트워크 게임 등으로 인해 인터넷상의 사용자 데이터는 끊임없이 증가하고 있는 상태이다[1]. 이와 같은 인터넷의 확산을 지원하기 위해서는 대용량의 고속 라우터가 요구된다. 현재 기가비트 속도의 라우팅을 지원하고 테라비트의 스위칭 속도를 지원하는 라우터가 상용화되는 추세이다.

고속 라우터는 패킷 포워딩을 처리하는 라인 인터페이스와 라우팅 프로토콜과 OAM(Operation And Maintenance)

을 처리하는 호스트 프로세서, 라인 인터페이스에서 처리되는 패킷을 스위칭하는 스위치 패브릭(Switch Fabric)으로 구성되어 있다[2, 3]. 라우터에서 성능에 영향을 미칠 수 있는 요소로 포워딩 패킷(Forwarding Packet)을 처리하는 포워딩 엔진(FE : Forwarding Engine), 라인 인터페이스간의 패킷 스위칭을 처리하는 스위치 패브릭의 하드웨어 부분과 라우팅 정보를 수집하여 FIB(Forwarding Information Base)를 생성하는 모듈, 비포워딩(non-forwarding) 패킷을 처리하는 모듈의 소프트웨어 부분을 고려할 수 있다.

하드웨어 요소로서 포워딩 엔진과 스위칭 패브릭의 성능 향상을 위한 라우팅 룩업 알고리즘, 최적의 버퍼 관리 알고리즘 그리고 향상된 실리콘(silicon) 기술에 대한 연구가 진행되고 있지만[4-6], 소프트웨어 요소의 성능 향상에 관련된 기술 연구는 미흡한 실정이다. 라우터에서는 라인 인터

† 준 회 원 : 한국전자통신연구원 선임연구원

†† 정 회 원 : 한국전자통신연구원 책임연구원

††† 정 회 원 : 한국산업기술대학교 컴퓨터공학과 교수

논문접수 : 2002년 6월 20일, 심사완료 : 2002년 10월 9일

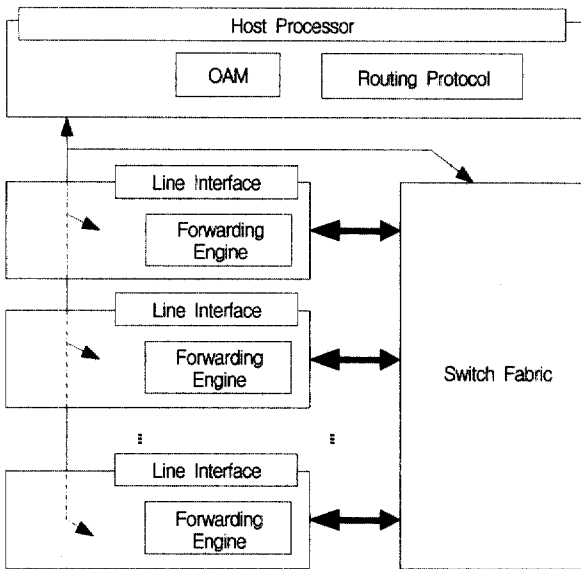
페이스 카드와 스위치의 고속화와 함께 정확한 패킷 라우팅을 처리하기 위해 신속한 라우팅 정보의 갱신이 필요하다. 이를 위해 유니캐스트 및 멀티캐스트 라우팅 프로토콜 패킷 그리고 FIB를 신속하게 처리할 수 있는 소프트웨어 구조가 필요하다. 본 논문에서는 라우팅 정보를 관리하는 유니캐스트 및 멀티캐스트 라우팅 프로토콜과 OAM 관련 패킷을 비포워딩 패킷으로 정의하고, 이를 처리하는 라인 인터페이스와 호스트 프로세서에서의 소프트웨어 구조와 프로세서간 통신 구조를 제시한다

본 논문의 구성은 다음과 같다.

서론에 이어 2장에서 라우터의 기본 구조를 설명하고, 3장에서는 비포워딩 패킷과 포워딩 패킷에 따른 처리 경로, 이를 위한 라인 인터페이스와 호스트 프로세서에서의 소프트웨어 구조 및 프로세서간 통신 구조에 대해 설명하고 끝으로 4장에서 결론을 맺는다.

## 2. 라우터의 기본 구조

라우터는 패킷 포워딩을 처리하는 라인 인터페이스와 라우팅 프로토콜과 OAM을 처리하는 호스트 프로세서, 라인 인터페이스에서 처리된 패킷을 스위칭하는 스위치 패브릭으로 구성되어 있다. (그림 1)은 고속 라우터의 기본구조를 나타낸다.



(그림 1) 고속 라우터 구조

라인 인터페이스는 물리적으로 외부 전송 시스템과 연결되는 접속장치와 포워딩 패킷의 헤더를 분석한 후 라우팅 룩업을 수행하여 포워딩 처리하는 네트워크 프로세서(NP : Network Processor), 포워딩 정보관리 기능 등으로 구성된다. 포워딩 정보관리 기능은 호스트 프로세서에서 생성한 라우팅 정보를 라인 인터페이스에서 룩업이 가능한 FIB로

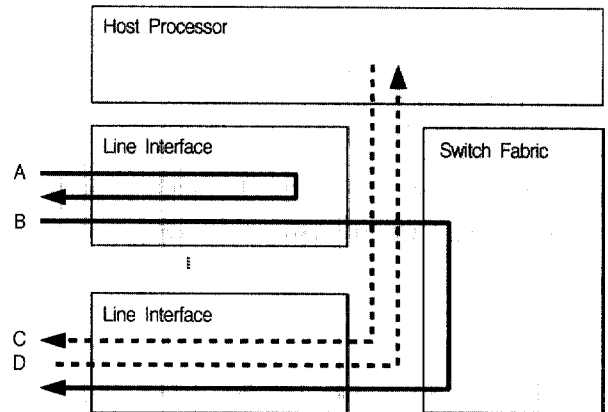
변경하는 역할을 한다. 스위치 패브릭은 포워딩 패킷을 처리하는 라인 인터페이스간의 패킷 스위칭 기능을 제공한다. 호스트 프로세서는 라우터의 기본관리 기능과 라우팅 프로토콜 기능을 수행하며 라우팅 정보를 관리한다.

라우팅 프로토콜은 동일한 도메인 혹은 다른 도메인의 라우터들과 경로 정보를 교환하며 최적의 라우팅 경로를 결정한다. 일반적으로 도메인 내의 라우팅 정보는 OSPF(Open Shortest Path First), IS-IS(Intermediate System to Intermediate System) 라우팅 프로토콜에 의해서 만들어지며 도메인간의 라우팅 정보는 BGP(Border Gateway Protocol)에 의해서 만들어진다[7, 8]. 또한 멀티캐스팅 라우팅을 위한 정보는 PIM(Protocol Independent Multicast), MOSPF(Multicast Extensions to OSPF)를 이용하여 생성한다[8-10]. 관리 기능은 원격 로그인 기능(Telnet), 파일전송(FTP : File Transfer Protocol), 시스템 구성 기능, 장애 관리 기능, 보안 감시 기능 등이 포함된다[11, 12].

## 3. 비포워딩 패킷 처리

### 3.1 포워딩 패킷과 비포워딩 패킷

라우터에서 처리하는 패킷을 포워딩 패킷과 비포워딩 패킷(Non-Forwarding Packet)으로 분류할 수 있다. (그림 2)에서 라우터가 처리하는 포워딩 패킷과 비포워딩 패킷의 처리 경로를 도시하였다.



(그림 2) 패킷 처리 경로

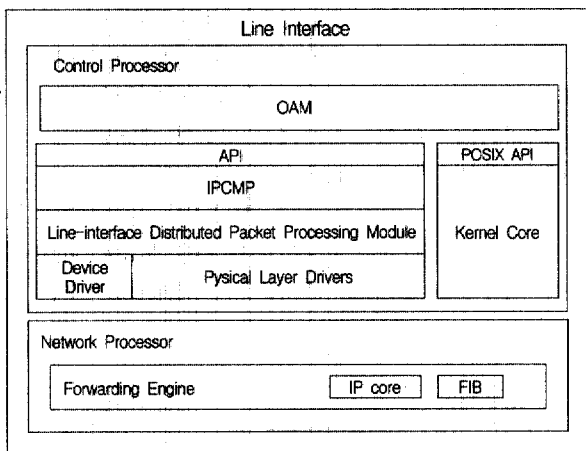
포워딩 패킷은 라인 인터페이스가 헤더를 분석한 후, 라우팅 룩업을 수행하여 포워딩 처리 가능 패킷으로 분류된 패킷이다. 포워딩 패킷은 네트워크 프로세서의 성능에 따라 처리 속도가 영향을 받게 된다. (그림 2)의 A, B 경로가 포워딩 패킷의 처리 경로이다.

패킷의 인터넷 프로토콜(IP : Internet Protocol) 목적지 주소가 라우터로 설정되어, 외부의 라우터 또는 외부 호스트로부터 전송되는 OSPF, BGP, RIP(Routing Information Protocol) 등의 라우팅 프로토콜 패킷, SNMP(Simple Net-

work Management Protocol)와 같은 망 관리용 프로토콜 패킷과 에러가 발생된 패킷은 라인 인터페이스에서 포워딩 되지 않고 호스트 프로세서로 패킷을 전달해야 할 필요가 있다. 이러한 패킷을 비포워딩 패킷으로 정의한다. (그림 2)의 C,D 경로가 비포워딩 패킷의 처리 경로이다. C경로는 에러 보고가 필요한 패킷으로 해당 목적지로 전송되는 경우와 라우팅 패킷이 외부 라우터로 전송되는 경우이다. D는 호스트 프로세서가 처리하는 비포워딩 패킷이 입력되는 경로이며 이 경로를 통해 외부에서 전달되는 라우팅 프로토콜 패킷, 에러 보고용 패킷이 처리된다. 다음 절에서 비포워딩 패킷을 처리하는 소프트웨어 구조를 살펴본다.

3.2 소프트웨어 구조

비포워딩 패킷을 처리하는 라인 인터페이스의 소프트웨어 구조를 (그림 3)에 도시하였다. 라인 인터페이스의 네트워크 프로세서는 외부에서 수신된 인터넷 프로토콜 패킷의 버전 검사, 체크섬 검사, 서비스 타입(TOS : Type of Service) 검사, TTL(Time To Live) 검사를 수행하고 IP 목적지 주소를 해석한 후, FIB를 이용하여 라우팅 룩업을 수행하여 패킷을 포워딩 처리한다.



(그림 3) 라인 인터페이스의 소프트웨어 구조

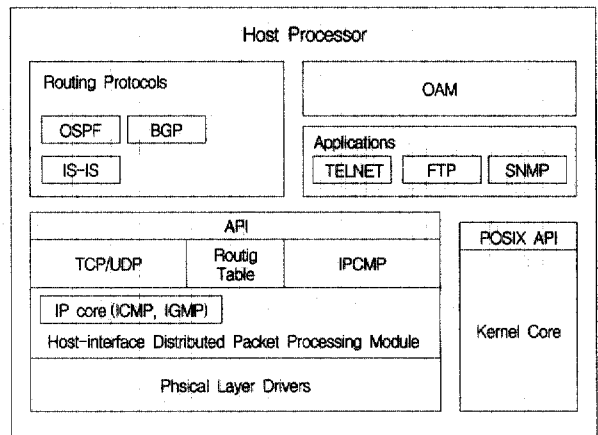
IP 목적지 주소가 라우터의 주소와 일치하는 패킷, IP 목적지 주소가 브로드캐스트 주소인 패킷, IP 목적지주소가 멀티캐스트 주소로 라우터에서 처리해야 하는 비포워딩 패킷은 디바이스 블록을 통해 분산 패킷 처리(Distributed Packet Processing) 모듈로 전달된다. 또한 라우팅 룩업이 실패한 경우에는 인터넷 제어 메시지 프로토콜(ICMP : Internet Control Message Protocol)의 에러 메시지를 생성하기 위해 분산 패킷 처리 모듈로 전달된다.

분산 패킷 처리 모듈은 디바이스 드라이버 블록을 통해 전달 받은 비포워딩 패킷을 처리한다. 먼저 패킷의 종류를 파악한 후, 라인 인터페이스의 분산 패킷 처리 모듈에서 처리할 것인지 혹은 호스트 프로세서의 분산 패킷 처리 모듈

에서 처리할 것인지 결정한다. 패킷이 인터넷 제어 메시지 프로토콜 패킷인 경우 메시지의 종류에 따라 처리 여부를 결정하여 처리가 불가능한 메시지인 경우에는 비포워딩 처리되는 인터넷 프로토콜 패킷과 함께 호스트 프로세서로 전달한다

OAM 모듈은 호스트 프로세서에서 IPCMP를 통해 전달하는 라인 인터페이스 제어용 메시지를 처리한다. IPCMP는 프로세서간 제어용 메시지의 송수신 수단으로 사용되는 것으로서 다음절에서 그 내용을 다룬다.

(그림 4)는 호스트 프로세서의 소프트웨어 구조이다. 디바이스 드라이버는 라인 인터페이스에서 전달하는 비포워딩 패킷을 분산 패킷 처리 모듈로 전달한다. 분산 패킷 처리 모듈은 패킷을 분류하여, 인터넷 프로토콜 패킷인 경우 패킷의 재조립, 옵션 처리를 수행하고 전송 제어 프로토콜인 TCP(Transmission Control Protocol) 또는 UDP(User Datagram Protocol)로 전달한다. 인터넷 제어 메시지 프로토콜 패킷인 경우에 분산 패킷 처리 모듈은 에코 응답(Echo Reply) 메시지, Redirection 메시지, Source Quench 메시지, 타임 스탬프 응답(Time Stamp Reply) 메시지를 처리하게 된다.



(그림 4) 호스트 프로세서의 소프트웨어 구조

3.3 프로세서간 통신 구조

라우터의 구조는 단일 프로세서-공유 버스(Single Processor-Shared Bus) 구조의 1세대 라우터, 포워딩 기능을 분산시킨 다중 프로세서-공유 버스(Multiple Processor-Shared Bus) 구조의 2세대 라우터, 2세대 라우터의 병목 구간인 공유 버스를 스위칭 패브릭으로 대체한 다중 프로세서-스위칭(Multiple Processor-Space Switching) 구조의 3세대 라우터로 그 단점을 보완하며 개선되어 왔다[3, 13].

현재는 인터페이스 카드에 라우팅 룩업, 포워딩을 처리하는 네트워크 프로세서를 사용하여 고속의 포워딩만을 처리하도록 하고 인터페이스간 패킷 스위칭은 스위칭 패브릭에서 처리하는 구조를 채용하고 있다.

이러한 구조에서는 라인 인터페이스와 호스트 프로세서

간 통신을 위한 IPC(Inter Processor Communication) 버스가 필요하다. 일반적인 IPC 방법으로 셀 버스(Cell Bus), HDLC(High Level Data Link), VME(Versa Modular Euro-card) 버스, ATM(Asynchronous Transfer Mode) 스위치 버스, 이더넷(Ethernet) 스위치 버스가 있다. 이 중에서 셀 버스와 VME 버스, HDLC 방식은 공유버스 방식으로 일대일 통신 방법을 사용하는 ATM 스위치 버스, 이더넷 스위치 버스에 비해 처리 성능이 떨어진다[14, 15]. 이더넷 스위치를 사용하는 경우에는, UDP/IP를 이용하여 간단하게 IPC를 구현할 수 있다[16].

라우팅 정보를 각 라인 인터페이스에 전달하기 위해서는 일대일 통신방식뿐 아니라 일대다 통신방법도 필요하다 따라서 UDP 방식의 비연결형 통신이 적합하며 제어용 메시지의 신속한 처리를 위해서 TCP/IP 또는 UDP/IP의 두 계층을 사용하는 IPC 방법 보다 하나의 프로토콜 계층에서 처리하는 것이 바람직하다. (그림 3), (그림 4)에서의 IPCMP(Inter Processor Communication Message Protocol)가 이러한 목적을 위해 사용된다. 기존의 TCP/IP 또는 UDP/IP 방법은 네트워크 계층인 IP 주소를 사용하여 패킷 전달을 처리하는 것으로 동일한 이더넷망에서 통신을 하는 내부망에서는 IP 주소를 사용하는 통신이 비효율적이다. 이에 반해 IPCMP는 라인 인터페이스의 맥(MAC) 주소를 이용하여 목적지, 근원지 주소를 설정하고, 데이터를 바로 이더넷 프레임에 인캡슐하여 송수신하게 함으로써 IP 기능을 사용하지 않는다. 이로 인해 IPCMP를 이용하는 내부 제어용 메시지는 보다 신속한 처리가 가능하게 되었다. IPCMP를 사용하기 위해 이더넷의 타입 필드에는 내부적으로 정의한 값을 이용하게 되며 (그림 5)와 같은 IPCMP의 헤더를 사용한다.

16-bit Source Port Number	16-bit Destination Port Number
16-bit IPCMP length	Undefined
data	

(그림 5) IPCMP 헤더

소스 포트 번호(Source Port Number)는 해당 근원지 응용프로그램의 포트 번호이며, 목적지 포트 번호(Destination Port Number)는 목적지 응용 프로그램의 포트 번호로서 메시지가 수신될 응용 프로그램을 명시한다.

IPMP는 사용자 인터페이스로 소켓 인터페이스를 제공하였으며 (그림 6)에 IPCMP를 사용한 응용 프로그램의 한 예를 나타내었다. 기존의 TCP/UDP 소켓에서는 AF\_INET을 사용하였으나 IPCMP 소켓을 사용하기 위해 "socket(AF\_IPCMP, SOCK\_DGRAM, 0);"과 같이 새로 정의된 AF\_IPCMP를 이용하여 소켓을 생성하고 이를 이용하여 응용 프로그램간 통신을 수행한다.

```

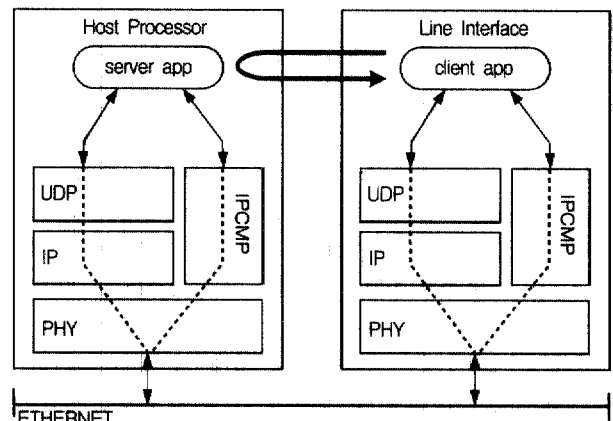
...
if (( sockfd = socket(AF_IPCMP, SOCK_DGRAM, 0) ) < 0) {
    printf("Can't open IPCMP socket\n");
    exit(1);
}
...
while ( tmp_count < max_packet){
    slen = sendto( sockfd, (char *)&s_buf,
        sizeof(msg), 0,(struct sockaddr *)&dst_addr, da_len);
    if (slen < 0) {
        printf("SENDTO ERROR\n");
        close(sockfd);
        exit(1);
    }
    rlen = recvfrom(sockfd, r_buf, 1500, 0,
        (struct sockaddr *)&src_addr, &sa_len);
    if (rlen < 0){
        printf("RECVFROM ERROR\n");
        close(sockfd);
        exit(1);
    }
    ...
}
...

```

(그림 6) IPCMP를 사용한 코드 예

### 3.4 IPC의 성능비교

IPMP를 적용한 호스트 프로세서와 라인 인터페이스간의 IPC 성능과 기존의 UDP/IP를 적용한 IPC 처리 성능을 비교하였다. 호스트 프로세서는 모토롤라 PowerPC MPC750 (300MHz) CPU를, 라인 인터페이스는 IBM사의 PowerPC 405GP(266MHz) CPU를 탑재하고 있으며 운영체제로 리눅스(Linux) 커널 버전 2.2.13을 사용하였다[17, 18]. 오픈 소스인 리눅스의 네트워크 커널에 IPCMP를 구현하여 내부 IPC 통신을 실현하였다. (그림 7)은 IPC 성능을 비교하기 위해 구성한 호스트 프로세서, 라인 인터페이스의 내부 구조도이다.



(그림 7) IPCMP 시험 구성도

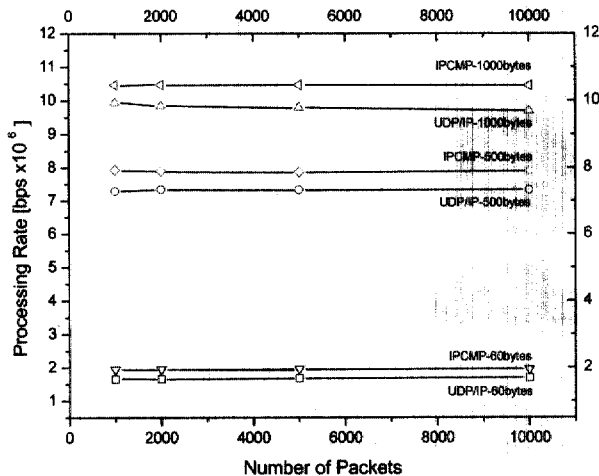
(그림 7) 환경에서와 같이 가변적인 크기의 패킷을 생성하여 전송하는 클라이언트 어플리케이션과 패킷을 수신하여 패킷의 소스 노드로 다시 보내주는 서버 어플리케이션

<표 1> 패킷 라운드 트립 시간 측정 결과

Type	UDP/IP								
Bytes	60			500			1000		
Number of Packet	1000	5000	10000	1000	5000	10000	1000	5000	10000
Avg. RTT(msec)	289.83	1428.56	2845.32	548.52	2727.79	5459.54	804.63	4083.43	8250.13
Type	IPCOMP								
Bytes	60			500			1000		
Number of Packet	1000	5000	10000	1000	5000	10000	1000	5000	10000
Avg. RTT(msec)	246.63	1225.71	2432.95	504.16	2541.01	5057.31	764.13	3820.52	7653.78

을 이용하여 패킷의 라운드 트립 시간(RTT: Round Trip Time)을 측정하였다. 60바이트, 500바이트, 1000바이트 패킷을 각각 1,000개, 5,000개, 10,000개씩 전송하여 IPCMP와 UDP/IP의 성능을 비교한 RTT 측정 결과를 <표 1>에 나타내었다.

RTT 측정 결과를 보면 IPCMP를 사용한 경우가 UDP/IP를 이용한 경우에 비해 평균 9.3% 정도 RTT가 줄어들음을 보여주었다(패킷 크기가 60바이트인 경우 평균 14.2%, 500바이트인 경우 평균 7.4%, 1000바이트인 경우 평균 6.2%). 이는 내부 프로세스 통신을 위해 두 계층에서 처리하는 패킷 프로세싱 절차를 하나의 계층에서 처리함으로써 얻는 이득이다. (그림 8)은 IPCMP 처리 성능을 UDP/IP 방식과 비교한 것으로 패킷 처리 능력이 평균 10.3% 향상됨을 보여준다. 60바이트의 패킷을 IPCMP로 처리할 경우 처리속도가 1.95Mbps, UDP/IP로 처리할 경우 1.67Mbps로 평균 16.7%의 처리 향상 결과를 보였다. 또한 500바이트 패킷의 경우 평균 7.9%, 1000바이트 패킷의 경우 평균 6.5%의 처리 향상 결과를 보인다.



(그림 8) IPCMP 처리 성능 비교

IPCOMP 처리 성능과 함께, IPCMP가 호스트 프로세서/라인 인터페이스의 프로세서간 데이터 송수신에 적합한 패킷

손실율을 가지는지 확인하기 위해 10Mbps의 이더넷을 지원 하는 호스트 프로세서에서 IPCMP 패킷의 손실율을 측정하였다. 패킷 발생 장치를 사용하여 60바이트 IPCMP 패킷을 1Mbps, 5Mbps, 8Mbps 속도로 발생시켜 손실율을 측정 한 결과, 패킷 전송 속도가 8Mbps인 경우 패킷의 손실은 없으며 8.3Mbps인 경우는 120개의 패킷 손실, 8.5Mbps인 경우는 약 0.4%의 패킷 손실을 나타내었다. 또한 UDP/IP를 이용하는 경우에도 이와 유사한 패킷 손실이 발생함을 확인 하였다.

FIB를 구성하는 라우팅 정보 엔트리는 일반적으로 32바이트의 크기를 가지며 최대 십만개의 FIB 엔트리를 동시에 전송할 경우 약 3.2Mbytes의 정보가 송신되게 된다. 따라서 FIB를 포함한 OAM 정보를 IPCMP를 이용하여 패킷 손실 없이 처리할 수 있게 된다.

4. 결 론

본 논문에서는 라우팅 정보를 관리하는 유니캐스트 및 멀티캐스트 라우팅 프로토콜과 OAM 관련 패킷을 비포워딩 패킷으로 정의하고, 이를 처리하는 라인 인터페이스와 호스트 프로세서에서의 소프트웨어 구조와 프로세서간 통신 구조를 제시하였다. 이러한 소프트웨어 구조와 프로세서간 통신 구조는 라인 인터페이스로부터 입력되는 라우팅 정보를 기반으로 구성되는 포워딩 정보의 신속한 전달이 필요한 고속 라우터 및 고속 스위치 시스템에서 요구되는 기술이다.

제시한 프로세서간 통신 구조는 기존의 방법에 비해 평균 10%의 처리 성능 향상을 보였으며 이 방법을 사용하여 신속한 내부 데이터의 처리와 라인 인터페이스 소프트웨어 구조의 불필요한 요소를 제거하여 소프트웨어의 비용을 최소화하였다.

인터넷의 확산으로 포워딩 패킷뿐 아니라 비포워딩 패킷도 함께 증가 하고 있으며 대용량의 고속 라우터 시스템 설계를 위해서는 비포워딩 패킷 트래픽에 대한 정확한 분석이 필요하다. 이와 관련하여 라우팅 프로토콜 트래픽과 FIB의 효율적인 분산 처리에 대한 연구를 지속적으로 수행할

예정이며 신뢰성 있는 내부 통신을 위해 흐름 제어(flow control)등의 IPCMP 부가 기능 추가에 대해 연구가 진행 중이다.

**참 고 문 헌**

[1] A. Bharagava and B. Bhargava, "Measurements and quality of service issues in electric commerce software," in Proc. Application-Specific System and Software and Technology, pp.26-33, 1999.

[2] S. Keshav, R. Sharma, "Issues and trends in router design," IEEE Communications Magazine, Vol.36, pp.144-151, May, 1998.

[3] V. P. Kumar, T. V. Lashman, D. Stiliadis, "Beyond Best Effort: Router Architectures for the Differentiated Services of Tomorrow's Internet," IEEE Communications Magazine, Vol.36, pp.152-164, May, 1998.

[4] N. McKeown, P. Gupta and S. Lin, "Routing lookups in hardware at memory access speeds," INFOCOM '98, Proceedings, IEEE, Vol.3, pp.1240-1247, 1998.

[5] S. Nilsson, G. Karlsson, "IP-address lookup using LC-tries," Selected Areas in Communications, IEEE Journal on, Vol.17, pp.1083-1092, June, 1999.

[6] C. Donpaul, et al, "Implementing scheduling algorithms in High-Speed Networks," Selected Areas in Communications, IEEE Journal on, Vol.17, pp.1145-1158, June, 1999.

[7] J. Moy, OSPF Version 2, RFC2328, April, 1998.

[8] Christian Huitema, "Routing in the Internet," 2nd Edition, Prentice Hall, 1999.

[9] D. Estrin, et al, "Protocol Independent Multicast-Sparse Mode (PIM-SM) : Protocol Specification," Internet Request For Comments RFC 2362, June, 1998.

[10] J. Moy, "Multicast Extensions to OSPF," Internet Request For Comments RFC 1584, March, 1994.

[11] F. Baker, "Requirements for IP Version 4 Routers," RFC 1812, June, 1995.

[12] Wang-Bong Lee, et al, "An Architecture of Distributed Multi-Gigabit IP Router," AIC 24th Conference, Seoul, Nov., 2000.

[13] A. Asthana, C. Delph, H. Jagdish and P. Kryzyanowski, "Towards a gigabit IP router," Journal of High-Speed Networks, Vol.1, No.4, 1992.

[14] J. Furnuas, et al, "A prototype for interprocess communication support, in hardware," Ninth Euromicro Workshop on Real-Time Systems, pp.18-24, 1997.

[15] Bup Joong Kim, et al, "Designed and Implementation of IPC Network in ATM Switching system," ICATM, pp. 148-152, 2001.

[16] W. R. Stevens, TCP/IP Illustrated, Addison-Wesley, New York, Vol.1, 1996.

[17] The Linux kernel Archives, <http://www.kernel.org>.

[18] LinuxPPC, <http://www.linuxppc.org>.



**이 왕 봉**

e-mail : leewb@etri.re.kr

1996년 숭실대학교 정보통신공학과(공학사)  
 1998년 숭실대학교 전기전자학부(공학석사)  
 1998년~1999년 현대정보기술 연구원  
 1999년~현재 한국전자통신연구원 선임 연구원 재직

관심분야 : 고속 라우터, 차세대인터넷, 멀티캐스팅 통신, 모바일 인터넷



**정 영 식**

e-mail : yschung@etri.re.kr

1983년 홍익대학교 전자계산학과(공학사)  
 1993년 한남대학교 전자계산공학과(공학석사)  
 1983년~현재 한국전자통신연구원 책임 연구원 재직

관심분야 : 고속 라우터, 시스템 프로그래밍

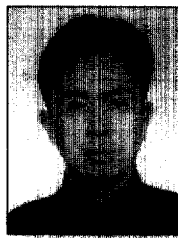


**김 태 일**

e-mail : ttikim@etri.re.kr

1983년 숭실대학교 전자계산학과(공학사)  
 1992년 정보처리기술사 취득  
 1983년~현재 한국전자통신연구원 책임연구원, 라우팅프로토콜팀장

관심분야 : 고속 라우터, 지능망, 망관리, 개방형 네트워크



**방 영 철**

e-mail : ybang@kpu.ac.kr

1994년 University of Oklahoma 전산학(이학사)  
 1997년 University of Oklahoma 전산학(이학석사)  
 2000년 University of Oklahoma 전산학(이학박사)

2000년~2002년 한국전자통신연구원 선임연구원  
 2002년~현재 한국산업기술대학교 컴퓨터공학과 전임강사  
 관심분야 : 고속 라우터, 차세대인터넷, 유니/멀티캐스팅통신, 모바일인터넷